Universitetet
i Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER THESIS

Study program/Specialization:

Master of Science /

Industrial Economy

Spring semester 2023

[✓ Open] or Restricted access

Writer(s): Vemund Edvard Refnin, Markus Aarekol Johannessen

Academic Administrator: Sigbjørn Landazuri Tveteraas

Supervisor(s): Atle Øglend

Thesis title: Can AIS data be used to predict fish processing at Grieg Seafood Stjernelaks?

Credits: 30

Keywords:

AIS, Python, Machine Learning, Classification,

Atlantic salmon, Norwegian fish farming industry.

Number of Pages: 130

Place and date:
Stavanger, 15-06-2023.

Preface

This Master's thesis signifies the conclusion of our academic pursuits at the University of Stavanger. Throughout our studies, we have completed a B.Sc. in Computer Science and an M.Sc. in Industrial Economics. The research conducted for this thesis is inherently technical in nature and bridges our knowledge in Computer Science intertwined with the economic incentive that underpins the inquiries under investigation.

We express our gratitude towards Ragnar Tveterås and Eivind Helland at Blue Planet and our supervisor, Atle Øglend, for their guidance and sound advice. Their proficiency and commitment have been instrumental in the realization of this thesis.

Our thanks also extend to Lars Martin Hetland from Grieg Seafood Stjernelaks for his cooperation and support. His valuable insights into the Norwegian fishing farm industry and the provision of crucial label data significantly augmented the depth and authenticity of our research.

We sincerely believe that the outcomes of our research and the associated discussion encapsulated in this thesis will make a meaningful contribution to academic studies regarding the applications of AIS data within the Norwegian fishing farm industry.

Abstract

This study investigates the potential of using Automatic Identification System (AIS) data to predict fish processing at Grieg Seafood Stjernelaks. The research involved creating and analyzing two processed datasets: *Labeled Days* and *Labeled Time Series*. The *Labeled Days* dataset uses the $Active$ label, indicating the days when fish processing occurred, while the *Labeled Time Series* dataset uses the $Direct$ label, indicating the specific times when fish was directly delivered by relevant vessels. Machine learning techniques, including feature engineering, decision trees, random forests, and dynamic time warping, were used to analyze the AIS data.

The results of this study highlight that the *Labeled Days* baseline utilizing temporal patterns to predict the activity status for Stjernelaks perform *excellent* in terms of Area Under the ROC Curve (*AUC-ROC*) score. However, the best machine learning model, 'Rand RFE RF,' outperforms the baseline by utilizing AIS data with an *AUC-ROC* score of `0.933`. No model outperformed the baseline for the *Labeled Time Series* dataset.

The study concludes that while AIS data shows promise in predicting if Stjernelaks is processing fish on any given day, it does not conclusively prove that AIS can be used to predict fish processing at Grieg Seafood Stjernelaks. The research faced limitations due to issues encountered with Kystdatahuset's API endpoint for fetching AIS data, and the scarcity of label data. These limitations may have affected the ability to fully answer the research question and should be addressed in future research.

Contents

Chapter 1

Introduction

This thesis was written in cooperation with the Norwegian aquaculture company *Blue Planet*[1]. We got in contact with Eivind Helland and Ragnar Tveterås from Blue Planet in the autumn of 2022. They pitched their hypothesis that publicly available *AIS*[2] data might be used to predict the future supply of processed fish entering the market from various actors in the Norwegian fish farming industry. Given the lack of extensive prior research on the applications of AIS, creating a fully functional product capable of performing such predictive analysis is quite extensive. The work needed to acquire enough data, clean it and then create a product capable of such analysis presents a considerable challenge. The extensive endeavor extends beyond the time constraints of a single-semester master's thesis. Therefore some limitations were made early on in close collaboration with Blue Planet, but also with our master's thesis supervisor Atle Øglend. This thesis is therefore to be considered as a *Proof of Concept*[3] for their theory and as a starting point for Blue Planet as they plan to start developing the product in the near future.

[1] Blue Planet AS is a Norway-based company offering consulting services and business development assistance to global seafood businesses, with a particular focus on sustainable fish production and fostering connections within the aquaculture industry [Blue Planet, 2023a].

[2] Automatic Identification System (AIS) is a tracking system used on ships and by vessel traffic services (VTS) for identifying and locating vessels by electronically exchanging data with other nearby ships, AIS base stations, and satellites [Kjerstad, 2022].

[3] A proof of concept (PoC) is a demonstration, typically small-scale, showing that a proposed idea, design, method, or technology is feasible and can potentially be developed into a working model or product [Kendig, 2015].

Consequently, the research question for this thesis and the concept we will attempt to acquire proof of is:
Can AIS data be used to predict fish processing at Grieg Seafood Stjernelaks?
If we successfully establish grounds for this research question, further work, and analysis may be built on top of the findings from this thesis. For Blue Planet, either answer to the research question will provide them value.

To answer our research question, it was further split into two sub-research questions. These sub-research questions (Sub-RQs) were formulated after carefully inspecting the available data provided by Grieg Seafood Stjernelaks:

- **Sub-RQ 1:** Can AIS data be used to predict if Stjernelaks is processing fish on any given day, regardless of the source being a waiting cage[4] or direct vessel delivery?
- **Sub-RQ 2:** Can AIS data be specifically used to predict if Stjernelaks is processing fish that has been directly delivered by a vessel on any given day?

Sub-RQ 1, addresses the broader application of AIS data to predict the activity status of Stjernelaks on any given day, regardless of the source of the fish. This could involve any fish processing activity at Stjernelaks, including those where fish is sourced from waiting cages or delivered directly from vessels. In other words, this sub-question aims to determine the general predictive power of AIS data for fish processing activity.

Sub-RQ 2, on the other hand, focuses on a more specific scenario - predicting Stjernelaks' activity status based on fish directly delivered by vessels. This sub-question aims to understand the potential of AIS data in predicting activities specific to fish being delivered directly by a vessel. This more detailed analysis could reveal specific patterns or trends tied to direct vessel deliveries.

This detailed analysis in **Sub-RQ 2** is linked to **Sub-RQ 1** because the activity status hierarchy of Stjernelaks is conditionally related such that if a vessel delivers fish directly, then the activity status of Stjernelaks is active. This is however

[4] Waiting cage refers to a designated enclosure where fish are temporarily held before being processed or transferred to other locations [Lars Martin Hetland Grieg Seafood Stjernelaks, 2023].

not necessarily true for the opposite when fish is not delivered by a vessel, because fish can be processed from the waiting cage instead. This conditional relationship between the **Sub-RQs** is illustrated below in figure 1.1.

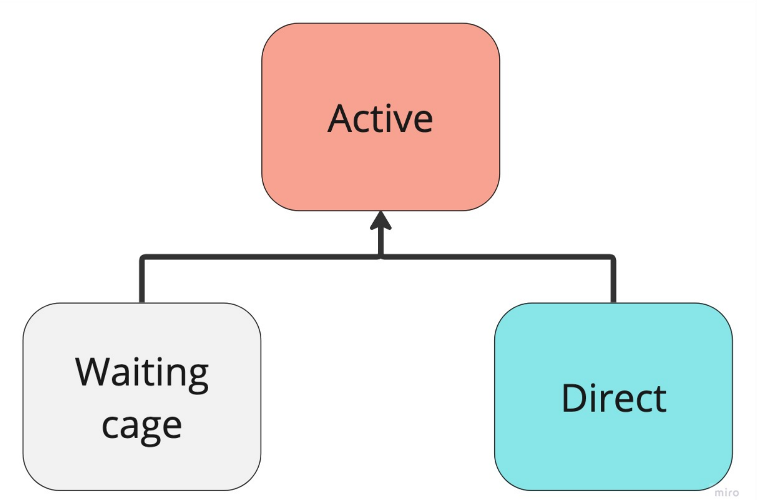


Figure 1.1: Sternelaks' activity status hierarchy.

In sum, these two sub-questions ensure a comprehensive exploration of the utility of AIS data in predicting fish processing at Stjernelaks - both in general and specific contexts - effectively covering the full extent of the research question.

The analysis is conducted such that if we achieve prominent positive results for both sub-questions, then we can conclude convincingly that the research question is fulfilled. Similarly, if we achieve prominent negative results, then we can conclude convincingly that by our suggested solution in this thesis, AIS can not be used to predict fish processing at Grieg Seafood Stjernelaks.

1.1 Motivation

A product that can predict the future supply of processed fish using publicly available AIS data, as hypothesized by Blue Planet, could be valuable for various stakeholders. For example, it could be helpful for fish processors and distributors, who could use the predictions to plan their production and supply chain operations. It could also improve the understanding of the environmental impact of fish transport. Additionally, this information could be helpful for policymakers and regulators, who could use it to monitor and manage the Norwegian fish farming industry and to ensure sustainable and responsible fishing practices. It could also be valuable for researchers and scientists, who could use it to study the dynamics of the aquaculture industry and the impact of aquaculture on the marine ecosystem. Overall, accurately predicting the volume of fish processed in Norway could be a valuable tool for supporting the sustainable and responsible management of the aquaculture industry. These are all AIS applications that can potentially be built around the topic that we, for this thesis, will attempt to acquire proof for.

1.2 Outline

The rest of this thesis is outlined as follows:

Chapter 2, Background: Introduces the reader to Grief Seafood Stjernelaks, the Norwegian fish farming industry, and to AIS data.

Chapter 3, Litterature Review: Describes relevant research to this thesis.

Chapter 4, Theory: Presents perhaps unknown relevant theory and terminology to the reader.

Chapter 5, Data: Illustrates the extensive data handling process behind the analysis.

Chapter 6, Methodology: Provides insight to the reader into how the analysis was conducted.

Chapter 7, Results: Presents the results of performing the analysis to the reader.

Chapter 8, Discussion: Discusses the results and connects findings to the research question.

Chapter 9, Conclusion: Concludes the thesis, introduces further work based on findings, and states the thesis' contributions.

Chapter 2

Background

2.1 Grieg Seafood Stjernelaks

Grieg Seafood Stjernelaks is a fish processing facility located at Helgøy island situated in Boknafjorden, Rogaland. Stjernelaks is one of five fish processing facilities in Rogaland, and in 2019, Stjernelaks was responsible for 1.72% of all fish processed in Norway [SSB, 2023].

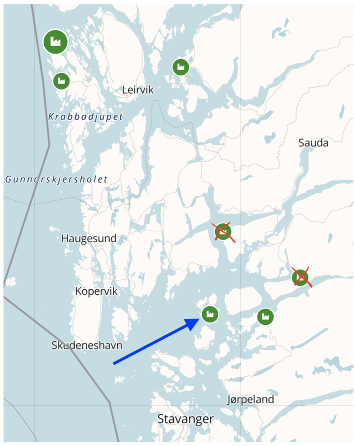


Figure 2.1: Processing facilities in Rogaland. Stjernelaks marked with the blue arrow. The crossed-out locations are not relevant [Blue Planet, 2023b].

We visited Stjernelaks in the early spring of 2023 and were welcomed by Lars Martin Hetland, Fishery Processing Manager of Stjernelaks. Lars Martin provided valuable insight into how the processing facility Stjernelaks operates and the rest of the Norwegian fish farming industry. We were given a guided tour of the facility, where we observed the entire process of gathering fish from waiting cages, processing it with automatic and manual labor, and the final packaging of the processed fish. Lars Martin did also provide us with historical fish processing data from Stjernelaks (further discussed in Chapter 5) that functions as *label data*[1] for the analysis presented in Chapter 6. The thought process and implementation process in this thesis are highly influenced by the insight gained from our visit to Stjernelaks, and the historical data provided to us by Lars Martin proved to be essential for our analysis. Lars Martin also established a line of communication for us to the captain of the wellboat *Ronja Polaris*[2]. From conversations with a vessel captain involved with transporting live fish in the Norwegian fish farming industry, we gained a lot of valuable knowledge regarding vessel movement and the delivery of fish.

2.2 The Fish Farming Industry in Norway and the World

Fish farming has been practiced for several thousand years. In Asia, species such as Carp and Tilapia have been cultivated for over 2000 years [Towers, 2010]. Today, fish farming of many different species is carried out worldwide. Although Norway exports large quantities of fish, it only accounts for 2.4% of all farmed fish in the world measured in quantity (tons) of fish [Misund, 2023]. China is the world's largest fish farmer, producing over half of the total quantity. Asia accounts for 90% of all fish farming in the world, where Carp is the most common species. However, Norway is the world's largest producer of Atlantic salmon.

[1] Label data refers to the act of assigning specific class labels or categories to individual data instances, enabling the training and evaluation of machine learning models.

[2] Ronja Polaris is a Fish Carrier vessel sailing under the flag of Norway, built in 2013, with a length of 75.8 meters and a breadth of 16 meters, and it provides real-time data about its location, status, and voyage details through the Automatic Identification System. [MarineTraffic, 2023]

There are different technologies for how to produce salmon, and the technology is constantly evolving. What is common for the different production technologies is that the fish must go through the same phases. According to Misund, the first phase is *broodstock production*. Here, the parent fish for new generations of farmed salmon are selected by mixing *milt and roe*[3]. After hatching, carefully selected offspring from the broodstock production are moved on to the smolt production. Here, the aim is to ensure that the osprey grows and eventually smoltify. This process takes place in freshwater. When the smoltification is completed, the salmon is called smolt and is ready to live in saltwater. The final phase is called fish production. The goal here is to feed the salmon to a certain size that can be sold further. Figure 2.2 below depicts the life stages of an Atlantic Salmon.

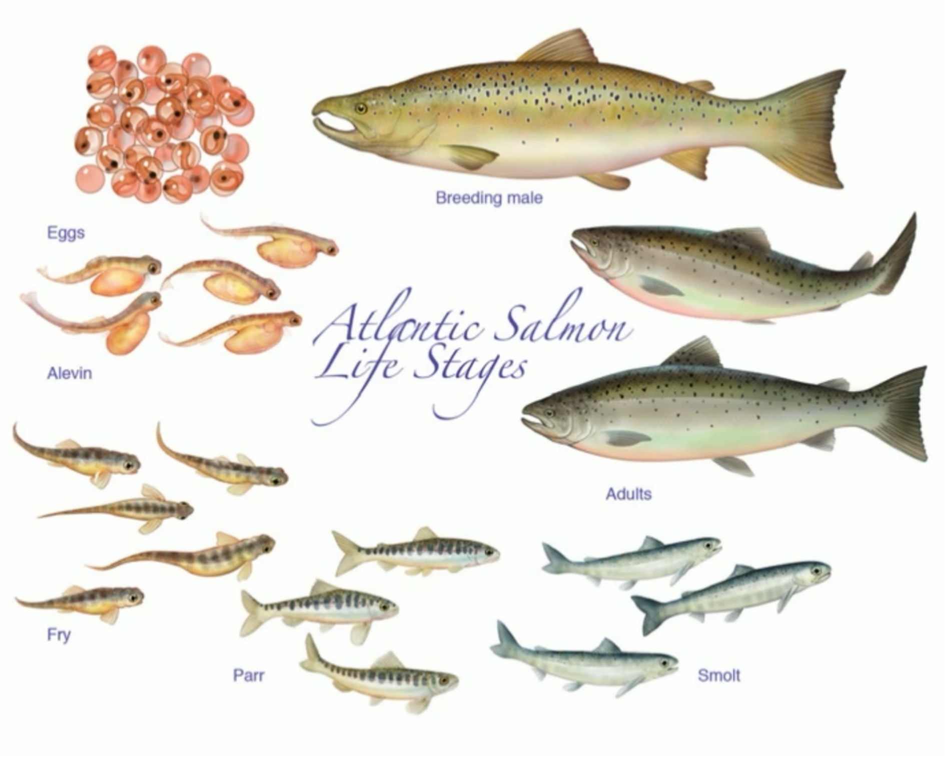


Figure 2.2: Atlantic salmon life stages [Harmon, 2011].

[3] Milt refers to the sperm of male Atlantic salmon, while roe refers to the eggs of female Atlantic salmon.

According to Misund, *intensive farming* is the prevalent method for salmon farming in Norway. Here, the fish live with high density in open or closed cages or closed tanks on land. The fish are fed and are constantly under human control. Intensive salmon farming is divided into various categories based on the production method. These include conventional facilities in the sea, exposed facilities far out to sea, closed facilities in the sea, and closed facilities on land. The environment, species, technology, and capital all play a role in determining what kind of farming facility is used. In Norway, conventional facilities have the longest history.

2.2.1 Conventional Salmon Farming

In conventional salmon farming, the fish start their lives in tanks on land before being transferred to open-sea cages. See figure 2.3 for an example of a conventional sea-based facility. Salmon osprey lives in freshwater for 8 to 18 months until they reach a weight of around 100 grams. They are fully smoltified at this weight and spend the rest of their lives in saltwater. The sea phase lasts for 12 to 18 months, depending on when they reach the desired weight for processing. This typical weight is between 3 and 6 kg.

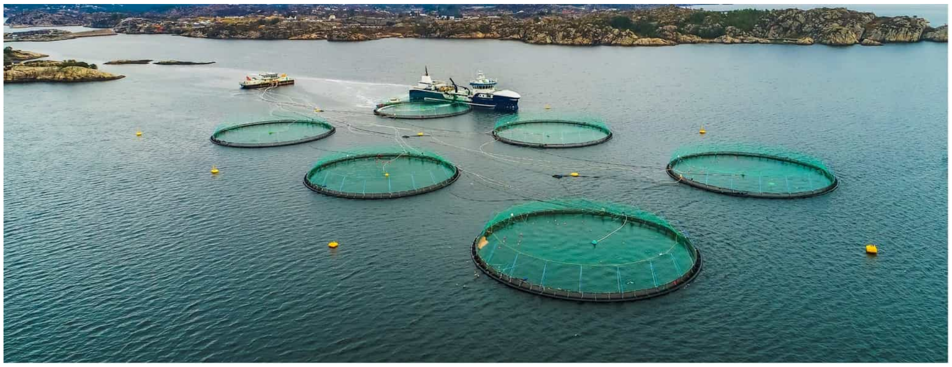


Figure 2.3: Conventional sea-based fishing facility [Misund, 2023].

The open sea cages provide for a natural flow of seawater, so no energy is required to move water. However, there is a risk of potential exchange of infections with the environment outside the cage and waste products from fish, feed, and treatments affect the environment outside the cage. In addition, there is a risk of escape from farmed salmon due to wear on the walls. Although farmed salmon originated from wild salmon in Norwegian rivers and share the same genes, farmed salmon have been selectively bred for traits that should not be mixed with wild salmon. Moreover, having thousands of salmon gathered in confined areas leads to the rapid spread of sea lice. The disadvantages associated with producing salmon in open sea cages have led to exploring possibilities for land-based production.

2.2.2 The Norwegian Fish Farming Industry

The Norwegian fish farming industry is divided into several *production areas*[4], which are geographic regions that the Norwegian government designates to regulate and manage fish farming activities. This thesis will be restricted to production area 2, now referred to as PO2, which is a specific production area within the Norwegian fish farming industry. It is located in western Norway, along the coast of Hordaland and Rogaland counties. This area is characterized by its rugged coastline, deep fjords, and strong ocean currents, which make it well-suited for fish farming. In PO2 the most commonly farmed fish species are Atlantic salmon and Rainbow trout which are mostly raised in conventional sea-based facilities. These fish farms are typically located in sheltered areas within the fjords, which provide protection from strong ocean currents and waves.

One of the key challenges that fish farmers in PO2 face, is the risk of disease outbreaks amongst the fish. To prevent disease outbreaks, it is important that fish farmers closely monitor their water quality and the health of the fish. Disease outbreaks in the fishing farm industry are also something that the Norwegian government highly regulates through the Norwegian Food Authority and the Norwegian Directorate of Fisheries. This is regulated through measures such as strict limits on the number of fish that are allowed per area, as well as requirements towards the use of environmentally friendly technologies and practices.

One way the Norwegian Directorate of Fisheries monitors the various actors in the Norwegian fish farming industry is by issuing permits limited by MTB (Maximum Allowed Biomass), which was introduced in 2005 [Fiskeridirektoratet, 2023]. There are two different MTB permits, one per company level, and one per site level. The MTB system means that the permit holder cannot have a standing

[4] There is a total of 13 production areas in the Norwegian fish farming industry [Regjeringen.no, 2022].

biomass (kilograms of live fish in seawater) exceeding the allowed MTB at the company level. At each site, the biomass cannot exceed the specified MTB for that particular site. The normal size of a permit is 780 tonnes, except for Northern Norway, where the permits have a size of 975 tonnes.

When the fish have reached the desired size and are ready for harvesting, they are transported to specialized fish processing facilities. Here the fish are either placed in waiting cages or directly transferred to a processing and preparation for sale phase. This process typically involves stunning and bleeding the fish, followed by gutting and cleaning. Most of these processes have in recent times become automatic according to Lars Martin Hetland (Grieg Seafood Stjernelaks, 2023). If the fish is placed in waiting cages it is because the processing facility awaits a more optimal time to start the processing of the fish. This could be due to the amount of fish delivered surpassing the capacity of the facility or other optimization factors. If the fish is placed in waiting cages it is usually processed no more than 1-2 days later.

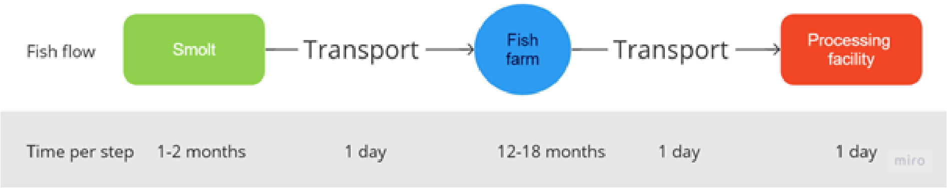
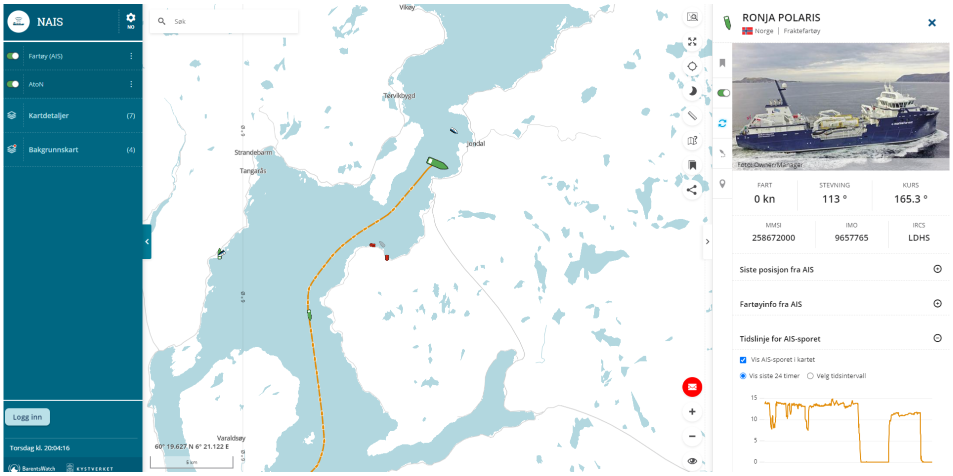


Figure 2.4: An example illustration of a fish's life cycle in the Norwegian fish farming industry. Information gathered from conversations with Grieg Seafood Stjernelaks, 2023.

At the processing facilities, the fish are rarely kept for more than a day before it is sold and transferred to every corner of the world. In figure 2.4 the transportation of fish happens twice during the fish life cycle in the Norwegian fish farming industry. When transported from the smoltification phase to fishing farms, and when transported from fishing farms to processing facilities. On both of these transportation routes, the fish are transported in fish-carrying vessels that are required to use AIS data, which lays the foundation for this thesis.

2.3 AIS - Automatic Identification System

AIS, which stands for Automatic Identification System, is a system used for identifying and tracking the movements of ships [Kjerstad, 2022]. The system consists of a transmitter and a receiver that transmits, amongst other information, the ship's identification, position, speed, and course. Additionally, information such as the type of vessel, destination, and more can also be transmitted. The receiver is often linked to radar and Electronic Chart Display and Information Systems (ECDIS), which allows the names and positions of vessels to be displayed in real-time. An example of this real-time display of name and positioning is illustrated below in figure 2.5 for the vessel Ronja Polaris.



Torsdag kl. 20:04:16

60° 19.627 N 6° 21.122 E

RONJA POLARIS
Norge | Fraktefartøy

FART	STEVNING	KURS
0 kn	113 °	165.3 °

MMSI	IMO	IRCS
258672000	9657765	LDHS

☑ Vis AIS-sporet i kartet
● Vis siste 24 timer ○ Velg tidsintervall

Figure 2.5: AIS data example from Kystverket.no showing real-time positioning of the wellboat Ronja Polaris.

According to Kjerstad, AIS has been mandatory for ships over 300 gross tonnes since 2002, mandated by the International Maritime Organization (IMO). Initially, the system was introduced as a supplement to radar as a collision avoidance system on ships, but it has since become essential for monitoring ship traffic, either through the reception of AIS signals on land-based stations or with special satellites [Kjerstad, 2022]. In later years EU and Norway also made the requirement for AIS equipment on all vessels measuring 15 meters in length and above mandatory.

Abbreviation	Full name	Description
AIS	Automatic Identification System	Maritime safety communications system that provides vessel information.
DWT	Deadweight Tonnage	Measure of vessel maximum carrying capacity.
ETA	Estimated Time of Arrival	Expected time a vessel will arrive at its destination.
IMO	International Maritime Organization	Can also refer to the ship's unique 7-digit IMO number.
MMSI	Maritime Mobile Service Identity	Unique 9-digit number assigned to a maritime radio station or navigation device. Used for communication and identification.
VHF	Very-High Frequency Radio Wave	Range of radio frequencies used for communication between vessels and maritime authorities. Frequency rate AIS systems utilize for data exchange.

Table 2.1: Overivew of AIS abbreviations [Yang, 2019].

The AIS data transceivers consist of two types, classes A and B, these two classes have different amounts of reported data fields and reporting frequencies. A ship's transceiver, which falls under class A, broadcasts information that can be grouped into 11 data fields. These data fields can be further classified into three types: *static information*, *dynamic information*, and *voyage-related information*. See table 2.2 below for a detailed classification and description of these data fields. The dynamic information is automatically transmitted every 2-10 seconds, depending on the ship's speed while it is moving, and every 3 minutes while it is anchored. The static and voyage-related information is broadcasted every 6 minutes, regardless of the navigational status.
Class B transponders, in comparison to Class A transponders, transmit a reduced set of data. They omit the IMO number, draught, destination, ETA, rate of turn, and navigational status. The reporting interval from Class B transponders is also sparser than those of Class A transponders, being a minimum of 5 seconds.

Data Field	Type	Description
AIS identity and location	Static	Maritime Mobile Service Identity (MMSI) and the location of the system's antenna on board.
Ship identity	Static	Ship name, IMO number, type, and call sign of the ship.
Ship size	Static	Length and width of the ship.
Ship position	Dynamic	Latitude and longitude (up to 0.0001 min accuracy).
Speed	Dynamic	Ranging from 0 knots to 102 knots (0.1-knot resolution).
Rate of turn	Dynamic	Right or left (ranging from 0 to 720 degrees per minute.
Navigation direction	Dynamic	Shipping course, heading, and bearing of the ship.
Navigation status	Dynamic	Includes 'at anchor', 'under way using engine(s)', and 'not under command'.
Timestamp	Dynamic	Second field of the UTC time when the subject data packet was generated.
Destination and ETA	Voyage-related	Destination port and the estimated time of arrival of the ship.
Draught	Voyage-related	Ranged from 0.1m to 25.5m. The depth of the ship's hull below the water line.

Table 2.2: Overview of the attributes of AIS data [United Nations Statistics Division, 2023].

It is important to note that raw AIS data may contain noise that may lead to wrong conclusions based on errors and inaccuracies that may still exist in the AIS data. The data most vulnerable to error and inaccuracies is the data that is being manually registered into the system. This data includes static information such as MMSI, ship's width and length, IMO number, name, type, call sign, and voyage-related information, such as the ETA, draught, and intended destination.

Another potential issue when analyzing AIS data is that some smaller vessels, below the 300 gross tonnages limit or the 15-meter limit, in the Norwegian fish fleet may not be required to carry AIS equipment, which may limit the coverage of the data. Another limitation of AIS data is the possibility of errors or inaccuracies in the data itself. This can be caused by technical issues with the AIS equipment, signal interference, or even deliberate manipulation of the data by the vessel operators. Therefore, it is important to carefully validate and clean AIS data to ensure its accuracy and reliability as Yang et al. mentions in their research paper [Yang, 2019].

The main idea behind this thesis is the fact that all of the vessels that carry fish to specialized processing facilities in the Norwegian fish farming industry are required to use AIS equipment. This is the reason Blue Planet crafted the hypothesis that AIS data might be used to create a product to estimate the future supply of processed fish entering the market. Information about the future supply of fish is not something that is commonly known, given that many of the actors in the Norwegian fish farming industry, such as Mowi [Mowi, 2023], Lerøy Seafood Group [Lerøy Seafood Group, 2023], Grieg Seafood [Grieg Seafood ASA, 2023] and Salmar [SalMar ASA, 2023] are traded on the Oslo Stock Exchange, and such information could affect the stock valuation, amongst other things. Thus creating the hypothesized value behind a product being able to perform such analysis, providing the incentive for this thesis.

The fish carrying vessels of interest, found by analyzing the AIS data (further discussed in Chapter 5) are the following:

- *Fish Factory* vessels:
  - A vessel equipped like a fish processing facility on the sea. They are large ships equipped with all the needed gear to process and freeze fish.
  - Can stay on the sea for long periods of time and is used by companies to boost efficiency in their fishing operations.
- *Fish Carrier* vessels:
  - Used to transport fish, supplies, and crew to and from fishing vessels.
- *Fishing Vessel*:
  - These are vessels that are equipped with fishing gear such as nets, lines, or traps, and some may also have facilities for storing and processing the catch on board.
  - These vessels are usually built to withstand tough sea conditions for extended periods, allowing them to operate in various fishing grounds, ranging from coastal areas to deep seas.
- *Live Fish Farrier (wellboat)* vessels:
  - Specially designed to carry live fish, which is important to the fish farming industry. The wells are filled with seawater, keeping the fish alive while under transport.
  - These vessels are used to both move fish from farms to specialized processing facilities, and to carry young fish from hatcheries to cages in the open sea.

How the vessels are categorized in the AIS data is specified by their design, equipment, and intended use.

Chapter 3

Literature Review

In general, there is not a lot of existing literature about applications of AIS. Between 2003 and 2018 Yang et. al [Yang, 2019] identified a mere 171 articles on the subject. Interestingly, it is only from 2014 onward that the advanced applications of AIS began to emerge. Trade is a subcategory of these advanced applications and is the subcategory that serves as a substantial inspiration and source of valuable insight for this study.

Within the trade subcategory, it is only a few articles. Yang et al. highlight three articles in his subcategory. (1) Roar Os et al. look at the accuracy of the estimations of the amount of seaborne crude oil exports based on AIS data [Roar Os et al., 2017]; (2) Jia et al. proposed an algorithm for automatically generating seaborne transport pattern maps based on AIS data [Jia et al., 2017]. The algorithm automatically detects major ports and zones and aggregates 'real-time' trade flows among them, and (3) Stein W. et al. used AIS data to analyze the location distribution of VLCC oil tankers on a global level [Stein W. et al., 2018].

While these highlighted articles enriched our understanding of AIS applications in trade, it was not until the years following Yang et al.'s publication that another research article emerged, aligning more closely with the questions posed in this study. This particular paper has attempted to estimate vessel payloads - information often concealed due to the industry's opaque nature. In our investigation, this research could prove beneficial for estimating the quantity of fish being transported by a vessel.

Jia et al. suggested the following multiple linear regression model in 2019 to estimate the cargo payload [Jia et al., 2019]:

$$\pi_{r,i} = \alpha_0 + \alpha_1 DWT_i + \alpha_2 T_{r,i} + \alpha_3 LB_i + \Delta_c \wedge_{r,c} + \Omega_n \Theta_{r,n} + \epsilon_i \tag{3.1}$$

where

$\pi_{r,i} =$ the cargo payload to be estimated for vessel i, voyage r;
$DWT_i =$ the deadweight (tonnes) for vessel i;
$T_{i,r} =$ the AIS-reported draught value for vessel i, voyage r;
$\text{LB}_i =$ the product of vessel i's length overall and beam (LOA * Beam);
$\wedge_{r,c} =$ a dummy variable matrix to indicate the country of the port call, $c = 1, 2, ..., 5$;
$\Theta_{r,n} =$ a cargo type dummy matrix, $n = 1, 2, ...$

This model was applied to estimate vessel payloads of coal and iron ore in Australia, Brazil, China, India, and South Africa. This model got an R^2 of 0.948 and a mean *VIF*[1] (variance inflation factors) of 5.78. A mean VIF above 10 is usually an indication that *multicollinearity*[2] needs to be dealt with [Robert M., 2007]. It is also mentioned that other models with fewer features perform well with the worst R^2 of 0.909 while still passing the collinearity VIF test.

Significant elements to note from this model include the use of deadweight, draught, overall vessel length, and beam length. All these parameters are available in the AIS data. However, the model becomes inapplicable to our study because of the discovery that vessels transporting fish maintain a consistent $draught$. Further discussion of this can be found in Chapter 6.

[1] Variance Inflation Factor (VIF) is a measure used in statistics to quantify the severity of *multicollinearity* in regression analysis by providing an index that estimates how much the variance of the estimated regression coefficients are increased due to multicollinearity [Investopedia, 2023].

[2] Multicollinearity refers to a situation in statistical modeling where two or more features in a dataset are highly correlated, which can potentially skew or mislead the model's understanding of the importance of each feature when making predictions [Gareth James, 2013].

Chapter 4

Theory

4.1 Machine Learning

According to Mitchell [Mitchell, 1997], *machine learning* (ML) is a subfield of *artificial intelligence* (AI) that focuses on the development of algorithms and statistical models that enable computers to perform tasks without explicit programming. It leverages computational and statistical methods to learn patterns from data, which then form the basis for decision-making, prediction, and action. This field of study is distinguished by its focus on learning from data, adapting to new data, and improving performance over time. ML algorithms can be broadly categorized into (1) *Supervised learning*, where algorithms learn from labeled data to predict outcomes for unseen data. (2) *Unsupervised learning*, where algorithms discover underlying structures in data without provided labels. And (3) *Reinforced learning*, where an agent learns to make decisions by interacting with its environment and receiving feedback in the form of rewards or penalties.

4.2 Classification

According to Kuhn and Johnson [Kuhn and Johnson, 2013], *Classification* is a type of supervised ML where the goal is to assign predefined categories (labels) to new instances based on patterns learned from labeled training data. Classification is a technique used in applications for handling spam detection, image recognition, and medical diagnosis. However, a common challenge in classification tasks is balancing between *overfitting* and *underfitting* the classification model. From figure 4.1 below, we see that overfitting occurs when the model learns the training data too well, capturing noise and outliers, which reduces its ability to generalize unseen data. On the other hand, underfitting occurs when the model fails to learn significant patterns from the data, usually due to its oversimplicity or insufficient training. The ideal classification model is trained 'just right' when the model has learned the underlying patterns of the data well enough to make accurate predictions, but not so well that it has memorized the noise or specific instances of the training data. One effective strategy to deal with the challenge of underfitting and overfitting is *rolling cross-validation (RCV)*, mentioned later in Section 4.3.2.

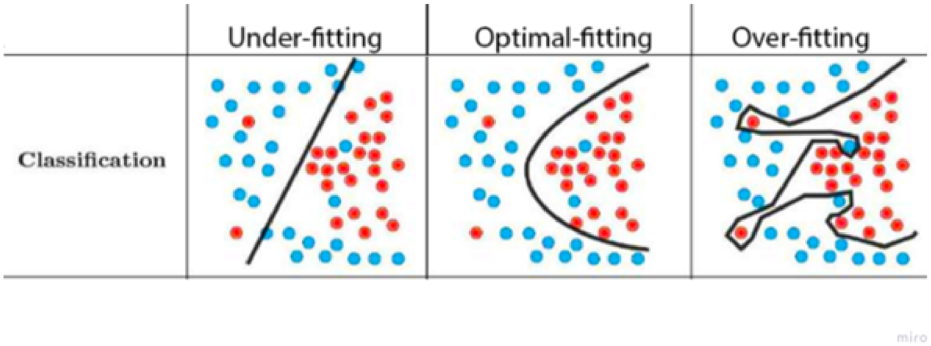


Figure 4.1: An illustration of how the quality of training a machine learning model may impact the final result of a classification model [Kuhn and Johnson, 2013].

4.3 Splitting Data

When dealing with the process of *feature engineering* (which will be extensively discussed in Section 6.5), generally the next step when preparing data to be consumed by ML algorithms is to split the dataset. For smaller sets of data, this step is particularly important. The use of proper splitting techniques is crucial to maximize the utility of limited data and ensure that the model generalizes well to unseen data.

4.3.1 Train, Test, and Validation Split

To effectively train and evaluate ML models, the data is typically split into three subsets: *training set*, *validation set*, and *test set*. The training set is used to train the ML model, allowing it to learn the underlying patterns in the data. The validation set is used during the model's training phase to tune the parameters and select the best-performing model. The test set is kept aside and used only after the model has been finalized, to evaluate the model's performance on unseen data.

For smaller datasets, the allocation of data between these subsets is vital, as each subset must be large enough to provide meaningful results. Typically, the data is divided such that the training set constitutes around 70-80% of the data, the validation set takes about 10-15%, and the test set comprises the remaining 10-15%. This proportion can, however, vary based on the size and nature of the dataset [Cawley and Talbot, 2010]. An illustration of this process can be seen in figure 4.2 below:

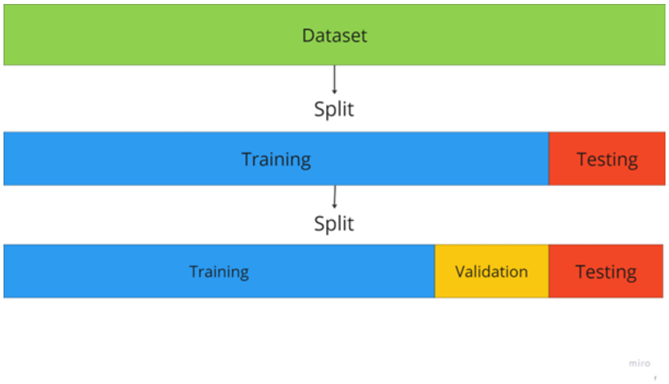


Figure 4.2: An illustration of the splitting of the dataset process, into training, validation, and testing data.

4.3.2 Rolling Cross-Validation (RCV)

Rolling cross-validation (RCV), also known as *walk-forward validation*, is a valuable strategy in *time series modeling*[1]. The value that it provides for time series modeling is its ability to preserve the temporal order of data points during model validation. As illustrated in figure 4.3 below, the model is initially trained on a 'window' of data, then it makes a prediction for the next time step. The window then 'rolls' forward in time, the true value for the predicted step is included in the training data, and a new future time step is predicted. This process continues *n* times until all time steps have been predicted. According to [Hyndman and Athanasopoulos, 2018], RCV provides a more realistic estimation of the model's performance, as it mimics the scenario of sequentially receiving data and making predictions for future events which would be the case for the data in this thesis.

[1] Time series modeling is a statistical approach that involves the analysis of sequential data points, usually collected at regular intervals, to forecast future values by leveraging the temporal dependencies between observations [Box et al., 2015].

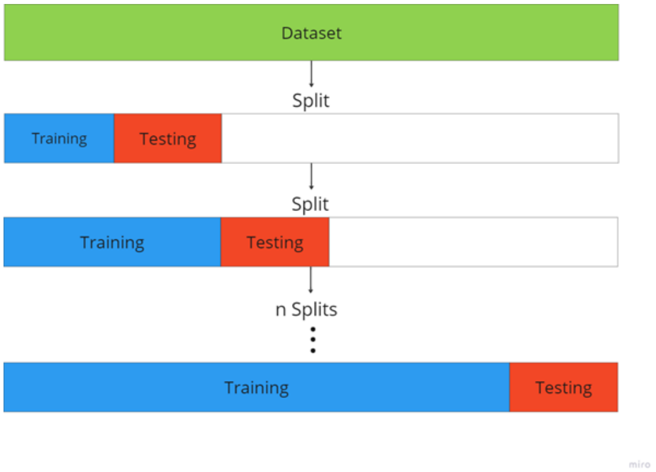


Figure 4.3: An illustration of rolling cross-validation.

Rolling cross-validation can however be quite computationally demanding since a new model is trained for every roll of the window. Despite this, for models handling time-series data, RCV is often worth the additional computational expense due to the improved performance it may provide when compared to a simple train, test, and validation split as mentioned in Section 4.3.1.

4.4 Decision Tree

Decision trees are a powerful and versatile method utilized in ML, data mining, and AI for tasks such as classification and prediction. Decision trees enable the effective segmentation of complex datasets into subsets based on specific attributes by employing a hierarchical structure consisting of nodes and branches. Decision trees can be visualized as inverted trees that allow for a clear representation of the decision-making process, such a decision tree can be seen below in figure 4.4. The root node embodies the starting point, while the terminal nodes, or leaves, signify the final classification or predicted value. In between, internal nodes are used to split the data further based on attribute values.

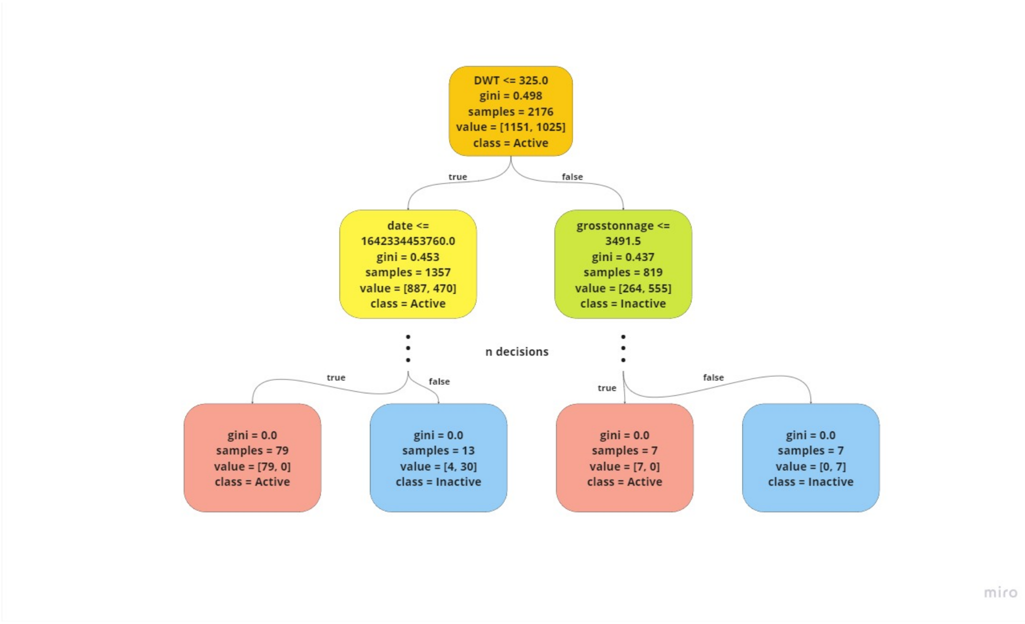


Figure 4.4: An illustration of a decision tree from our application. The tree has been minimized for presentation purposes.

Column	Description
DWT	AIS attribute measuring a vessels maximum carrying capacity.
date	Number of seconds that have passed since since January 1, 1970. 00:00:00 (UTC).
$grosstonnage$	AIS attribute for the overall internal volume of a vessel.
gini	Probability of misclassifying a random sample from a node if it was labeled according to the distributions of that node. See equation 4.1.
samples	The number of observations that reach a particular node in a Decision tree.
value	The distributions of samples across different classes in the specific node.
class	Statistical Binary classification class.

Table 4.1: Decision tree descriptions, minimized for presentation purposes.

By following a path from the root node to a leaf, one can derive valuable insights into the relationships between variables and the importance of specific attributes in determining the final outcome. The decision trees in our thesis were implemented using the $DecisionTreeClassifier()$ from the Scikit-learn library [Scikit-Learn Developers, 2023].

There are several ways to select the best attribute at each of the nodes. For this thesis, the *Gini impurity* method was utilized as the splitting criterion for the decision tree models. Gini impurity refers to the overall measure of impurity or disorder. It ranges from 0 to 1, where 0 indicates perfect purity (all elements belonging to the same class) and 1 indicates maximum impurity (all elements evenly distributed among different classes). The goal of the method is to find the splits that maximize the homogeneity of the subsets and improve the overall performance of the decision tree model [IBM, 2023a]. Gini impurity for a binary classification problem is denoted by the following formula:

$$Gini(P) = 1 - \sum_{i=1}^{k} p_i^2 \tag{4.1}$$

where

$P =$ is a set of items;
$p_i =$ is the probability of picking an item labeled with the i-th class

The key benefits of decision trees are that they are easy to interpret due to their boolean logic and visual representations. They are quite flexible and may be used for both classification and regression tasks. The algorithm also excels at discovering if two variables are highly correlated, when the split occurs, the algorithm will only choose one of the features to split on. Despite their strengths, however, decision trees are not without limitations. They are prone to overfitting, especially in the presence of noisy data or when the tree becomes excessively complex, which for the AIS data examined in this thesis might easily happen. Techniques such as pruning and ensemble methods like random forests can help mitigate these shortcomings [IBM, 2023a].

4.5 Random Forests

Random forests (RF) represent an *ensemble learning* method that merges multiple decision trees to produce more accurate and robust predictions. By employing the wisdom of the crowd, random forests effectively address the limitations of individual decision trees, such as overfitting and sensitivity to data fluctuations. See figure 4.5 below for an example of how a random forest was implemented using the $RandomForestClassifier()$ form the Scikit-learn library [Scikit-learn Developers, 2023b].

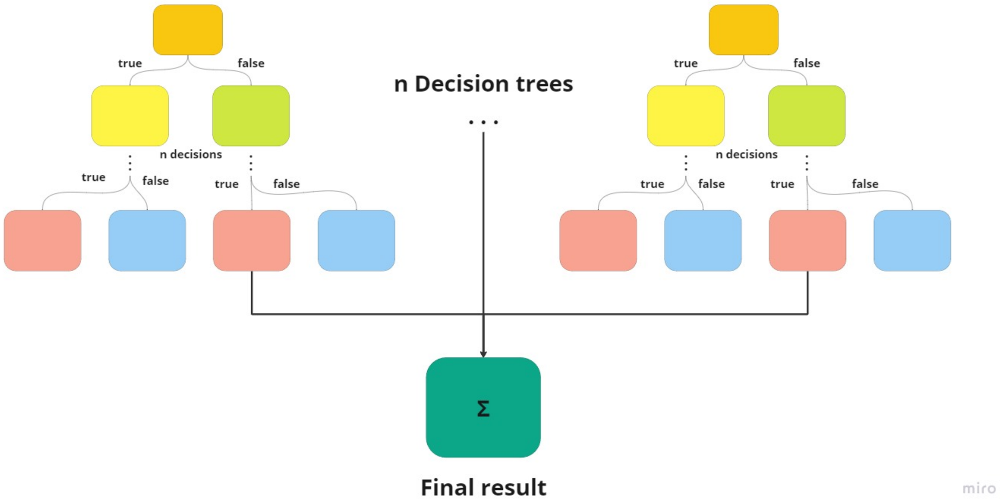


Figure 4.5: An illustration of a random forest from our application. The forest has been minimized for presentation purposes.

The random forests algorithm function by generating numerous decision trees during the training phase. Each decision tree is constructed using a random subset of the training data, obtained through *boostrapping*[2], and a random selection of features at each split. The final prediction is determined by aggregating the individual outputs of the trees, either through majority voting for classification tasks or averaging for regression tasks.

[2] Bootstrapping is a statistical resampling technique that involves creating numerous replications of the original sample, each randomly drawn with replacement, to estimate the sampling distribution of a statistic [Efron, 1979].

An essential aspect of random forests is the ability to evaluate the model's performance using out-of-bag error estimation (OOB). As each tree is trained on a bootstrapped subset of the data, the remaining samples, known as out-of-bag instances, can be used to validate the model. By calculating the prediction error for these instances, an unbiased estimate of the overall performance can be obtained. Random forests do also provide a natural measure of feature importance by examining the impact of each attribute on the model's performance. The importance of each feature is estimated by calculating the average decrease in impurity when that feature is used to split the nodes across all trees in the forest [IBM, 2023b].

4.6 Dynamic Time Warping (DTW)

Dynamic Time Warping, first introduced in 1983 by J. Kruskall and M. Liberman [Kruskall and Liberman, 1983], is an algorithm for measuring similarities between two *temporal time series*[3], which may vary in speed or timing and thus not perfectly sync up. Such time series could be those of wellboats delivering fish ready for processing to processing facilities. Suppose that we have two simple arrays containing time series data for wellboat A and B, these data points can for the sake of this example be the distance between the respective wellboat and Stjernelaks:

```
Wellboat A = [1, 1, 2, 3, 2, 0]
wellboat B = [2, 2, 1, 4, 3, 0]
```

In order to measure the similarity between the time series for wellboat `A` and `B`, we can simply use the *Euclidean distance* formula 4.2 and calculate the straight line distance between each of the points in the arrays in the n-dimensional space.

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4.2}$$

[3] Temporal time series refers to a sequence of data points collected at successive time intervals, capturing the evolution of a variable or phenomenon over time.

However, this calculation becomes more complex when we add the array containing the time series for wellboat C to the mix:

```
Wellboat C = [0, 1, 1, 2, 3, 2, 1]
```

Now we have a third array of unequal length compared to A and B. The challenge now becomes how do we determine which component should map to which. DTW responds to this challenge by using a *dynamic programming*[4] approach in order to find an optimal alignment between the two time series A and C or B and C. The goal of applying DTW is to find a match that minimizes the cumulative distance between the two time series while addressing potential non-linearities in time.

The optimization problem for Dynamic Time Warping can be expressed as follows [tslearn Contributors, 2023]:

$$\text{DTW}(i,j) = d(i,j) + \min \begin{cases} \text{DTW}(i-1,j) & \text{(insertion operation)}, \\ \text{DTW}(i,j-1) & \text{(deletion operation)}, \\ \text{DTW}(i-1,j-1) & \text{(match operation)} \end{cases} \tag{4.3}$$

where

- DTW(i,j) represents the cumulative DTW distance at position (i,j) in the DTW matrix.
- $d(i,j)$ is the local cost or distance between elements i and j of the two sequences being compared.
- DTW$(i-1,j)$ represents the cumulative DTW distance of the previous position in sequence A (insertion operation).
- DTW$(i,j-1)$ represents the cumulative DTW distance of the previous position in sequence B (deletion operation).
- DTW$(i-1,j-1)$ represents the cumulative DTW distance of the previous position in both sequences (match operation).

[4] Dynamic programming is a problem-solving method that breaks down complex problems into smaller overlapping subproblems, solving each subproblem only once and storing the results for efficient computation.

The first step to the DWT algorithm is to construct a distance matrix D, between two time series, for the following example, wellboat `A` and wellboat `C` will be used, where each cell $D[i][j]$ in this distance represents the Euclidean distance between $A[i]$ and $C[j]$.

Table 4.2: Distance matrix D

	0	1	1	2	3	2	1
1	1	0	0	1	2	1	0
1	1	0	0	1	2	1	0
2	2	1	1	0	1	0	1
3	3	2	2	1	0	1	2
2	2	1	1	0	1	0	1
0	0	1	1	2	3	2	1

The second step in the DTW algorithm is to create a cumulative matrix, M, using dynamic programming. Each cell $M[i][j]$ will be the sum of $D[i][j]$ and the minimum of $M[i-1][j-1]$, $M[i-1][j]$, and $M[i][j-1]$. For the first row and the first column, the current distance to the previous cumulative distance is added since there's only one path.

Table 4.3: Cumulative distance matrix M

	0	1	1	2	3	2	1
1	1	1	1	2	4	5	5
1	2	1	1	2	4	5	5
2	4	2	2	2	3	3	4
3	7	4	4	3	3	4	6
2	9	5	5	3	4	3	4
0	9	6	6	5	7	5	4

Finally, the DTW distance between the two time series is the value in the last cell in the cumulative distance matrix $M[A.length][C.length]$, which for this case equals 4.

The third step is to backtrack from $M[A.length][C.length]$ to $M[0][0]$ to find the optimal alignment path. The rule for backtracking is to move from $M[i][j]$ to the cell that gives the minimum cumulative distance, which could be either one of $M[i-1][j-1]$, $M[i-1][j]$, and $M[i][j-1]$ as mentioned above. The optimal warping path for this example is illustrated in figure 4.6 below:

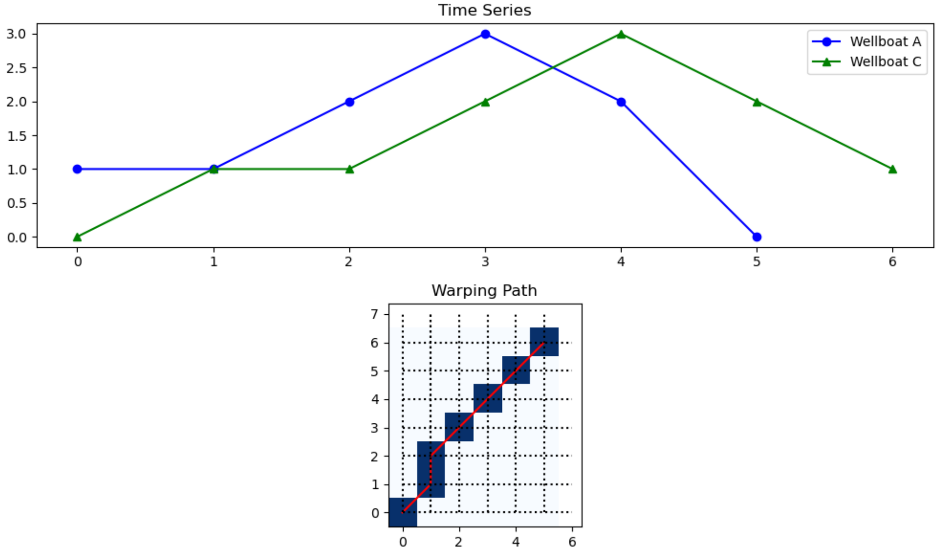


Figure 4.6: Dynamic Time Warping algorithm performed for the time series of wellboats `A` and `C`.

This warping path suggests that the first two elements in the time series for wellboat `A` $(1, 1)$ align with the three first elements of the time series for wellboat `C` $(0, 1, 1)$, and so forth. This visualization of the warping path shows how Dynamic Time Warping, in this example has accounted for the shifted and stretched pattern in the time series data for wellboats `A` and `C`.

Chapter 5

Data

This study employs two types of data. The first type of data used in this thesis is publicly available AIS data from *Kystdatahuset*[1], consisting of historical data from 2013 - today's date, which are being continuously updated. AIS data serves as *input data*[2] for the analysis. The second data type used in this thesis is the fish processing data from Stjernelaks, which includes production data from their operations since 2016. The processing data serves as *label data* for the analysis. This chapter details the extensive process of data collection and preprocessing that had to be done to make the data consumable for the ML models. This chapter also provides a description of the data content used throughout the thesis.

Various online public sources offer AIS data, including Barentswatch, Marine Traffic, and Kystdatahuset. After comparing these sources, Kystdatahuset was selected due to its API with advanced filtering capabilities and diverse data types, such as *vessel information*, *AIS tracks*, and *AIS points*. Furthermore, Kystdatahuset was the most straightforward to set up, offered free data, and

[1] Kystdatahuset is a significant initiative by the Norwegian Coastal Administration (Kystverket) aimed at providing easy and efficient access to maritime traffic data for internal and external users. The platform offers several means for data retrieval, including a dashboard for interactive analysis, API for automated data transfers, and a data-sharing portal for larger dataset downloads, covering areas such as traffic statistics, vessel details, and navigational patterns [Kystverket, 2023].

[2] Input data refers to the information or variables provided as input to a system or algorithm for processing or analysis.

provided customer service. Throughout the thesis, several API endpoints available via Kystdatahuset's API [Kystdatahuset, 2023] were examined. Ultimately the following two endpoints were utilized to create the final two datasets for the later analysis (Chapter 6):

1. **Kystdatahuset API 1: POST /api/tracks/within-area**

   Fetches all *tracks* within a given geographic area and time range. Returns information about vessels and their tracks represented as *GeoJSON*[3] objects.

2. **Kystdatahuset API 2: POST /api/ais/positions/for-mmsis-time**

   Fetches *positions* of AIS vessels for a given set of MMSI (abbreviation in table 2.1), numbers, and time range. Returns vessel information, latitude, longitude, and time.

AIS Positions: Refers to the individual data points transmitted by a vessel's AIS data. These data points, or *positions*, include specific information like the vessel's current location (latitude and longitude), speed, direction (course), and other details such as the vessel's identity, type, and status. Each AIS position is essentially a snapshot of the vessel's state at a particular point in time.

```
{
    "positions": [
      {
        "mmsi": 257999000,
        "datetime utc": "2021-01-01T00:00:00",
        "longitude": 5.85643,
        "latitude": 59.22842,
      }
      ...
    ]
}
```

Listing 5.1: AIS position example.

[3] GeoJSON is a format for encoding a variety of geographic data structures [GeoJSON, 2023].

AIS Tracks: A series of AIS positions linked together over time to visualize the path or route that a vessel has taken. An AIS track gives you a historical view of the vessel's movements. It's like connecting the dots between each AIS position to create a continuous line that represents the vessel's journey.

```
{
  "mmsi": 257999000,
  "starttime": "2021-01-01T00:00:00",
  "endtime": "2021-01-01T23:59:59",
  "geometry": {'type': 'LineString', 'coordinates': [[5.85643,
     59.22842], [5.85577, 59.2285], [5.85627, 59.22896]]}
}
```

Listing 5.2: An AIS track example.

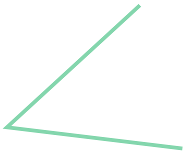


Figure 5.1: Visualization of the AIS track example in listing 5.2.

The process extensively explained in this chapter is illustrated below in figure 5.2. The figure shows how Data type 1: *Fish processing data from Stjernelaks* (Section 5.1) and Data type 2: *AIS data* (Section 5.2) is passed through the *Data pipeline* (figure 5.3 below) to create the final two datasets: *Labeled Days* and *Labeled Time Series*. See the following tables 5.1 and 5.2 for a representation of a randomly picked sample from these final two datasets.

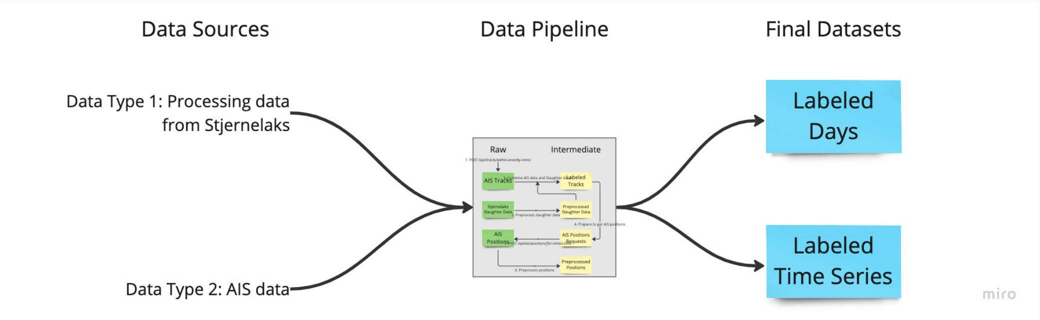

Data Sources
Data Pipeline
Final Datasets
Data Type 1: Processing data from Stjernelaks
Data Type 2: AIS data
Raw
Intermediate
1. POST /api/tracks/within-area/by-mmsi
AIS Tracks
3. Combine AIS data and Slaughter data
Labeled Tracks
Stjernelaks Slaughter Data
2. Preprocess slaughter data
Preprocessed Slaughter Data
4. Prapare to get AIS positions
AIS Positions
5. POST /api/ais/positions/for-mmsis-time
AIS Positions Requests
6. Preprocess positions
Preprocessed Positions
Labeled Days
Labeled Time Series
miro

Figure 5.2: From Data Sources, through Data Pipeline, to the Final Datasets.

Column	Description	Example
day	Represents a specific day of the month.	29 (for the 29th day of the month)
weekday	Represents a specific day of the week, with values ranging from 0 to 6.	1 (for Tuesday)
week	Represents a specific week number within a year.	13 (for the 13th week of the year)
visit*	Indicates if a relevant vessel has sailed within a 500m radius of Stjernelaks a specific day	True
Active	(Label) A boolean indicator of Stjernelaks activity status on a specific day.	True

Table 5.1: A randomly picked sample from the *Labeled Days* dataset with description. Features marked with (*) are AIS features. *Active* is the label the models will try to predict.

The *Labeled Days* dataset will be used in the analysis to answer the first subpart of the research question, **Sub-RQ 1**: Can AIS data be used to predict if Stjernelaks is processing fish on any given day, regardless of the source being a waiting cage or direct vessel delivery?

Column	Description	Example
day	Represents a specific day of the month.	11 (for the 11th day of the month)
weekday	Represents a specific day of the week, with values ranging from 0 to 6.	0 (for Monday)
week	Represents a specific week number within a year.	2 (for the 2nd week of the year)
direct_distance_sum_norm*	Represents the sum of all DTW distances between a specific time series and all other time series labeled as True. This sum is then normalized (0 to 1).	0.022290
closest_distance_to_stjernelaks*	Represents the Euclidean distance in kilometers between Stjernelaks and the specific time series' closest position to Stjernelaks.	0.0211
$Direct$	(Label) A boolean indicator of whether Stjernelaks received fish directly from a relevant vessel a specific day.	True

Table 5.2: A randomly picked sample from the *Labeled Time Series* dataset with description. Features marked with (*) are AIS features. $Direct$ is the label the models will try to predict.

The *Labeled Time Series* dataset will be used in the analysis to answer the second subpart of the research question, **Sub-RQ 2**: Can AIS data be specifically used to predict if Stjernelaks is processing fish that has been directly delivered by a vessel on any given day?

The following data pipeline was developed to ensure consistent data collection quality, illustrated in figure 5.3. Throughout this chapter, this pipeline figure will be referenced multiple times. The pipeline was developed and implemented with scalability in mind enabling it to be utilized for future add-ons with different time periods or locations.

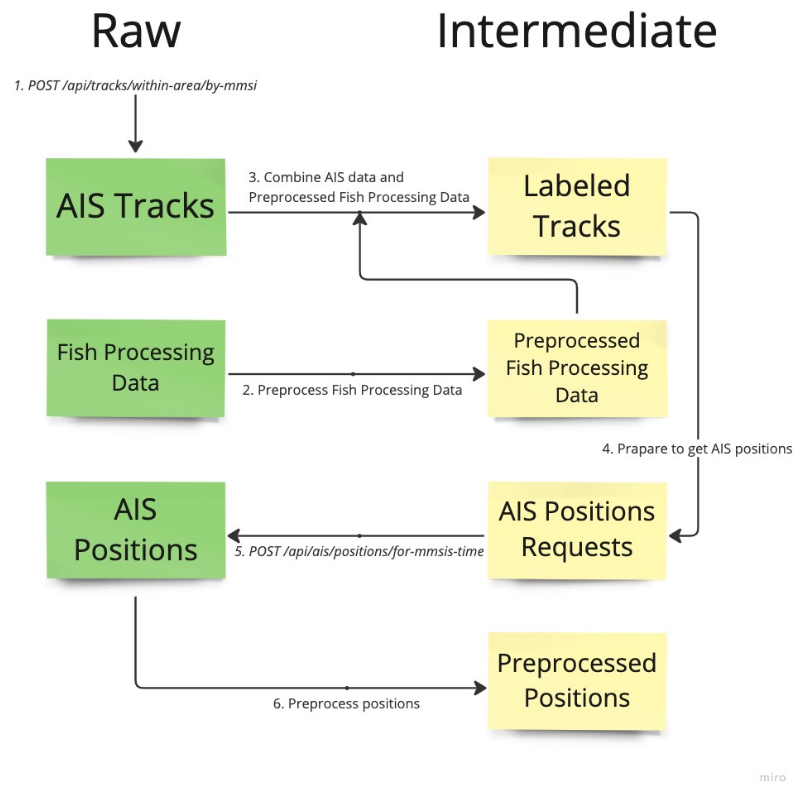

Raw
Intermediate
1. POST /api/tracks/within-area/by-mmsi
AIS Tracks
3. Combine AIS data and Preprocessed Fish Processing Data
Labeled Tracks
Fish Processing Data
2. Preprocess Fish Processing Data
Preprocessed Fish Processing Data
4. Prapare to get AIS positions
AIS Positions
5. POST /api/ais/positions/for-mmsis-time
AIS Positions Requests
6. Preprocess positions
Preprocessed Positions
miro

Figure 5.3: In this data pipeline, green represents *raw* datasets and yellow denotes preprocessed raw datasets, also referred to as *intermediate* data. The arrows symbolize transitions, showing how datasets are used to create new datasets, represented by a box at each arrow's end. The prefixed numbers indicate the order of transitions, starting with 1 and ending with 6.

5.1 Data Type 1: Stjernelaks Fish Processing Data

Transition number 2 in figure 5.3 represents the preprocessing of the *raw*[4] data received from Stjernelaks. The raw data comprises Excel sheets for each year since 2016, with varying structure and quality of information. One example of this varying structure is that starting from 2022 the `vessel` column is used to describe which vessel delivered fish. In the years before 2022, this column was reserved for comments only.

The raw Stjernelaks data is manually inserted and therefore prone to have many human errors [Barchard and Pace, 2011]. It is known as best practice among researchers to make use of corrective strategies to discern outliers in their datasets, including the utilization of graphical representations and diagnostic statistics, as noted by [Mavridis and Moustaki, 2008], and [Tukey et al., 1977]. In the analysis of the raw Stjernelaks data, such techniques were deployed. The iterative process of data cleansing was carried out using graphical representations and diagnostic statistics until the data reached an acceptable standard. This practice wasn't exclusive to the raw processing data; it was universally applied across the pipeline, encompassing all *raw*, *intermediate*, and later *processed* final datasets.

[4] The data referenced as raw in this thesis is the untouched Excel sheets received from Stjernelaks. The raw data referenced in relation to AIS data is the untouched responses from API endpoints.

From the raw Stjernelaks data, the data pipeline extracts two labels from each row:

1. $Active$ **Label (boolean): Used as the Label in the final dataset *Labeled Days*.**
   True when the amount of fish processed > 0. False otherwise.
2. $Direct$ **Label (boolean): Used as Label in the final dataset *Labeled Time Series*.**

   True when a relevant vessel is registered. False otherwise.

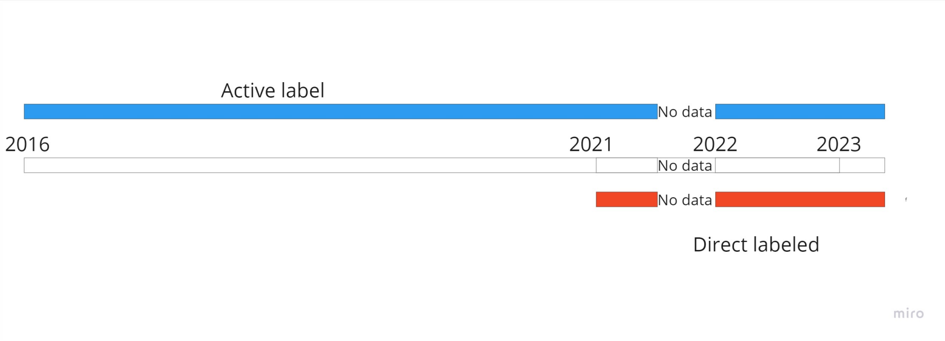


Figure 5.4: Time periods where we can extract $Active$ and $Direct$ labeled data.

Unfortunately, prior to 2021, it is only possible to obtain the $Active$ label. Additionally, Stjernelaks fish processing data from April to December 2021 is missing for unknown reasons. As a result, the only available period with both $Direct$ and $Active$ labeled data is from the beginning of 2021, all of 2022, and the start of 2023, resulting in approximately 1.5 years or 547 days. The amount of $Active$ labeled days is approximately 5.5 years or 2 007 days.

The $Active$ label is based on the amount of processed fish. Figure 5.5 below visualizes the amount of processed fish summed up for each year.

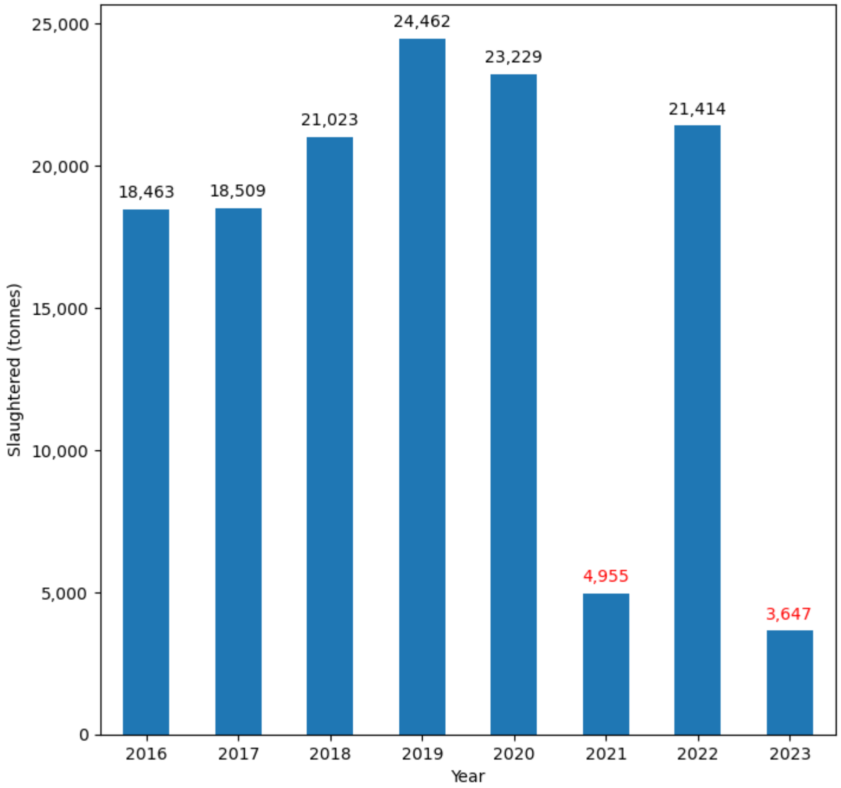


Figure 5.5: Fish processed in tonnes, represented annually. Outliers are marked with red.

The total amount of fish processed hovers around 18,000 to 25,000 tonnes annually when the outliers from 2021 and 2023 are disregarded. This makes sense when compared to Stjernelaks' history of total annual processing restrictions. In 2013 their restrictive capacity got increased from 15,000 to 25,000 tonnes annually [Meling, 2021]. Later in 2022, it got increased again from 25,000 to 35,000 tonnes annually [Statsforvalteren, 2022]. The outlier year 2021 has a low total processed amount because of the earlier mentioned missing Stjernelaks fish processing data from this year. The other outlier year, 2023, is low since the information cut-off from Stjernelaks was at the beginning of March 2023. This is further clarified by figure 5.6 below, describing the cumulative amount of processed fish annually.

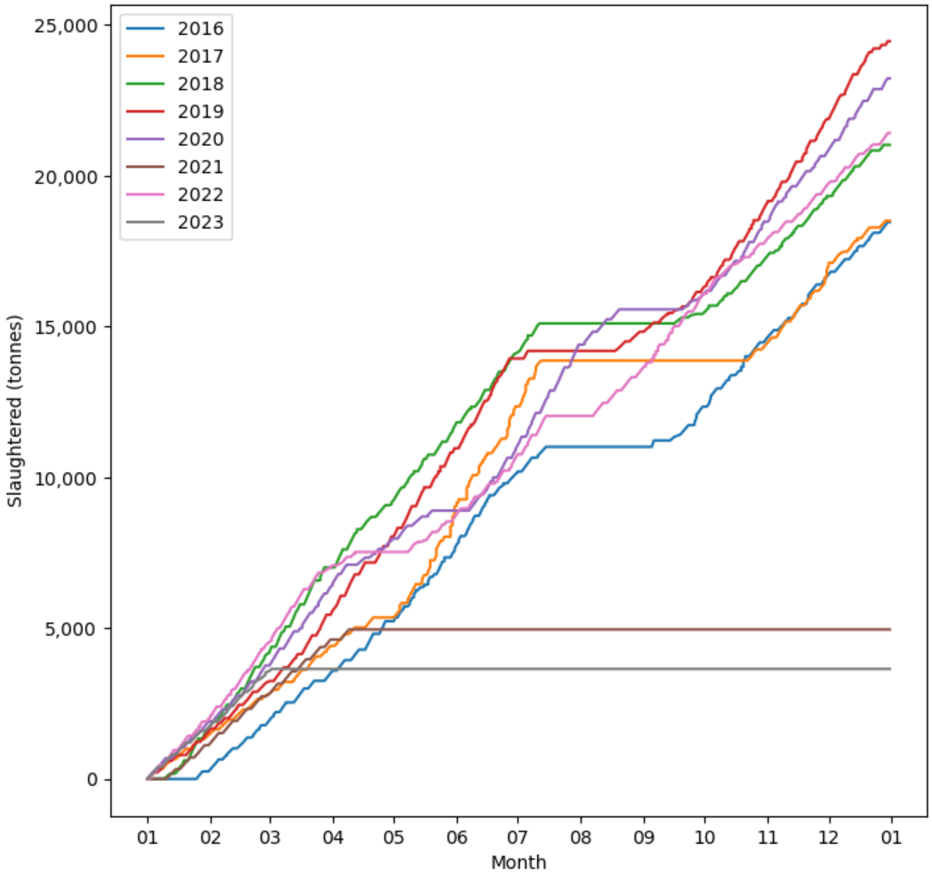


Figure 5.6: Cumulative development of the amount of fish processed each year in tonnes.

In figure 5.6 the growth is determined by the amount of fish processed in tonnes. If no fish processing was recorded the line is flat. From the figure, it can be observed that the outliers: the brown 2021 line is flat from around 04 (April), and the grey 2023 line is flat from around 03 (March). For the other years, they all follow a common trend. They increase until around 06 (June) when they start to flatten, until the start of 09 (September) when they gradually begin to increase again.

If the line is flat and no fish is processed, the processing facility state is set as $Inactive$ for that day by the thesis' solution. Consequently, if the line is growing, meaning fish is being processed, the processing facility state is set as $Active$ that day. Figure 5.7 below takes a closer look at which months have the highest amount of $Inactive$ days:

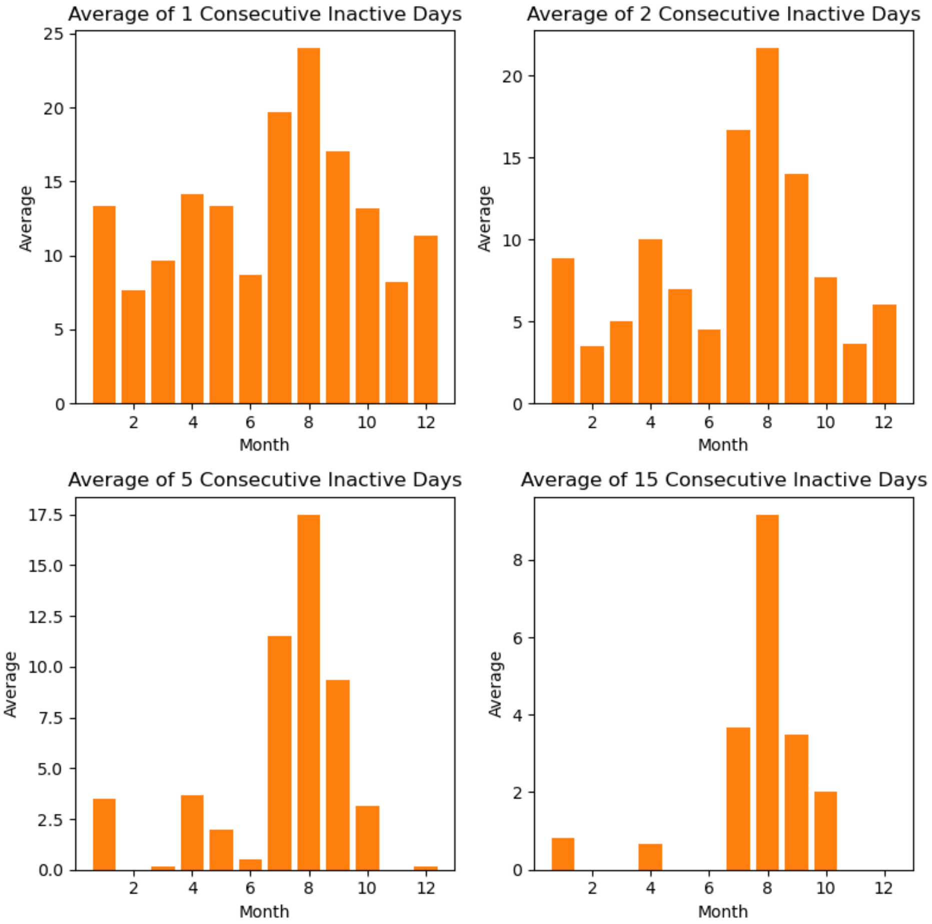


Figure 5.7: The bars represent the average number of respectively 1, 2, 5, and 15 consecutive days with no fish processing across all months since 2016. Outlier years 2021 and 2023 are disregarded in this representation.

Figure 5.7 visualize where the cumulative lines in figure 5.6 are flat. The sub barplots displaying the average of 1 and 2 consecutive *Inactive* days indicate that it is normal to have a few sporadic *Inactive* days throughout the year. For the sub barplots when the consecutive *Inactive* days are increased to 5 and 15, it becomes evident that a few months separate themselves from the rest. These are the months from 07 (July) to 10 (October), which have, on average longer periods of consecutive *Inactive* days. Reasons for this trend can, for example, be caused by vacations, market strategy, or less fish supply.

From figure 5.7, we can see that different months have a skewed distribution of consecutive *Inactive* days. Can this also be the case for weekdays? Figure 5.8 below highlights this question.

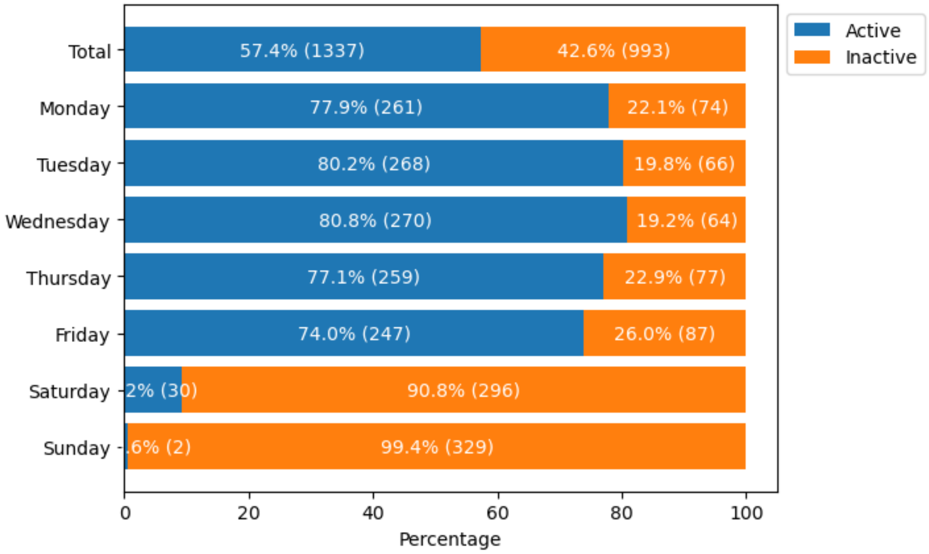


Figure 5.8: Compares the percentage distribution of *Active* and *Inactive* days for weekdays, including the total distribution in the top bar. Outlier years 2021 and 2023 are disregarded in this presentation.

As seen in figure 5.8, there is nearly zero fish processing on weekends. This observation will later on in this thesis have a significant influence on selecting the baseline and the performance of the predictive models. Additionally, the 'total' percentage distribution bar indicates that the classes can be considered balanced.

So far in this chapter, the number of *Active* and *Inactive* days has been explored, but it is also crucial to delve further into what constitutes an *Active* day. For a more comprehensive understanding, figure 5.9 below illustrates the daily quantity of processed fish. This representation will provide a tangible measurement for an *Active* day, strengthening the analysis presented in this thesis.

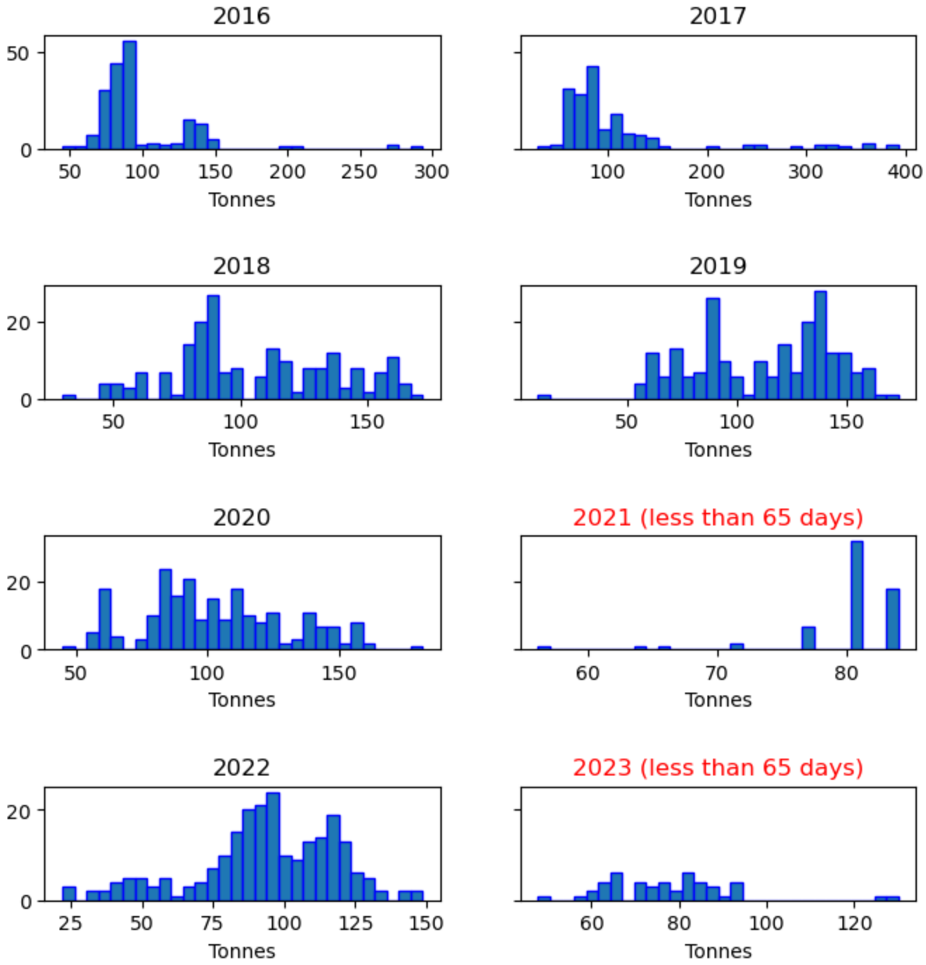


Figure 5.9: Histogram tonnes of fish processed per day.

The processing capacity at Stjernelaks is designated at 90 tonnes per shift (Lars Martin Hetland, Grieg Seafood Stjernelaks 2023). With that in mind, 2016 and 2017 in the figure display an unusually high volume of fish processed. It's important to note that these years, as shown in figure 5.5, actually processed fewer fish compared to subsequent years. This lower total volume can be attributed to a reduced number of $Active$ days. Which in turn leads to a lower total number of tonnes processed, despite some days showcasing extraordinary quantities of fish processed. This pattern is depicted in the figure 5.10 below.

All years tend to peak around Stjernelaks' capacity at 90 tonnes. However, the histograms in figure 5.9 are not normally distributed around this amount. Through conversations with Stjernelaks, we know that an additional shift might be added occasionally that may explain this. An additional shift will consequently increase the amount of fish processed that day. Other factors that do not necessarily have an annual seasonality are, for example, fish supply, vessel capacity, and weather.

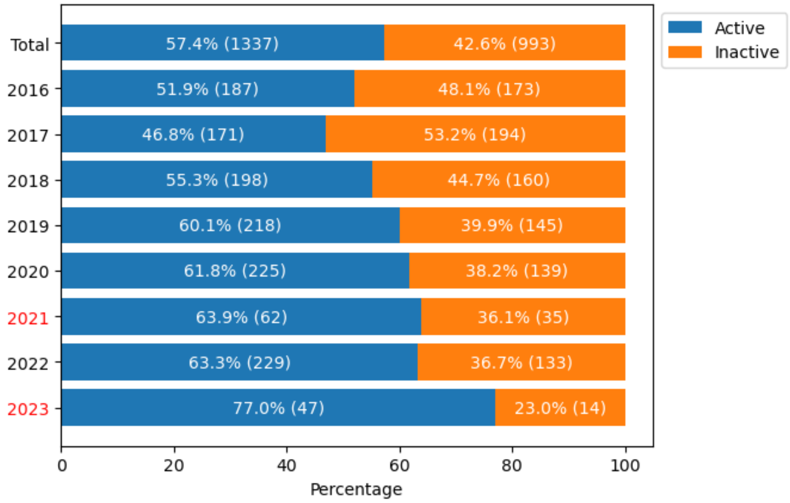


Figure 5.10: Compares the percentage distribution of $Active$ and $Inactive$ days for all years, including the total distribution in the top bar. Outlier years 2021 and 2023 are included in this presentation

In figure 5.10 we can see an increasing trend in the ratio of $Active$ days. However, this does not directly relate to the yearly amount of processed fish earlier visualized in figure 5.5.

Similar patterns and observations were made when investigating the $Direct$ label. These are not presented due to the limitations mentioned later in Chapter 8.

5.2 Data Type 2: AIS Data

The next data type used throughout this thesis is the AIS data. The gathering of AIS data is represented in the previously mentioned data pipeline figure 5.3 as transition 1. Transition 1 involves collecting the AIS data from Kystdatahuset's API, as mentioned the endpoints used are:

1. **Kystdatahuset API 1: POST /api/tracks/within-area**

   Fetches all tracks within a given geographic area and time range. Returns information about vessels and their tracks represented as GeoJSON[5] objects.

2. **Kystdatahuset API 2: POST /api/ais/positions/for-mmsis-time**

   Fetches positions of AIS vessels for a given set of MMSI (Abbreviation in table 2.1), numbers and time range. Returns vessel information, latitude, longitude, and time.

The ideal way to gather the AIS data would be to gather all AIS positions for all vessels since 2016 within a relevant area surrounding Grieg Seafood Stjernelaks' geographical position. Unfortunately, the Kystdatahuset API endpoint (2) lacks an area filter like this. Additionally, as mentioned in Chapter 2, AIS positions are reported roughly every 7 seconds, yielding nearly 190 million samples from a single vessel in the period of interest, with a significant portion of the data being unnecessary for the analysis.

[5] Recall that GeoJSON is a format for encoding a variety of geographic data structures [GeoJSON, 2023].

Kystdatahuset's API endpoint (1) includes an area filter technique called *geofencing*. This geofencing has been used to limit redundant data and fetch every track registered within a 500-meter radius of Stjernelaks since 2016. The geofencing from (1) is used to fetch all relevant tracks, which in turn is implemented in (2) to ensure that every vessel that has delivered fish to Stjernelaks is present in a new raw dataset, called *AIS Positions* (transition 1 to 5 in figure 5.3). The AIS Positions dataset includes vessels that deliver fish and potentially irrelevant positions from vessels that do not deliver fish. This issue is later addressed in Chapter 6. The length of the radius used in the geofencing was determined through a trial-and-error approach. Factual data of vessels known to deliver fish to Stjernelaks, as for example, the fish carrier vessel *Ronja Polaris*[6], were used to understand ship movement close to the processing facility. Too small of a radius risked not including vessels that delivered fish to Stjernelaks. Too big of a radius risked including vessels that do not deliver fish to Stjernelaks, which would be difficult to filter out later.

The Kystdatahuset's API endpoint (1) only supports receiving points in a polygon. This was solved by calculating 10 points in a circle using the Haversine formula with the processing facility's coordinates as the center and 500 meters as the radius.

The Haversine formula is used in navigation, providing great-circle distances between two points on a sphere from their longitudes and latitudes. It's important in navigation because it considers the curvature of the Earth. The Haversine formula is denoted by [Azdy and Darnis, 2019]:

[6] Recall that Ronja Polaris is a Fish Carrier vessel sailing under the flag of Norway, built in 2013, with a length of 75.8 meters and a breadth of 16 meters, and it provides real-time data about its location, status, and voyage details through the Automatic Identification System. [MarineTraffic, 2023]

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right), \tag{5.1}$$

$$c = 2 \cdot \mathsf{atan2}\left(\sqrt{a}, \sqrt{1-a}\right), \tag{5.2}$$

$$d = R \cdot c, \tag{5.3}$$

where:

- φ_1, λ_1 is the latitude and longitude of the first point in radians,
- φ_2, λ_2 is the latitude and longitude of the second point in radians,
- $\Delta\varphi = \varphi_2 - \varphi_1$,
- $\Delta\lambda = \lambda_2 - \lambda_1$,
- a is the square of half the chord length between the points,
- c is the angular distance in radians,
- R is the radius of the Earth (mean radius = 6,371 km),
- d will be the distance between the two points (along the sphere's surface).

The formula essentially works by computing the spherical distance between two points, given their longitudes and latitudes. It's particularly useful in calculating the shortest distance between points on the Earth's surface, as represented by a spherical surface. Figure 5.11 below illustrates the geofencing placed around Stjernelaks.

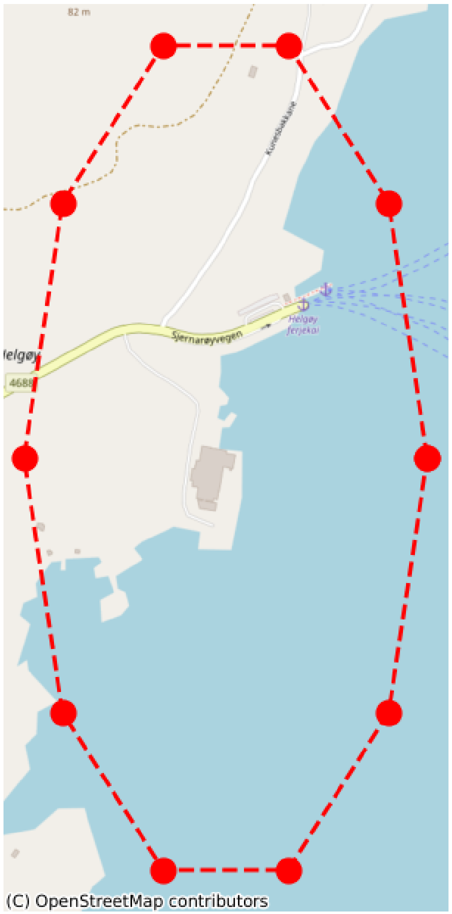


Figure 5.11: Geofencing around Stjernelaks illustrated.

By visualizing the 10 points, the result is a stretched-out circle. However, this is expected due to the distortion occurring when representing the globe as a two-dimensional map. In the figure 5.12 below we can see traces of this stretched-out circle by seeing that the outside borders are taking an oval shape.

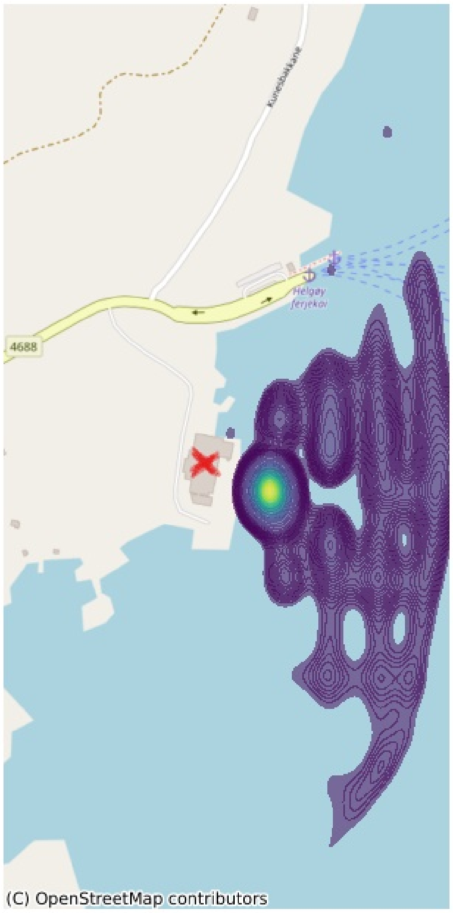


Figure 5.12: Kernel density of a representative sample of tracks within the geofencing of Stjernelaks. 77.48% of 2909 total samples of relevant vessels. The location of Stjernelaks processing facility is marked with red X.

In order to display the kernel density in figure 5.12, a representative sample was selected by using Yamane's formula [Yamane, 1967], with a marginal error of 0.01. Yamane's formula is denoted by:

$$n = \frac{N}{1 + N(e)^2} \tag{5.4}$$

where

- n is the number of samples,
- N is the size of the population,
- e is the margin of error,

The result from API endpoint (1) returns over 20 different AIS columns. Here are the columns relevant to this thesis:

Column	Description
MMSI	Ship's ID.
starttime	Start time of the track.
endtime	End time of the track.
shiptypenor	Type of the ship.
shiptypenor2	Detailed type of the ship. E.g., Live Fish Carrier.
geometry	Geometric shape or feature described using GeoJSON format.
draugth	See draught in table 2.2.
dwt	See dwt in table 2.1.

Table 5.3: Overview of the relevant columns generated from Kystdatahusets API (1).

The kernel density illustrated in Figure 5.12 comprises information derived from the geometry of vessels deemed relevant. These relevant vessels are identified through filtering processes using the $shiptypenor$ and $shiptypenor2$ columns from table 5.3. The $shiptypenor$ column must be designated as *Fish*, and the $shiptypenor2$ column must be classified as one of the following relevant vessels: *Live Fish Carrier (Well Boat)*, *Fish Carrier*, *Fishing Vessel*, or *Fish Factory Ship*. This categorization was constructed utilizing Stjernelaks fish processing data post 2021, when Stjernelaks started registering the specific names of vessels that delivered fish for processing. The categories that formed the list were determined by investigating the $shiptypenor2$ classifications of these vessels.

Figure 5.12 shows two purple paths into the yellow area from the east. This resonates with our understanding of how the processing facility operates. Either it delivers fish directly by connecting to a tube on the north side or it delivers fish to the waiting cages that are south of the tube. The empty spot between the north and south path can be explained by a buoy preventing the ships from sailing there.

With assistance from Blue Planet and Stjernelaks, the correct time periods to fetch positions were determined to be; one day before and one day after a vessel sailed within the 500-meter radius of Stjernelaks. In other production areas of Norway, where transportation from fish farms to processing facilities is extended, these intervals should be longer. However, Stjernelaks primarily receives fish from nearby farms. Limiting the time period positions are fetched significantly reduces the amount of data. As a result of this, approximately 6 million positions are fetched (instead of 4.5 billion). The following figure 5.13 describes which periods positions were fetched from.

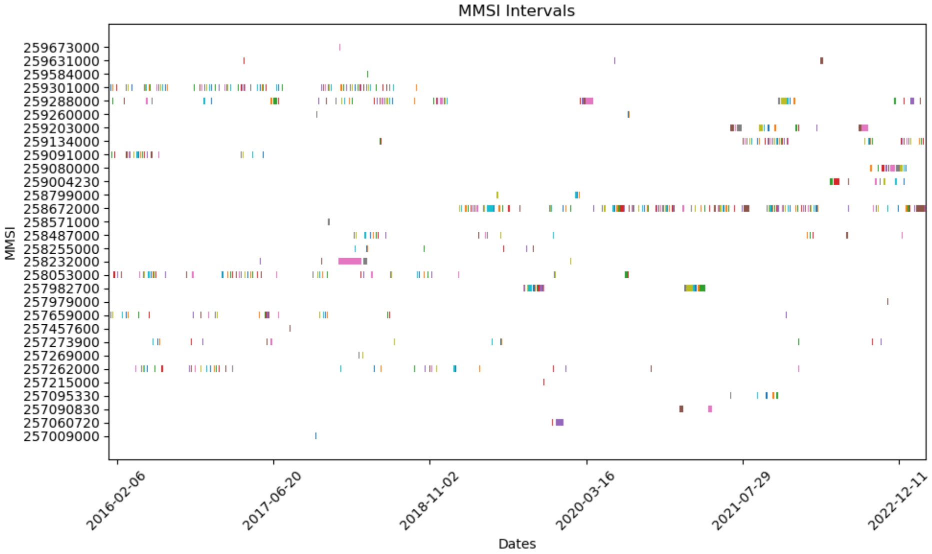


Figure 5.13: Visualization of AIS positions requests.

Upon examining the fetched AIS positions (transition 5 in figure 5.3) using Kystdatahuset's API (2), we discovered that columns displaying data based on previous points, such as `seconds to previous point`, contained numerous errors. For example, there were rows with identical timestamps for the same vessel. In these cases of duplicate occurrences, only the first row was selected, and duplicates were removed, even if they had slightly different values in other columns. In total, 43 313 duplicate rows were removed from the total 6 million positions (approx `0.72%)`. These issues were addressed in transition 6 from the data pipeline, figure 5.3.

A visualization of the positions fetched within the time period one day before, to one day after a relevant vessel sailed within the 500m radius of Stjernelaks, can be seen in figure 5.14. From kernel density applied to the figure, we see that the fetched positions make sense since the highest density is situated at Stjernelaks, Helgøy.

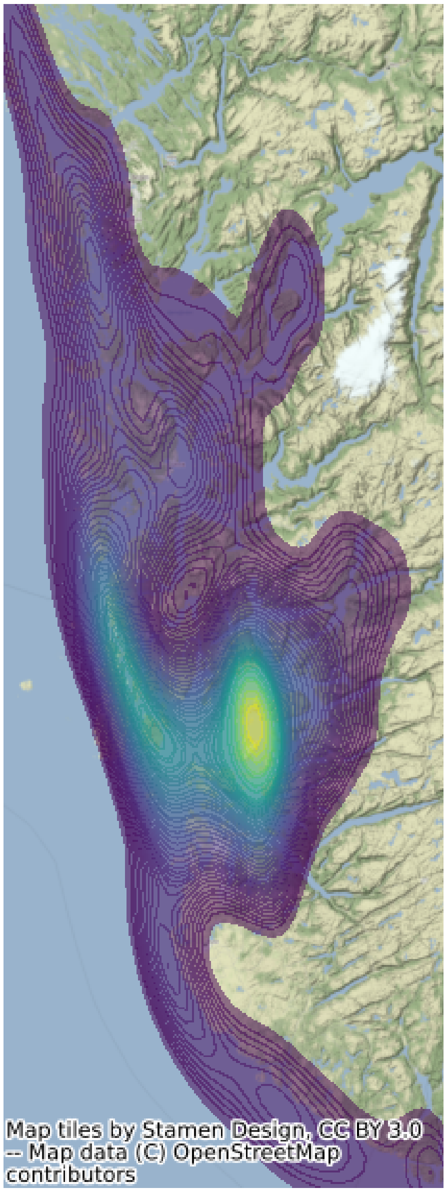

Map tiles by Stamen Design, CC BY 3.0
-- Map data (C) OpenStreetMap
contributors

Figure 5.14: Kernel density of a representative sample of positions. 0.17% of total positions using Yamane's formula.

5.3 Processed Data

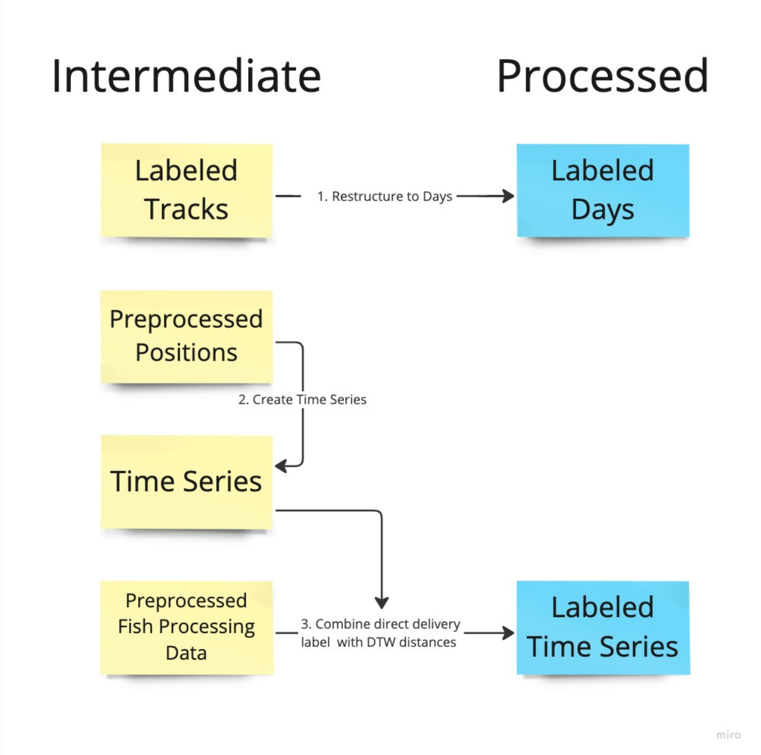


Figure 5.15: Yellow denotes preprocessed raw datasets, also referred to as *intermediate* data. Blue represents the *processed* datasets. These datasets are the final product of the pipeline and are used in the analysis.

This data pipeline in figure 5.15 is a continuation of the data pipeline 5.3. It explains how some of the intermediate datasets, 'Labeled Tracks,' 'Processed Positions,' and 'Preprocessed Fish Processing Data' from the previous data pipeline, are used to create two processed datasets. These are the datasets *Labeled Days* and *Labeled Time Series*. Recall that the *Labeled Days* dataset uses the $Active$ label, and the *Labeled Time Series* uses the $Direct$ label. The pipeline's transitions utilize advanced techniques that will be discussed further in Chapter 6. The first transition, marked as number 1 in figure 5.15, includes feature engineering (see Section 6.5) to create the *Labeled Days* dataset, which is the first of the two processed datasets used in the final analysis. All the following chapters will exclusively use one of these two processed datasets marked with blue in the data pipeline figure 5.15.

The creation of the second processed dataset starts at transition number 2 in figure 5.15, which uses the AIS positions to create time series. A single time series consist of multiple positions from a vessel for single day. In other words, a time series represents a vessel's sailing path for one day. A problem with this approach is that some vessel voyages lasting over midnight will be split into two different time series. However, we know that the delivery of fish usually does not happen around midnight. Therefore, the period around fish delivery is likely captured in the time series, which is the period of interest.

The process of labeling time series, transition 3, introduces a different set of challenges compared to the labeling techniques used for labeling days. Therefore, a crucial step in our analysis involves identifying the appropriate label for each individual time series. The $Direct$ label would indicate whether the specific time series under consideration pertains to a vessel delivering fish. We also labeled the time series of vessels the day before and after they sailed within the 500m radius of Stjernelaks. This means that some of the time series might not have any registered positions within the 500m radius. These time series are not filtered out, but rather labeled as $Direct =$ False. This was done to keep the dataset balanced, ensuring the models generalize well. The time series that can be labeled as $Direct =$ True, are the ones that coincide with the vessels registered in the Stjernelaks fish processing data on the registered day. Unfortunately, the dataset only started recording vessels that delivered fish for direct processing in 2021. This limitation is a significant consideration that must be acknowledged when investigating the outcomes of this thesis' analysis.

The intermediate *Time Series* dataset from figure 5.15 had roughly 2000 time series. However, only 422 of these could be labeled with the $Direct$ label. In total, 213 time series were labeled as True, and the remaining 209 were labeled as False. Regrettably, 41 of the $Direct =$ True labels were not matched to any time series. The source of this problem was that Kystdatahuset's API endpoint (1) seemed to neglect some tracks. This problem could have been addressed by relying less on the faulty endpoint and fetch more positions. Unfortunately, the problem was discovered late in the semester and was not addressed due to time constraints.

In transition 3 in figure 5.15, the DTW distances between time series were also calculated and merged to create the final *Labeled Time Series* dataset. This

DTW calculation is discussed further in Section 6.9.

5.4 Data Chapter Summary

This chapter is essential to the analysis and the results described later in this thesis. For studies that rely on large amounts of data, it is essential that the underlying data has gone through a cleaning process and is reliable. Clean data increases overall productivity and facilitates informed decision-making by providing high-quality information [Tableau, 2023].

Key points to keep in mind when moving on from this chapter are the following:

- **Data type 1: Stjernelaks data**
  - From January to June we see that the production is steadily increasing. From June to September, we see that the production stagnates towards a flat period. From September to December, we see that the production is steadily increasing again.
  - From the data, we see that 75-80% for the weekdays, the processing facility is $Active$. We see that they have close to 0% $Active$ days for the weekend.
  - We also see that the amount of fish processed on $Active$ days varies a lot, thus making it complicated to predict precise amounts of fish processed.
  - Amount of tonnes of fish processed each day highly fluctuates.
  - A possible weakness in the final Stjernelaks dataset is that employees manually enter all the raw Stjernelaks data. Additionally, in the cleaning, some intuition on our part had to be made in interpreting data.
- **Data type 2: AIS data**
  - Time intervals with positions are fetched from the day before to the day after a relevant vessel was within Stjernelaks' geofencing.
  - Kystdatahuset's API endpoint (1) for fetching tracks did not fetch all relevant tracks. Consequently, the final datasets *Labeled Days*

and *Labeled Time Series* are missing some AIS information. This is important to keep in mind when discussing the results of the analysis.

- The *Labeled Days* dataset will be used in the analysis to answer the first subpart of the research question, **Sub-RQ 1**: Can AIS data be used to predict if Stjernelaks is processing fish on any given day, regardless of the source being a waiting cage or direct vessel delivery?
- The *Labeled Time Series* dataset will be used in the analysis to answer the second subpart of the research question, **Sub-RQ 2**: Can AIS data be specifically used to predict if Stjernelaks is processing fish that has been directly delivered by a vessel on any given day?

Chapter 6

Methodology

Concerning the work done in this thesis, an overview was created of how various actors operate within the Norwegian fish farming industry. This information is updated as of spring 2023. The purpose of creating this overview was to gain an in-depth understanding of how the Norwegian fish farming industry functions, and thus attempt to map the overall interest in a product able to predict the future processing of fish. The method behind acquiring this information included a literature review, exploring available data, visiting relevant companies, and conducting interviews.

6.1 Literature Review and Data Exploration

Similar to other literature studies, this thesis is based on existing data and knowledge. As mentioned in Chapter 3 (Literature review), there exists little prior research on the applications of AIS data. Therefore our work may be considered to be pioneering the field of discovering if AIS data may have applications within the Norwegian fish farming industry. However, many of the technical aspects and deductions are based on existing knowledge within various data science and statistical topics.
Ideally, when conducting a predictive analysis such as in this thesis, we would have access to a lot more data from several actors within the Norwegian fish farming industry. Unfortunately, due to the sensitivity of this data, we were

turned down by many of the actors we contacted when attempting to acquire data. In this industry, innovative technology is extensively utilized, and there is fierce competition in the market. Knowledge is valuable information that the actors prefer to keep confidential. However, Grieg Seafood Stjernelaks responded positively to our outreaches and provided us with valuable insight and data (Stjernelaks fish processing data from Section 5.1) that are used extensively throughout this thesis.

6.2 Company Visits and Interviews

Measures were early set in place in an attempt to visit as many sites as possible to gain a more practical understanding of how the industry functioned. During our visit to Stjernelaks, we were allowed to discuss and explore what worked well and identify potential challenges for the industry. We also discovered future challenges when using AIS to describe the aquaculture. One of the challenges identified was that the wellboats always travel with a consistent $draught$[1] This is because the vessel is always carrying approximately the same weight of cargo, either in water or fish. If a vessel travels with half of its fish-carrying capacity, the remaining half is filled with seawater from the site the fish was picked up. The fish also have approximately the same density as water [Lars Martin Hetland Grieg Seafood Stjernelaks, 2023]. Initially, we had a theory that the AIS attribute $draught$ could be used to calculate how much fish each ship was loaded with. Due to this newly learned information, this proved not to be a feasible solution going forward.

6.3 Technical Framework

Given the fact that this thesis is to be considered a Proof of Concept for Blue Planets' continued research into this topic, it was important to choose a framework that can scale if the PoC proves successful. Therefore we opted for Python [Python Software Foundation, 2023] as the programming language for this thesis. Python is a good choice for data analysis projects due to several

[1] Recall from table 2.2 that draught is an AIS attribute used to record the depth of the ship's hull below the water line.

reasons, amongst others, its versatility, robust libraries, community, integration, and compatibility, and that it comes with Jupyter notebooks[2]. The code is also structured using the Cookiecutter Data Science template, which is; *a logical, reasonably standardized, but flexible project structure for doing and sharing data science work* [Peter Bull, 2023]. This template choice allows Blue Planet to easily 'pick up the pace' after the delivery of this thesis.

6.4 Can AIS Data be used to Predict Fish Processing at Grieg Seafood Stjernelaks?

To reach an answer to our research statement, we have combined all information and data gathered to create the most valid analytic approach we see possible. The way ML models are trained is by feeding them large amounts of data, allowing them to iteratively adjust their parameters until their predictions match desired outcomes [Hastie et al., 2009]. Therefore, as described in Chapter 5, for this kind of analysis, we should ideally have a lot more data available than we do to be able to properly train the best model possible. However, we only have the Stjernelaks data available, and our model is thus limited to the number of entries within that data.

Originally the main focus of this thesis was to investigate whether AIS data could be used to describe fish transportation along the coast of Norway. And, in turn, be used to predict the future supply of fish entering the market. However, to be able to do this, we would have to have sufficient sources of true data about volumes of fish transported by fish carriers. Attaining access to this data proved to be troublesome. Additionally, calculating fish volumes using the proposed regression model by Jia et al. (2019) equation 3.1 from Chapter 3 would not hold due to the special nature of the AIS attribute $draught$ for carrier vessels. Consequently, we had to deviate from the original plan. To comply with these constraints, the research questions had to be modified.

After exploring, cleaning, and verifying the integrity of the available data, we looked into approaching the research question with a classification model. Arguments can be made that a classification model might be more robust to noisy

[2] Jupyter Notebooks are interactive computing environments that allow users to create and share documents containing live code, visualizations, and explanatory text.[Jupyter Development Team, 2023]

data or outliers than regression models. This is because the goal of classification is to assign class labels, which can be less sensitive to small deviations in the data compared to regression, which predicts continuous values. When dealing with smaller datasets, simpler models tend to perform better than complex models, as they are less likely to overfit the data. This concept aligns with the principle of Occam's razor: *the simplest model that fits the data is also the most likely to generalize well to unseen data* [Domingos, 1999].

6.5 Feature Engineering

Feature engineering is the process of using domain knowledge to create features that make ML algorithms function more efficiently. It stands as an indispensable step in any successful ML project. A good feature selection process can prove the difference between a poorly performing model and a highly successful one [Zheng and Casari, 2018]. Following the steps detailed in Chapter 5 on how raw data was processed into an intermediate form (the transitions in figure 5.3), this section will now address how the final preparation of data was performed to be able to feed it into the ML models. According to Brownlee [Brownlee, 2019], ML models learn from the input data presented to them. However, not all data is equally instructive. By converting raw data into features that highlight the underlying structures and relations, the models can be allowed to learn more effectively and precisely. Better features often also lead to better model performance in terms of evaluation metrics such as *accuracy*, *sensitivity*, *precision*, *F-measure*, and *AUC-ROC* referenced later in this chapter.

The main objective of this section is to transform the intermediate data into a processed format that ML algorithms can effectively consume. This step involves selecting, transforming, and engineering the data to increase the predictive accuracy of the models.

To achieve this, several strategies are followed. These include normalizing numerical features, handling categorical variables, dealing with missing values, creating new features, and potentially reducing dimensionality when appropriate. Each of these steps aims to improve the machine learning model's performance by presenting the data to enhance the underlying learning algorithm's effectiveness [Goodfellow et al., 2016].

6.5.1 Addressing Time Variables

Time variables such as dates needed to be decomposed into separate features such as `days`, `months`, `weeks`, and `seasons`. This process aids the model by describing the cyclic patterns in the data, allowing it to capture potential seasonal or temporal influences on the outcomes. The code snippet below in Listing 6.1, depicts some of this logic.

```python
def split_date(df, date_column = "Date"):
    df.loc[:, "hour"] = df[date_column].dt.hour
    df.loc[:, "day"] = df[date_column].dt.day
    df.loc[:, "weekday"] = df[date_column].dt.weekday
    # Monday=0, Sunday=6.
    df.loc[:, "weekend"] = df[date_column].dt.weekday > 4
    df.loc[:, "week"] = df[date_column].dt.isocalendar().week
    df.loc[:, "month"] = df[date_column].dt.month
    df.loc[:, "quarter"] = df[date_column].dt.quarter
    df.loc[:, "season"] = pd.cut(df[date_column].dt.month, \\
    [0, 2, 5, 8, 11, 13], labels=[1, 2, 3, 4, 1], ordered=
   False)
    df.loc[:, "season_name"] = pd.cut(df[date_column].dt.month
   , [0, 2, 5, 8, 11, 13],
    labels=["winter", "spring", "summer", "autumn", "winter"],
     ordered=False)
    df.loc[:, "year"] = df[date_column].dt.year
    return df
```

Listing 6.1: Splitting dates into separate features.

6.5.2 Addressing Non-Numerical Data Types

Important to a majority of the feature engineering steps in this section is that ML models primarily only accept numerical inputs. In Listing 6.2 below, the `Date` column is transformed to a *Unix timestamp* [UnixTimestamp, 2023], Unix timestamps represent time in seconds since January 1, 1970, providing a numerical representation of the date-time object. However, there exist some advanced frameworks such as *TensorFlow*[3] that can handle more diverse data types, such

[3] TensorFlow is an open-source, flexible, and comprehensive machine learning and numerical computation framework developed by Google, which provides a suite of tools to develop and train complex neural network models across a variety of platforms. [TensorFlow, 2023]

as strings[4].

```
    # Changing Date column to Unix timestamps
    # 10**9 divides the integers by 10^9 to convert them to Unix timestamps (seconds since January 1, 1970).
    df['UnixDate'] = df['Date'].astype('int64') // 10**6
```

Listing 6.2: Converting date to Unix timestamp.

6.5.3 Addressing NaN Values

It is also important to handle the existence of *NaN* values[5] in datasets. This issue was addressed by implementing suitable data imputation routines to ensure the integrity of the data without introducing bias. Examples of this *NaN* handling are illustrated in `line 7` for Listing 6.3, and `line 7:8` for Listing 6.4 below. For the datasets in this thesis, imputing with zeros makes sense because the *NaN* values represent instances where no measurement of fish processing was taken.

```
1 def create_tree(df, label_variable, shuffle=False):
2     # get all columns that are not the label variable
3     feature_cols = [col for col in df.columns if col != label_variable]
4     X = df[feature_cols]  # Features
5 
6     # set Nan to 0
7     X = X.fillna(0)
8     ...
```

Listing 6.3: NaN handling of features.

```
1 def make_stjernelaks_labeled_processed_data(label = "Active"):
2     ...
3     # Fill NaN values with 0
4     df["Totalt"] = df["Totalt"].fillna(0)
5 
6     # Set labels to be 1 if Totalt is greater than 0, else 0
7     df[label] = df["Totalt"].apply(lambda x: True if x > 0 else False)
8     ...
```

Listing 6.4: NaN handling of label data.

[4] A string is a sequence of characters (like letters, numbers, and symbols) used to represent text or data in computing and programming languages.

[5] NaN values represent missing or undefined data points in a dataset.

6.5.4 Addressing Categorical Data

Categorical data comprise distinct groups or categories rather than numerical values. The ML algorithms that are used in this thesis require numerical input, which is the reason why the categorical values in our datasets, like $shiptypenor2$, were processed through the encoding strategy *One-hot encoding*[6] transforming them into a format ready to be utilized by ML algorithms. Such encoding is shown in Listing 6.5 below:

```python
# One-hot encoding
onehot_shiptypenor2 = pd.get_dummies(df["shiptypenor2"])

# Concatinate one-hot encoded columns with original dataframe
df = pd.concat([df, onehot_shiptypenor2], axis=1)

# Remove old shiptypenor2 column
df = df.drop(columns=["shiptypenor2"])
```

Listing 6.5: One-Hot encoding.

6.5.5 Addressing Feature Scaling

Despite the fact that the analysis part of the thesis relies heavily on utilizing decision trees, which do not necessitate normalization or standardization of data, it was included regardless. The rationale was to maintain a consistent pipeline, so other classification models easily can be added in the future that might be sensitive to the scale of the features. `('normalize,' MinMaxScaler())` was used to normalize the features in a fixed range from 0 to 1. The `MinMaxScaler` is found in the Scikit-learn library and is essential as it helps to avoid certain features dominating others due to their scale. They also help ML algorithms to converge faster.

[6] One-hot encoding is a process of converting categorical data into a binary format where each category is represented by a unique binary vector, with all positions set to '0' except for one position set to '1' that corresponds to the specific category [Raschka and Mirjalili, 2017].

6.5.6 Feature Selection

The last step of feature engineering is the feature selection process. Feature selection is a critical process in ML that involves selecting the most useful features in the data for training and testing the model. The goal of feature selection is to improve the model's performance by reducing overfitting, improving accuracy, and reducing training time.

From the two processed final datasets *Labeled Days* (See table 5.1) and *Labeled Time Series* (See table 5.2), an initial feature selection was conducted by utilizing domain knowledge acquired through working with field specialists during the course of this semester. This initial feature selection was conducted to remove the features that were clearly non-informative and irrelevant, such as, for example, the feature `hour`, which always was zero. These were removed because they could potentially add noise to the second round of feature selection, called *Recursive Feature Elimination*[7] (RFE). The resulting features from the first manual selection were used to create two feature sets for each of the two processed final datasets,

For the *Labeled Days* dataset the two feature sets, now referred to as *AIS inclusive feature set* and *AIS non-inclusive feature set*, are the following:
AIS inclusive feature set = `[day, weekday, week, visit*]`, and the
AIS non-inclusive feature set = `[day, weekday, week]`.

For the *Labeled Time Series* dataset, the two feature sets are:
AIS inclusive feature set = `[day, weekday, week, direct_distance_sum_norm*`,
and the AIS non-inclusive feature set = `[day, weekday, week]`.

The feature sets containing no AIS features consist of only features derived from the date, *temporal features*[8], these are `day`, `week`, `weekday`. In the AIS-inclusive feature sets, the features marked with (*) are features derived from AIS data. This partitioning of the features is done to be able to compare results from models trained with the AIS inclusive feature set and the AIS non-inclusive feature set.

[7] Recursive Feature Elimination (RFE) is a feature selection method that iteratively removes features, trains a model using the remaining features, and evaluates model performance until the optimal number of features is achieved [Guyon et al., 2002].

[8] Temporal features are characteristics derived from timestamped data, representing the progression of time or patterns that occur over time.

In constructing a feature set for predictive modeling, it is important to ensure that multiple features encapsulating the same seasonal patterns are not included. These features are likely highly correlated, as they convey redundant information. Mathematically, two variables X and Y are said to be correlated if they change together at a consistent rate, which is captured by the correlation coefficient. If X and Y represent two features encoding the same seasonal patterns, their values will likely rise and fall in tandem with the seasons, leading to a high correlation coefficient. This redundancy can be problematic as it may hamper the interpretability of the model and, in some cases, lead to overfitting.

In this study, the feature sets were carefully selected to avoid the inclusion of redundant features. For instance, the temporal features were analyzed extensively. Given that certain temporal features such as `month`, `quarter`, and `season` reflect similar seasonality patterns, only `month` was selected. Statistical methods and data visualization techniques were used to understand the correlation and *variance inflation factor* (VIF)[9] among these features.

The final chosen features capture the required seasonality and trend information without causing *multicollinearity*[10]. This selection was guided by carefully examining the VIF for each potential feature, and the decision was made to keep all features that demonstrated a VIF of less than 10, as per the commonly used rule of thumb [Gareth James, 2013]. This measure helped to ensure that the models for this thesis would not be adversely affected by multicollinearity, thus enhancing their robustness and reliability. AIS features such as $draught$ and dwt highly correlated to $grosstonnage$ and were removed due to their VIF exceeding 10. $Grosstonnage$ was prioritized because it did not contain null values.

The second round of feature selection was performed using Recursive Feature Elimination. RFE is a feature selection method that automatically selects the most relevant features in the provided set of features. The process works by recursively removing features and thus builds a model using the remaining attributes while calculating model accuracy. The process of selecting features using RFE is unique for each ML model and will be further discussed below in

[9] "Variance inflation factor measures how much the behavior (variance) of an independent variable is influenced, or inflated, by its interaction/correlation with the other independent variables."[Investopedia, 2023]

[10] Recall multicollinearity refers to a situation in statistical modeling where two or more features in a dataset are highly correlated, which can potentially skew or mislead the model's understanding of the importance of each feature when making predictions.

Sections 6.7 and 6.8.

One limitation of the feature set in this thesis is that certain variables, such as holidays and fish sickness reports, are not included. We know these could provide value. For example, if the fish is sick, it can not be delivered to waiting cages and must be processed directly. However, the Stjernelaks fish processing data started registering fish sickness in 2022. Before 2022 there were no recorded fish sicknesses. Additional data is registered and publicly available from *Barentswatch*[11] which provide their own API that can be used for integration in a potential future improved solution.

Moving forward from this section, the AIS inclusive and AIS non-inclusive feature sets (selected from the features in tables 5.1 and 5.2) are used to train, test, and validate the ML models.

[11] BarentsWatch is a comprehensive, integrated digital platform developed by the Norwegian government to provide public access to a wide range of data and services related to the marine and coastal environments of Norway [BarentsWatch, 2023].

6.6 Seasonal Naive Method

The seasonal Naive method is a forecasting technique used in time series analysis. It is a variant of the Naive Forecasting method, which perhaps is the simplest way to forecast a time series. The Naive Forecasting method simply sets all forecasts to be the value of the last observation and is denoted by equation 6.1 below:

$$\hat{y}_{h|T} = y_T \tag{6.1}$$

where:

h $=$ forecast horizon
y$_T$ $=$ last observation

Whereas the Seasonal Naive method sets the next period's value equal to the current period's value. It is denoted by equation 6.2 below:

$$\hat{y}_{T+hT} = y_{T+h-m(k+1)} \tag{6.2}$$

where:

$_{T+hT}$ $=$ The forecasted value at time $T + hT$ (the h-th future seasonal cycle after time T)
m $=$ The seasonal period
y$_{T+h-m(k+1)}$ $=$ The actual value at time $T + h$ minus the product of the seasonal length m and $(k + 1)$, where k represents the number of complete seasons that have passed by time $T + h$

The seasonal naive method is a useful starting point, or baseline, to the analysis of this thesis because it is very easy to calculate, and any more complicated method should at least outperform it to be considered effective. The Seasonal Naive method assumes that the future will look exactly like the corresponding period in the past. As for this case, if there exists monthly Stjernelaks fish processing data for the Monday of week 2 January 2023, the prediction for the Monday of week 2 January 2024, would be that actual value. In practice, this would not hold, and it would not be a very good predictive model. However, when used as a comparison to the other Machine Learning models used in this thesis it works as a good baseline to determine the effectiveness and usefulness of the other models.

6.7 Decision Tree

As mentioned in Chapter 4, decision trees are a powerful tool used in ML classification and prediction tasks and are one of the models that will be compared to the Seasonal Naive method (Section 6.6). The decision trees in this thesis were implemented using the $DecisionTreeClassifier()$ from the Scikit-learn library [Pedregosa et al., 2011]. See Chapter 7 for detailed results from each of the following steps.

6.7.1 Decision Tree Refinement, Step 1: Tree

An iterative refinement approach was used to reach the best possible decision tree results. It is considered good practice to start simple, and gradually add complexity to learn from each step and make informed decisions about what to try next. First, a default decision tree was created using $DecisionTreeClassifier()$ with Scikit-learns default *hyperparameters*[12]. The default decision tree was trained and tested on the AIS inclusive and non-inclusive feature sets.

6.7.2 Decision Tree Refinement, Step 2: RFE Tree

The next step in the iterative refinement of the decision trees was implementing them using Recursive Feature Elimination. This is the second feature selection following the manual one mentioned in Section 6.5.6. RFE was implemented on both feature sets. If the RFE implementation with access to AIS inclusive feature set does not eliminate AIS features from the mix, then it determines that the AIS features are amongst the most important ones. If that is the case,

[12] Hyperparameters are adjustable parameters that you set prior to training an ML model, which influence the model's learning process and overall performance on the dataset [Goodfellow et al., 2016].

and the model using these features scores better than the RFE implementation that is not using AIS, then it is possible to conclude that AIS data are improving the model's predictive capabilities.

The number of selected features is determined by the implementation of RFE. The default implementation selects the square root of the total available features. If the result is not an integer, it is floored to the closest integer. E.g., $\sqrt{3} \approx 1.732$, so $\lfloor \sqrt{3} \rfloor = 1$. Refinement step 2 uses the default implementation. However, step 3 is further refined to use a specified number of features. This is further explained in the next Sections 6.7.3 and 6.8.1.

6.7.3 Decision Tree Refinement, Step 3: Grid RFE Tree

The final step in the iterative refinement of the decision trees was to tune their hyperparameters using the Scikit-learn function
$GridSearchCV$ [Scikit-learn Developers, 2023a]. This function uses a grid of specified hyperparameter values. It performs a cross-validated training process for each parameter, resulting in the combination that provides the best model performance. Listing 6.6 below illustrates how this hyperparameter grid was defined for this step:

```
param_grid = {
    'feature_selection__n_features_to_select': [3],
    'decision_tree__criterion': ['gini', 'entropy'],
    'decision_tree__max_depth': [None, 2, 4, 6, 8, 10],
    'decision_tree__min_samples_split': [2, 5, 10],
    'decision_tree__min_samples_leaf': [1, 2, 4],
}
search = GridSearchCV(pipeline, param_grid, cv=tscv, scoring='roc_auc')
```

Listing 6.6: Hyperparameter tuning using GridSearchCV.

$GridSearchCV$ trains and evaluates the model with each variation of the possible parameters listed in the code snippet, and stores the best result for each iteration. The evaluation is performed with rolling cross-validation (Section 4.3.2) and is scored using *ROC-AUC* which is one of the evaluation metrics that will be further discussed in Section 6.10.6.

6.8 Random Forests

As mentioned in Chapter 4 random forests are a powerful tool used in ML classification and prediction. It is also one of the models that will be compared to the Seasonal Naive method (Section 6.6). The random forests in this thesis were implemented using the $RandomForestClassifier()$ from the Scikit-learn library [Scikit-learn Developers, 2023b].

6.8.1 Random Forests Refinement, Step 3: Rand RFE RF

For random forests, the same refinement process was followed as for decision trees in the previous section. Refinement step 1 and 2 looks identical to the refinement steps for decision trees, therefore these are skipped, but the results can be viewed in Chapter 7. For the random forests refinement step 3, the random forests implementation includes a hyperparameter for the number of decision trees in the forest. See Listing 6.7 below for how the hyperparameter grid was implemented for this step.

```
1 param_grid = {
2     'feature_selection__n_features_to_select': [3], # OR
     [2,3,4]
3     'classifier__n_estimators': [100, 200, 500],
4     'classifier__max_features': ['sqrt', 0.2, 0.5],
5     'classifier__max_depth': [None, 2, 4, 6, 8, 10],
6     'classifier__min_samples_split': [2, 5, 10],
7     'classifier__min_samples_leaf': [1, 2, 4],
8 }
9 search = RandomizedSearchCV(pipeline, param_grid, n_iter=
    n_iter, cv=tscv, scoring='roc_auc')
```

Listing 6.7: Hyperparameter tuning using RandomizedSearchCV.

Line number 2 in listing 6.7 specifies that the `n_features_to_select` should either be `[3]` or `[2, 3, 4]`. The model tuned on `[3]` is referred to as 'Rand RFE RF Fixed,' and the model tuned with `[2, 3, 4]` is referred 'Rand RFE RF' in Chapter 7.

Due to the increased complexity and additional hyperparameters in random forests compared to single decision trees, hyperparameter tuning can be considerably more time-consuming. This time complexity is particularly significant when using Scikit-Learn, which doesn't support GPU acceleration, making the computation slower. This leads to an exponential time increase in training the model. To mitigate this issue, $RandomizedSearchCV()$ was used instead of using the exhaustive $GridSearchCV()$, which was used for hyperparameter tuning of the decision trees. $RandomizedSearchCV()$ doesn't search the entire parameter space but rather samples a fixed number of parameter settings based on the given distributions. This results in a faster and more efficient search process, which is advantageous when dealing with more complex models like random forests. The scoring system used for $RandomizedSeachCV()$ is *AUC-ROC* which is the same as for $GridSearchCV()$.

6.9 Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW), presented in Chapter 4, is a prominent method in this thesis analysis and is only used on the *Labeled Time Series* dataset. DTW allows for measuring the similarity between two temporal sequences of AIS positions that may vary in frequency and duration. The essence of DTW is its ability to align sequences in a non-linear fashion, accommodating distortions and shifts in the time dimension. This is important because we want to look at time series data that may have similar patterns but are not perfectly synchronized in time (recall the example from Section 4.6). When observing two different vessels that deliver fish directly to a processing facility, their movement can be very similar, but one might spend a longer time at the dock than the other vessel. DTW is perfect for these scenarios where it can capture the similarities regardless of different time spent at the dock or elsewhere along the voyage.

Based on conversations with Stjernelaks, we know that all vessels dock on the north side when directly delivering fish for processing. Therefore, capturing this delivery with a small geofencing around the docking area might be intuitive. This will require specific knowledge about exactly where the docking area is and will not be easily applicable when considering locations other than Stjernelaks. This is why DTW was chosen because it does not require any specific prior knowledge about the location and can be universally applied to other locations.

The idea is that the time series for vessels delivering fish directly for processing should have a small DTW distance compared to other vessels' time series delivering fish directly to Stjernelaks. When a single time series has been compared to all other time series of vessels that are known to deliver fish directly ($Direct$ = True) then these distances can be summed. If this sum is relatively low, the summed time series describes a vessel delivering fish directly. If the sum is relatively high, the time series probably describes a vessel that did not deliver fish. To describe this we constructed the following formula:

$$\sum_{i=1}^{n} DTW(Ts_{direct,i}, Ts_{current}) < T \tag{6.3}$$

where:

n = Number of time series labeled as $Direct$ = True;
DTW = Function to calculate the DTW distance between two time series;
Ts$_{direct,i}$ = The i-th time series labeled as $Direct$ = True;
Ts$_{current}$ = The time series under consideration;
T = Relative threshold, determining what is considered relatively high or relatively low.

The same can be done for time series labeled $Direct$ = False. If the sum is relatively low, the time series probably describe a vessel that did not deliver fish. And the opposite, if the sum is low, the time series probably describes a vessel that delivered fish.

These two sums are used as features in the AIS inclusive feature set from the *Labeled Time Series* dataset. These features are direct_distance_sum_norm* and not_direct_distance_sum_norm* (see table 5.2). However, not_direct_distance_sum_norm* was removed due to high VIF because it is highly correlated to direct_distance_sum_norm*. The decision tree and random forest are fed these features and try to find the thresholds between relatively high or low distances.

The application of Dynamic Time Warping (DTW) is computationally intensive, with a runtime that grows exponentially as the number of time series increases. However, only 422 of the time series are labeled, so computing the DTW for the unlabeled time series would not yield valuable insights. This significantly reduces the computational load, making the task more manageable.

Nonetheless, if the entire set of time series were labeled, strategies to cope with the computational complexity would be required. Several strategies could be implemented, including (1) Pruning and early stopping to halt unnecessary computations. (2) Approximate the time series data using *Piecewise Aggregate Approximation* (PAA)[13] [Keogh et al., 2001] or *Symbolic Aggregate Approximation* (SAX)[14][Lin et al., 2007], reducing data dimensionality while preserving critical structure. (3) Clustering or sampling methods could be used to identify representative subsets of the time series. (4) Lowering the resolution of the time series could also be an effective strategy. And lastly, (5) Exploring alternatives to DTW that may provide a better trade-off between computational efficiency and accuracy.

While processing 422 time series is less challenging than handling the original 2000, it still necessitates substantial computational power. To address this, we employ *multiprocessing*[15], which allows us to capitalize on our available computational resources. In Listing 6.8 below, multiple tasks are initialized in `line 1`, with each task representing a pair of time series to be compared. Next, in `line 3`, a pool of worker processes corresponding to the number of CPU cores is created. Finally, in `line 4`, DTW calculations are applied to each task using the Pool's `imap` function.

```
1 tasks = [((time_series_data[i][["longitude", "latitude"]].
    to_numpy(), time_series_data[j][["longitude", "latitude"]].
    to_numpy(), i, j)) for i in range(n_time_series) for j in
    range(i + 1, n_time_series)]
2
3 with mp.Pool(mp.cpu_count()) as pool:
4     results = list(pool.imap(DTW.calculate_dtw_distance, tasks
    ))
```

Listing 6.8: DTW calcualations with parallel processing.

[13] Piecewise Aggregate Approximation (PAA) is a dimensionality reduction technique used in time series mining that transforms the original time series data into a representation consisting of a sequence of equal-sized segments where each segment is represented by its mean value.

[14] Symbolic Aggregate approXimation (SAX) is a symbolic representation of time series that reduces dimensionality and allows for the application of data mining methods by assigning symbols to ranges of data.

[15] Multiprocessing is a method of executing multiple concurrent processes in a system, with each process running on a separate CPU or core, as opposed to a single process at any one instant.

Moreover, during these DTW calculations, it's crucial to avoid any *data leakage* complications. Data leakage refers to a situation where information from outside the training dataset is used to train the model. This can lead to overly optimistic and misleading measures of model performance. One effective strategy to prevent data leakage is to split the time series into training and test sets before performing the DTW calculations. This separation ensures that the DTW calculations are carried out independently on the training and test sets. Not following this partitioning strictly could result in data leakage. DTW calculations involving a blend of time series from both the training and test sets could inadvertently become features in the training set. Therefore, the necessity of conducting DTW calculations strictly within the confines of the assigned training set cannot be overstated.

The specific implementation of the Dynamic Time Warping (DTW) calculations in this study leverages the Python package `fastdtw` [Project, 2023] and employs the Haversine formula used and introduced in Section 5.2 for distance computation. Listing 6.9 below illustrates how `fastdtw` is called to calculate DTW distance for two distinct time series.

```python
def calculate_dtw_distance(time_series_pair):
time_series_1, time_series_2 = time_series_pair
distance, _ = fastdtw.fastdtw(time_series_1, time_series_2,
    dist=haversine_distance)
return distance
```

Listing 6.9: DTW calculation implementation.

6.10 Model Evaluation Metrics

To evaluate and compare the prediction methods and models in this thesis, we will make use of five key metrics: *Accuracy*, *Sensitivity*, *Precision*, *F-measure* and the *AUC-ROC*. By employing multiple evaluation metrics, we can gain a more nuanced understanding of the model's performance across various dimensions, instead of relying on a single generalized measure.

6.10.1 Accuracy

Accuracy is a fundamental metric for binary classification, providing a holistic overview of a model's performance. It provides a ratio of the correctly predicted instances to the total instances in the dataset.

$$accuracy = \frac{\text{cases predicted right}}{\text{all cases}} \tag{6.4}$$

6.10.2 Sensitivity

Sensitivity is also a fundamental metric when using binary classification. It provides a measure for the proportion of actual positive cases that the model correctly identified.

$$sensitivity = \frac{\text{True Positive}}{\text{Actual positive}} \tag{6.5}$$

6.10.3 Specificity

Specificity, also known as the true negative rate, is another fundamental metric in binary classification that measures the proportion of actual negative instances that the model correctly identifies. It's particularly important in situations where the cost of a false positive is high.

$$specificity = \frac{\text{True Negative}}{\text{Actual Negative}} \tag{6.6}$$

6.10.4 Precision

The *precision* metric provides a ratio for the true positives to all the instances that the model predicted as positive.

$$precision = \frac{\text{True positive}}{\text{Predictive positive}} \tag{6.7}$$

6.10.5 F-measure

The *F-measure*, known as the *F1* score, conveys a harmonic mean of *precision* and *sensitivity*, providing a balanced measure of the model's performance. It encapsulates both the model's ability to correctly identify positive instances *(sensitivity)* and its ability to avoid false alarms *(precision)* into a single metric.

$$F - measure = \frac{2 * \text{Precision} * \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}} \tag{6.8}$$

6.10.6 Area Under the ROC Curve (AUC-ROC)

According to Fawcett [Fawcett, 2006], The Area Under the Receiver Operating Characteristics Curve, often abbreviated as AUC-ROC, is another key metric for binary classification problems. The ROC curve illustrates the performance of the classification model at all classification thresholds, plotting the true positive rate *(sensitivity)* against the false positive rate (1-*specificity*) at various threshold settings.

The AUC-ROC is the area under this ROC curve, ranging from 0 to 1. An AUC-ROC value of 0.5 implies that the model has no discrimination capacity to distinguish between positive and negative classes, essentially performing no better than random guessing. On the other hand, an AUC-ROC of 1.0 signifies that the model has perfect discrimination ability, correctly classifying all instances.

The AUC-ROC can be interpreted as the probability that a randomly chosen positive instance is ranked more highly than a randomly chosen negative instance by the classifier, assuming that positive ranks higher than negative, denoted by:

$$AUC - ROC = P(\text{score}(X_+) > \text{score}(X_-)) \tag{6.9}$$

where X_+ and X_- are randomly chosen positive and negative instances respectively, and score(X) is the classifier's scoring function.

Generally, according to Fawcett, an AUC-ROC score of 0.7 to 0.8 is considered *acceptable*. 0.8 to 0.9 is considered *excellent*, and more than 0.9 is considered *outstanding*.

While *accuracy*, *precision*, *sensitivity*, *specificity*, and the *F-measure* provide valuable insight into our classification model's performance, the Area Under the Receiver Operating Characteristic *(AUC-ROC)* metric will serve as our primary evaluation tool due to its distinct advantages. Unlike *accuracy*, which can present a misleadingly optimistic view of the model's performance when class distributions are imbalanced, *AUC-ROC* remains robust and unaffected by the prevalence of each class. Compared to *precision* and *sensitivity*, which only reflect the model's performance at a specific threshold, *AUC-ROC* evaluates the model's discriminative power across all possible thresholds. This makes it an ideal tool for the case of this thesis, where various external factors could influence the choice of a threshold, and the trade-off between *sensitivity* and *specificity* needs to be carefully assessed. Furthermore, while providing a balanced view of *precision* and *sensitivity*, the *F-measure* still hinges on a specific threshold. *AUC-ROC*, on the other hand, encapsulates the model's performance more comprehensively. Consequently, due to its robustness to class imbalance and its versatility in capturing the model's performance across all thresholds, *AUC-ROC* stands as the most significant metric in evaluating the classifier's performance.

6.11 Chapter Summary

Implementation choices mentioned in this chapter were all done in an attempt to ensure that the final solution is scalable. In this way, the implementation facilitates further research and development of Blue Planet's hypothesized product.

In turn, the six key evaluation metrics will be applied to the results from the baseline, decision trees, and random forests. By applying these evaluation metrics, the strengths and weaknesses of each of the models may be identified. It also makes it possible to determine their effectiveness when compared to the baseline, which is a naive and simple prediction method. To be considered an effective method, results should, at the very least, outperform the baseline's results. All models have also been implemented in such a manner that makes it possible to measure performance when AIS data is used and not used. The results will be presented in the next chapter.

Chapter 7

Results

This chapter will present the results after performing the extensive data handling process (Chapter 5) and implementing all the steps mentioned in Chapter 6. This chapter aims to highlight the difference the inclusion of AIS has on the performance of the predictive models versus when no AIS is used. The performance of each model is recorded using the evaluation metrics mentioned in the previous chapter. These results will be measured against the baseline which is either *the Seasonal Naive method* to answer **Sub-RQ 1**, and *The Naive method* to answer **Sub-RQ 2**. The models should beat the baselines in order to be considered effective.

7.1 Labeled Days Results

Recall that the *Labeled Days* dataset referred to in this thesis is the following:

Column	**Description**
day	Represents a specific day of the month.
weekday	Represents a specific day of the week, with values ranging from 0 to 6.
week	Represents a specific week number within a year.
visit*	Indicates if a relevant vessel has sailed within a 500m radius of Stjernelaks a specific day.
$Active$	(Label) A boolean indicator of Stjernelaks activity status on a specific day.

Table 7.1: *Labeled Days* dataset with description. Features marked with (*) are AIS features. $Active$ is the label the models will try to predict.

The AIS inclusive feature set is `[day, weekday, week, visit*]`, while the AIS non-inclusive feature set is `[day, weekday, week]`.

7.1.1 Labeled Days Baseline: Seasonal Naive Method

When employing the *Labeled Days* dataset with the Seasonal Naive Method, more specifically the 'Naive Weekday/Weekend' which is explained later, we get the following results displayed in table 7.2:

Evaluation metric	Score
Accuracy	0.839
Sensitivity	0.814
Specificity	0.867
Precision	0.875
F-measure	0.843
AUC-ROC	0.841

Table 7.2: Seasonal naive method results.

The results demonstrate that the Seasonal Naive Method achieved an *accuracy* of `0.839`, meaning it correctly predicted whether Stjernelaks was $Active$ or $Inactive$ in 83.9% of all cases. This denotes a significant level of predictive power. The *sensitivity* score of `0.814` suggests that the method is quite proficient at identifying $Active$ states, correctly doing so in 81.4% of instances. However, with a *specificity* of `0.867`, the model identifies $Inactive$ states correctly 86.7% of the time. While this is a good rate, this could potentially lead to false alarms, incorrectly predicting Stjernelaks as $Inactive$ when it is, in fact, $Active$. The method exhibits a *precision* of `0.875`, which means when it predicts Stjernelaks to be $Active$, it is correct 87.5% of the time. Although this is a reasonably high accuracy rate, there is potential for further improvement. The *F-measure*, the harmonic mean of *precision* and *sensitivity*, is `0.843`, suggesting a good balance between these two metrics, which is often desirable. Lastly, the *AUC-ROC* score of `0.841` indicates that the model has an 84.1% chance of correctly distinguishing between $Active$ and $Inactive$ instances for any randomly chosen pair. While this is an excellent score, it implies room for more refined or complex models to potentially improve this result.

Exploring Possible Alterations to the Seasonal Naive Method

This subsection explores possible alterations to the Seasonal Naive method to exhaust potential enhancements, offering an in-depth understanding of how these alterations impact model performance. The alterations encompass different aspects of temporal patterns and incorporate additional context, such as, for example, visits by relevant vessels through AIS data for 'Naive Visit' (below), to provide a more comprehensive picture of the activity of Stjernelaks. This investigation aims to identify a robust and versatile baseline with which more sophisticated models like decision trees and random forests can be compared.

The different alterations are defined as the following:

- 'Naive 365 days ago': Predicts Stjernelaks' activity status by directly using its activity status from the same date 365 days prior.
- 'Naive Weekday': Predicts Stjernelaks' activity based on its activity status from the same weekday of the corresponding week in the previous year.
- 'Naive Weekend': Predicts Stjernelaks' activity status based on whether the current day is a weekend. $Active$ for all weekdays (mon-fri), and $Inactive$ for all weekends (sat-sun).
- 'Naive Weekday/Weekend': Improves the method's robustness by combining the weekend and weekday predictions. Taking into account both the specific day of the week and whether it's a weekend or a weekday.
- 'Naive Visit': Incorporates AIS data and predicts Stjernelaks' activity status based on whether a relevant vessel visit occurred.
- 'Naive Visit/Weekend': Further refines the model by combining predictions based on vessel visits and whether it's a weekend, leveraging both vessel activity and temporal patterns for a more nuanced prediction.

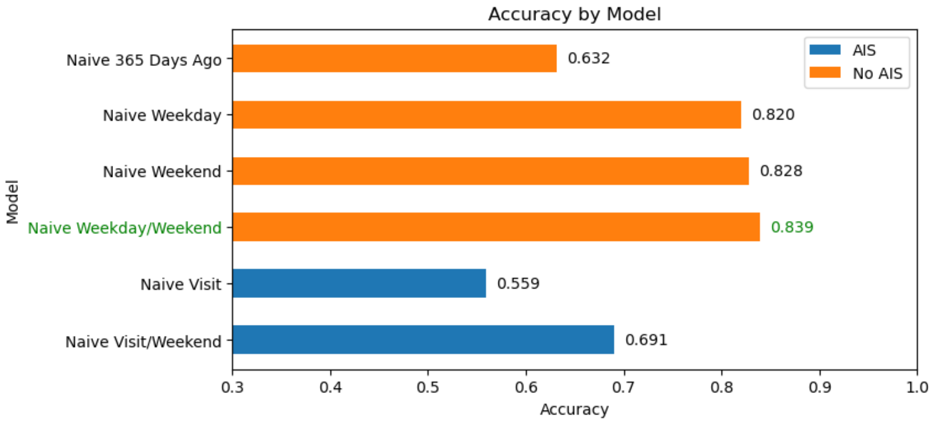


Figure 7.1: *Accuracy* for different Seasonal Naive method alterations.

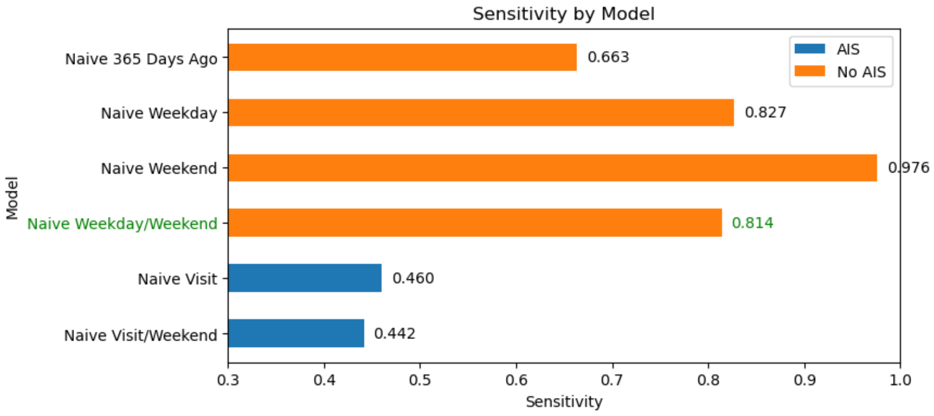


Figure 7.2: *Sensitivity* for different Seasonal Naive method alterations.

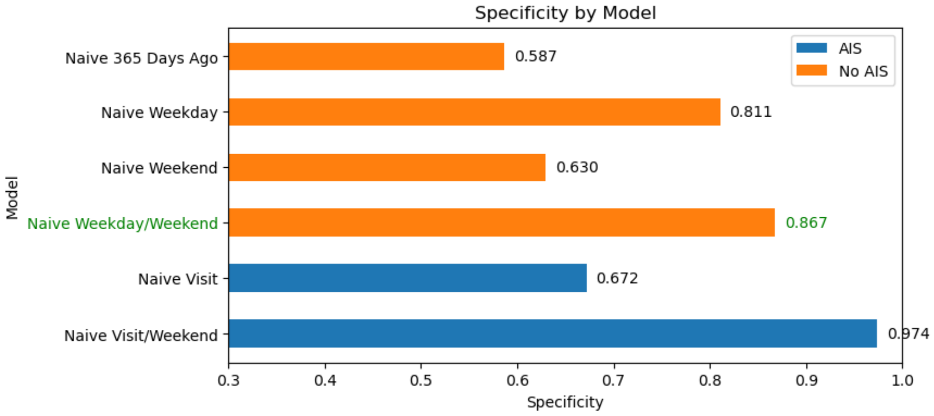


Figure 7.3: *Specificity* seasonal naive method alteration.

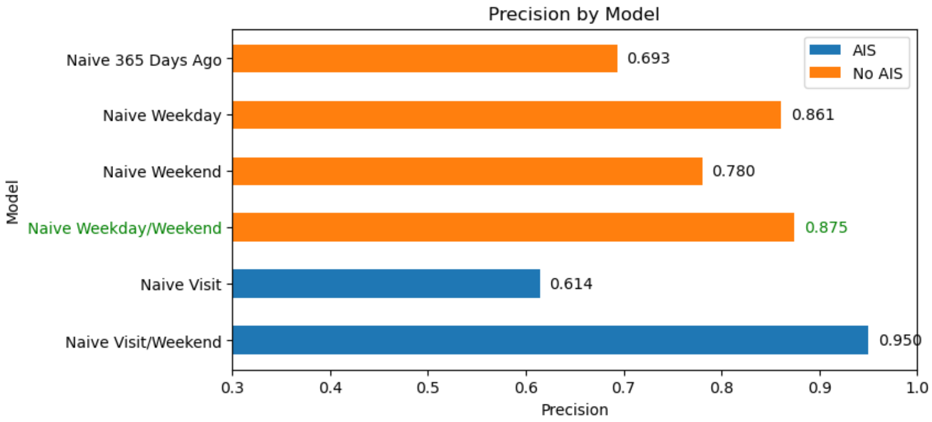


Figure 7.4: *Precision* seasonal naive method alteration.

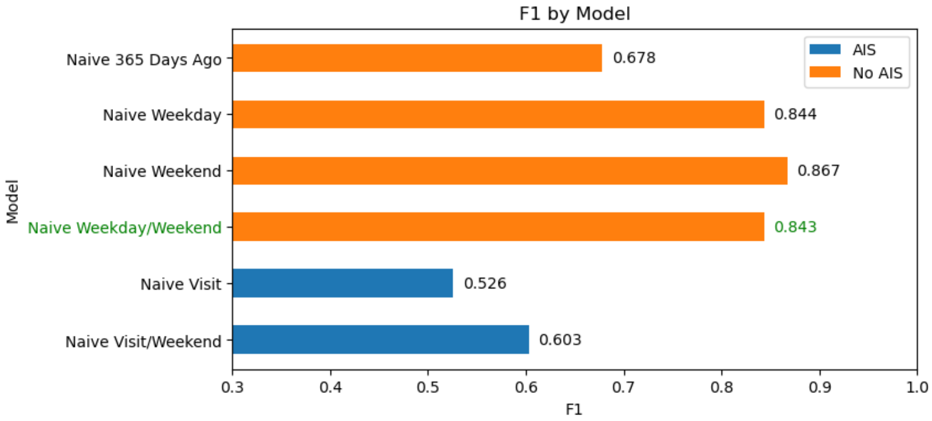


Figure 7.5: *F-measure* seasonal naive method alteration.

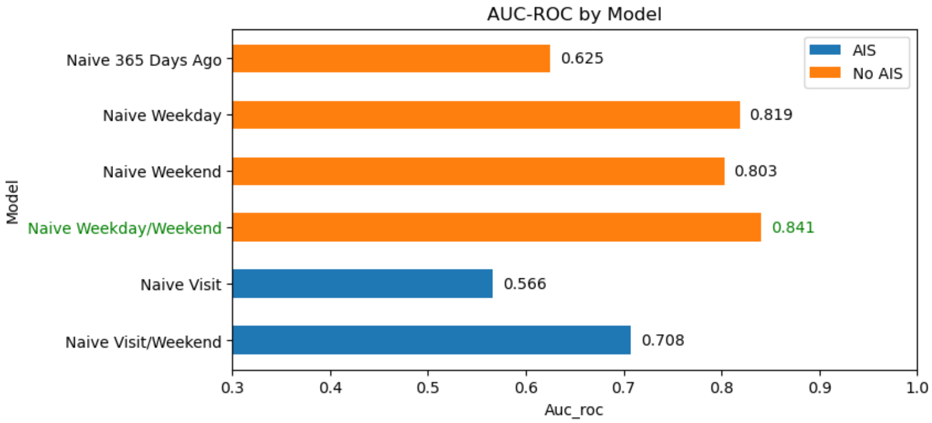


Figure 7.6: *AUC-ROC* seasonal naive method alteration.

After investigating the results from the alterations in the tables above, we observe that 'Naive Weekday/Weekend' (results marked with green) provided balanced results across the metrics with scores above 0.8 for all of the evaluation metrics. Rather than solely relying on one factor - either temporal patterns (weekends/weekdays) or historical activity data (from last year), this method leverages both, which may explain its greater performance compared to other alterations. Since it also provided the highest score for the *AUC-ROC* metric, this method was selected as the baseline to beat for the more sophisticated models.

The figure 7.7 below, visualizes the strong *AUC-ROC* performance for the 'Naive Weekday/Weekend' alteration of the Seasonal Naive Method. The sharp rise of the ROC curve indicates that the model is capable of achieving a high true positive rate at a very low false positive rate. This high true positive rate achieved at a low false positive rate means that the model performs well in correctly predicting $Active$ state for Stjernelaks while minimizing the misclassification of $Inactive$ state for Stjernelaks as $Active$.

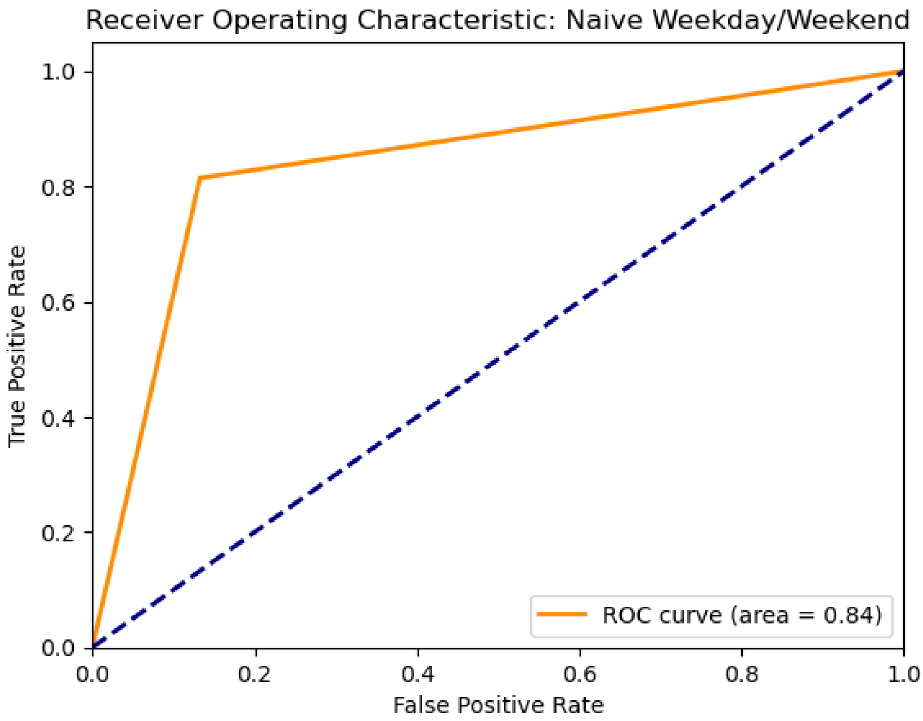


Figure 7.7: *AUC-ROC* for the 'Naive Weekday/Weekend' alteration of the Seasonal Naive Method.

7.1.2 Labeled Days Results vs Baseline

In the following section, a comparative analysis of the ML models mentioned in Chapter 6, both fed the AIS inclusive and the AIS non-inclusive feature sets, benchmarked against the recently established baseline, will be presented. The selected baseline is the 'Naive Weekday/Weekend' alteration of the Seasonal Naive method, which demonstrated the highest *AUC-ROC* score compared to the other alterations.

The different models across the `y-axis` in the figures below are defined as the following:

- 'Naive Weekday/Weekend': The baseline, mentioned in the previous Section 7.1.1.
- 'Tree': The default decision tree mentioned in Section 6.7.1.
- 'RFE Tree': The decision tree after Recursive Feature Elimination, mentioned in Section 6.7.2.
- 'Grid RFE Tree': The decision tree after both Recursive Feature Elimination and $GridSearchCV()$, meaning that hyperparameters are tuned. Mentioned in Section 6.7.3.
- 'RFE RF': Random forests after Recursive Feature Elimination, mentioned in Section 6.8.1.
- 'Rand RFE RF Fixed': The random forests after both Recursive Feature Elimination and $RandomizedGridSearchCV()$, meaning that hyperparameters are tuned as mentioned in Section 6.8.1, the number of features are fixed to 3 features.
- 'Rand RFE RF': Similar to 'Rand RFE RF Fixed, however the number of features is not fixed and can be any number in the hyperparameter grid.

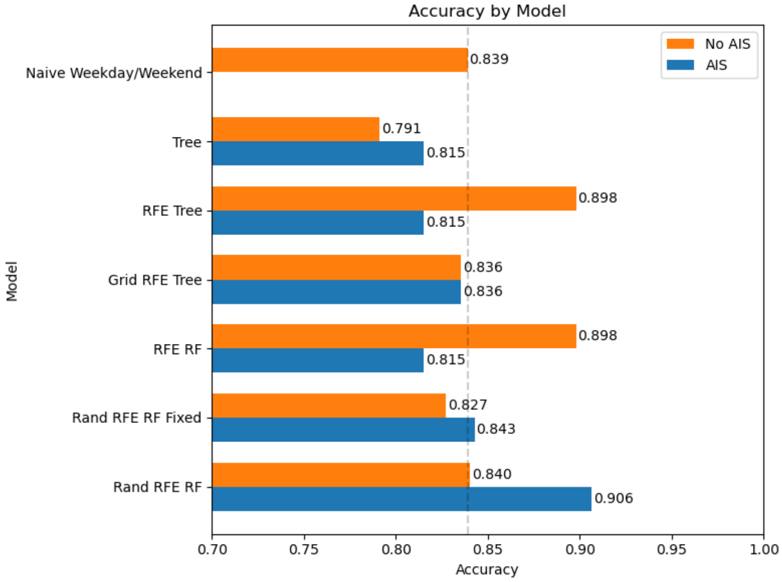


Figure 7.8: *Accuracy* by Model versus baseline.

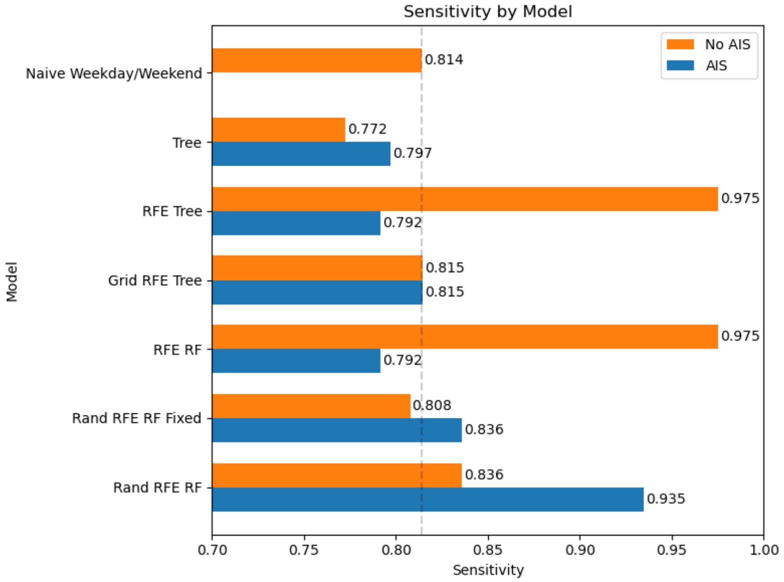


Figure 7.9: *Sensitivity* by Model versus baseline.

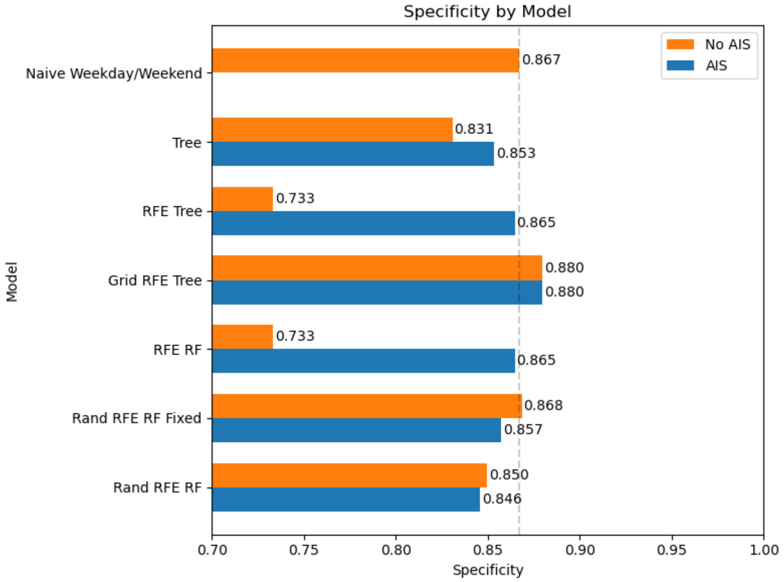


Figure 7.10: *Specificity* by Model versus baseline.

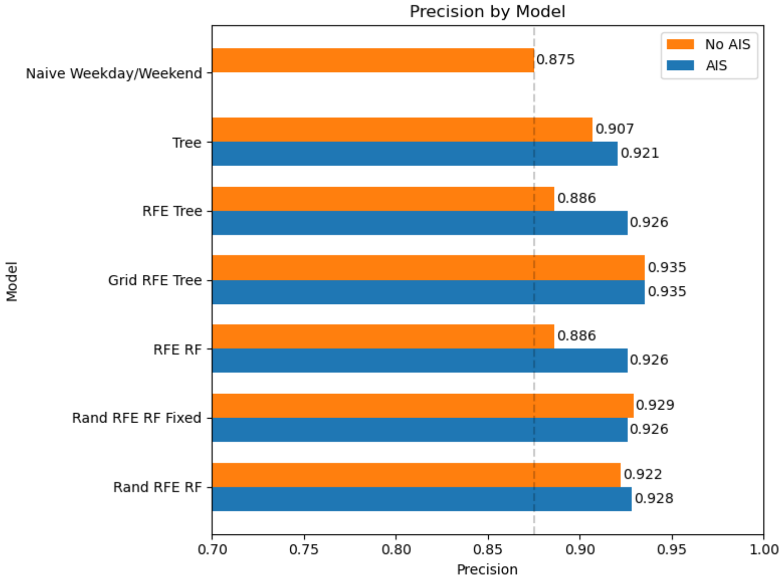


Figure 7.11: *Precision* by Model versus baseline.

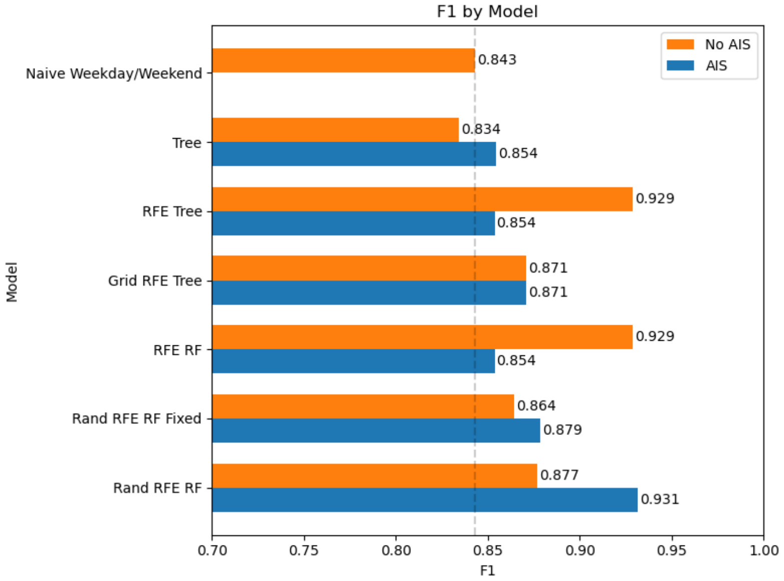


Figure 7.12: *F-measure* by Model versus baseline.

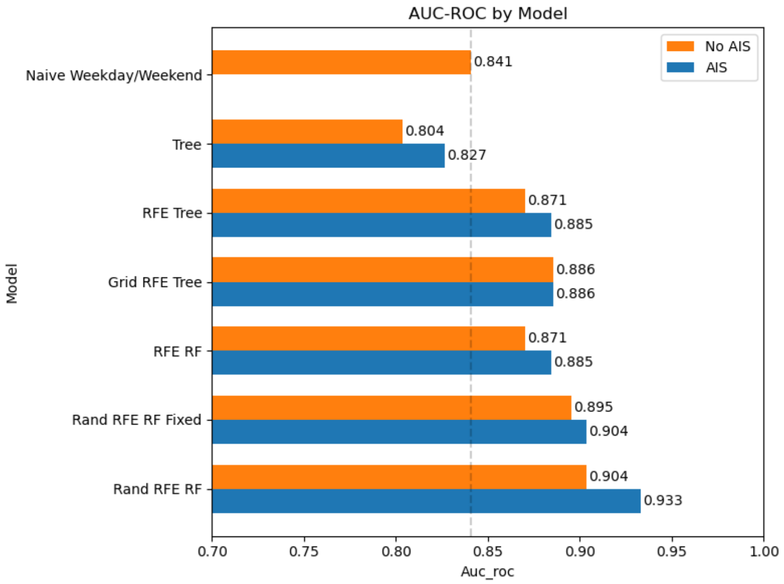


Figure 7.13: *AUC-ROC* by Model versus baseline.

From the results above, we see that the random forests model 'Rand RFE RF,' augmented with Recursive Feature Elimination and $RandomizedGridSearchCV()$ and trained on the AIS inclusive feature set, emerged as the best-performing model in terms of *AUC-ROC*. This model distinguished itself from the rest with an *AUC-ROC* score of `0.933`, suggesting a significantly superior ability to differentiate between $Active$ and $Inactive$ days compared to the other models and the baseline. The model was not only *outstanding* in terms of *AUC-ROC*, but it also demonstrated high performance across all the other metrics, outperforming the baseline for all of them.

Additionally, the performance improvement with the AIS inclusive feature set is noteworthy. The AIS inclusive feature set, containing information regarding visits from relevant vessels, has seemingly enriched the feature space and helped the model capture more complex patterns in the data. The substantial increase in *AUC-ROC* when using the AIS inclusive feature set underscores the importance of feature selection and the use of domain-specific information to bolster the predictive power of ML models in this context.

The figure 7.14 below illustrates the *outstanding AUC-ROC* performance of the 'Rand RFE RF' model utilizing both RFE and $RandomizedGridSearchCV()$ trained on the AIS inclusive feature set. Compared to the *AUC-ROC* plot of the 'Naive Weekend/Weekday' illustrated in figure 7.7, we see an even steeper initial ascent. This shows that the model is extremely capable of identifying true positive cases at an exceptionally low false positive rate. This means that this model is superior to the baseline in correctly predicting the $Active$ state while further minimizing the misclassification of the $Inactive$ state as $Active$.

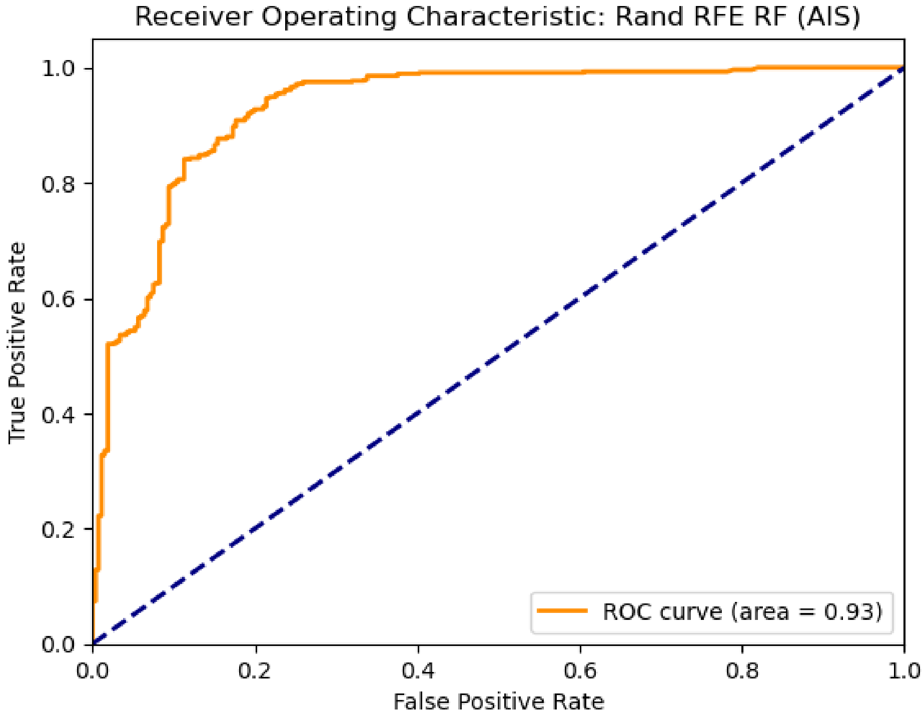


Figure 7.14: *AUC-ROC* for the 'Rand RFE RF.'

The 'wiggly' nature of the ROC curve indicates a higher granularity in the model's performance across various threshold settings, providing a more detailed representation of its performance characteristics. This could be attributed to more complex model architecture, additional data points, or the effect of the AIS inclusive feature set. With an *AUC-ROC* score of 0.933, this model demonstrates superior predictive capability and the potential for even greater performance, given its more nuanced ROC curve.

The Feature Importance of Rand RFE RF

In the AIS non-inclusive feature set, the most crucial features as determined by their importance scores were 'weekday' and 'day' with the importance scores listed in the table 7.3 below.

Feature	Importance
day	0.726299
weekday	0.273701

Table 7.3: Feature Importance, AIS non-inclusive feature set.

However, a notable change was observed when AIS features were included in the feature set. While 'weekday' and 'week' remained important features, 'visit*,' a feature specific to AIS data, appeared as a significant predictor with an importance score of 0.09. The 'day' feature decreased in importance, signifying that the inclusion of the AIS feature shifted the model's reliance from temporal towards vessel-specific importance. See table 7.4 below.

Feature	Importance
weekday	0.719927
week	0.174250
visit*	0.090585
day	0.015237

Table 7.4: Feature Importance, AIS inclusive feature set.

The inclusion of the AIS feature improved the model performance. The *AUC-ROC* increased by 3.2% from 0.904 (without AIS) to 0.933 (with AIS). This result supports the hypothesis that AIS data, providing context-specific vessel information, add value to the model's predictive performance, ultimately enhancing its generalizability and reliability.

As previously mentioned, $RandomizedGridSearchCV()$ optimizes the hyperparameters for the random forests. For the best-performing model 'Rand RFE RF' with AIS inclusive feature set, the optimal hyperparameters from the hyperparameter grid mentioned in Section 6.8.1 for 'Rand RFE RF' were the following:

- The number of features to select was 4;
- The number of trees in the forest was 100;
- The minimum number of samples required to split an internal node was 2;
- The minimum number of samples required to be a leaf node was 2;
- The maximum number of features to consider when looking for the best split was 0.2;
- The maximum depth of the tree was 4.

The configuration of these hyperparameters contributed to the superior performance of the model and subsequently the increase to the *AUC-ROC* metric observed when AIS features were included.

7.2 Labeled Time Series Results

Recall that the *Labeled Time Series* dataset referred to in this thesis is the following:

Column	Description
day	Represents a specific day of the month.
weekday	Represents a specific day of the week, with values ranging from 0 to 6.
week	Represents a specific week number within a year.
direct_distance_sum_norm*	Represents the sum of all DTW distances between a specific time series and all other time series labeled as True. This sum is then normalized (0 to 1).
closest_distance_to_stjernelaks*	Represents the Euclidean distance in kilometers between Stjernelaks and the specific time series' closest position to Stjernelaks.
$Direct$	(Label) A boolean indicator of whether Stjernelaks received fish directly from a relevant vessel on a specific day.

Table 7.5: *Labeled Time Series* dataset with description. Features marked with (*) are AIS features. $Direct$ is the label the models will try to predict.

The AIS inclusive feature set is `[day, weekday, week, direct_distance_sum_norm*]`, while the AIS non-inclusive feature set is `[day, weekday, week]`.

- The *closest_distance_to_stjernelaks** is an AIS feature only used to establish the baseline: Naive Method.
- *direct_distance_sum_norm** is derived from Dynamic Time Warping distances.

7.2.1 Labeled Time Series Baseline: Naive Method

Similarly to the analysis of *Labeled Days* dataset, we established a baseline for comparing the performance of the ML models on the *Labeled Time Series* dataset. An equally extensive process was performed, and we found the best baseline not using AIS to be 'Naive Weekend.' This is not a Seasonal Naive method because it only uses data from today and no past observations. The alteration 'Naive DTW Visit/Weekend' is also shown in the results because it outperforms the other models and the baseline. The baseline, 'Naive Weekend', 'Naive DTW Visit/Weekend' and 'DTW Grid RFE RF' is defined as follows:

- 'Naive Weekend': Predicts $Direct$ label for a time series based on whether the current day is a weekend. $Direct$ = True for all weekdays (mon-fri), $Direct$ = False for all weekends (sat-sun).
- 'Naive DTW Visit/Weekend': Combines predictions based on vessel visit and whether it's a weekend. A visit is defined as a visit if the closest_distance_to_stjernelaks* is less than a fine-tuned threshold.
- 'DTW Grid RFE RF': This is essentially the same as 'Rand RFE RF.' The difference is that it is trained on *Labeled Time Series* dataset with DTW features. The Random forests after both Recursive Feature Elimination and $GridSearchCV()$, meaning that hyperparameters are tuned.

7.2.2 Labeled Time Series Results vs Baseline

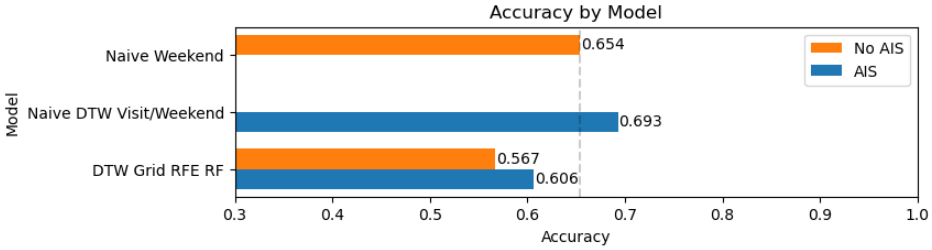


Figure 7.15: *Accuracy* DTW Grid RFE RF vs baseline.

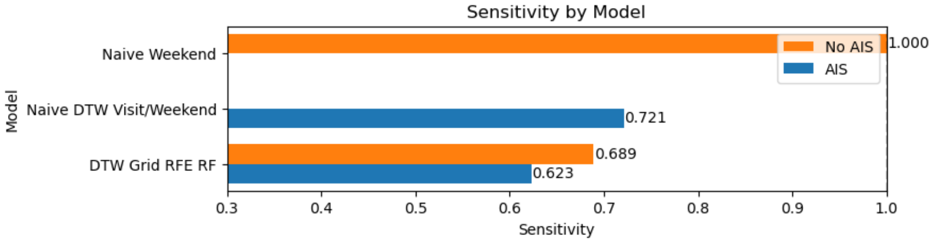


Figure 7.16: *Sensitivity* DTW Grid RFE RF vs baseline

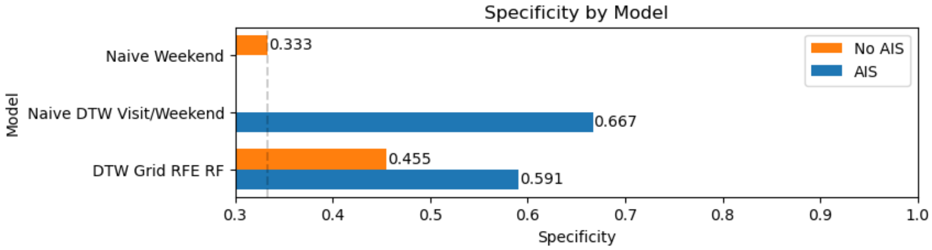


Figure 7.17: *Specificity* DTW Grid RFE RF vs baseline

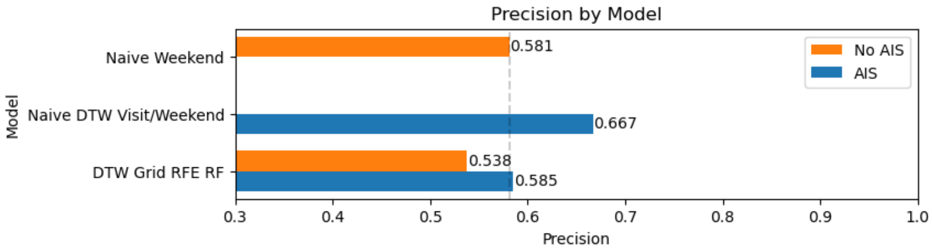


Figure 7.18: *Precision* DTW Grid RFE RF vs baseline

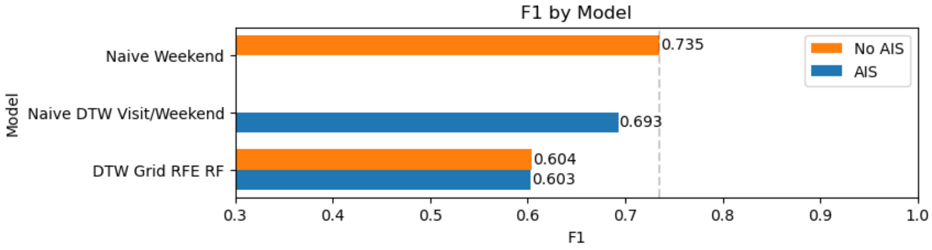


Figure 7.19: *F-measure* DTW Grid RFE RF vs baseline

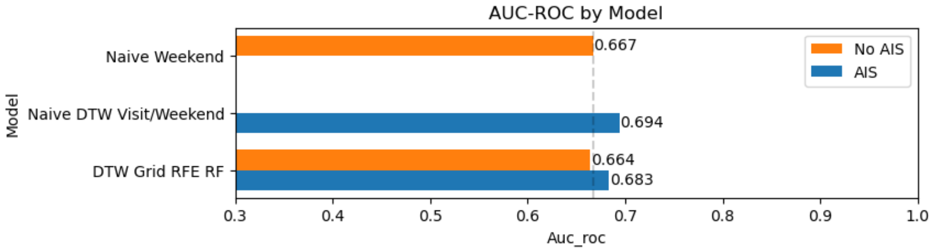


Figure 7.20: *AUC-ROC* DTW Grid RFE RF vs baseline

From the results above, we can observe that the random forests model 'DTW Grid RFE RF' augmented with Recursive Feature Elimination and $GridSearchCV()$ and trained on the AIS inclusive feature set from the *Labeled Time Series* dataset slightly outperforms the *AUC-ROC* for the baseline. However, it did not outperform the 'Naive DTW Visit/Weekend', a far less sophisticated model.

The Feature Importance of DTW Grid RFE RF

The most crucial features determined by their importance scores were 'weekday' and 'day,' with the importance score listed in the table 7.6 below. $GridSearchCV()$ found the best results when using two features.

Feature	Importance
weekday	0.715
day	0.285

Table 7.6: Feature Importance, feature set without AIS.

However, a notable change was observed when AIS data was included in the feature set. The 'weekday' feature decreased in importance while 'day' increased. The 'week' feature was included and deemed important. Still, the most interesting observation is the improvement in *AUC-ROC* score by 2.86% from 0.664 to 0.683 despite not deeming any AIS feature important.

Feature	Importance
day	0.439749
week	0.422107
weekday	0.138144

Table 7.7: Feature Importance, feature set with AIS included.

Chapter 8

Discussion

In this chapter, we delve into the results derived from our analysis of the two datasets: *Labeled Days* and *Labeled Time Series*. Both of these datasets were instrumental in our quest to answer our research question, helping us understand the role and significance of AIS data in predicting fish processing at Grieg Seafood Stjernelaks. Through careful consideration of dataset and model characteristics, we strive to shed light on the intricate dynamics that influence model performance and highlight areas for future exploration.

8.1 Sub-RQ 1: Labeled Days Dataset

Recall the figure 7.13 from the previous Chapter, highlighted again in figure 8.1 below. The figure illustrates the resulting *AUC-ROC* scores for all the visited models compared to the baseline 'Naive Weekday/Weekend'.

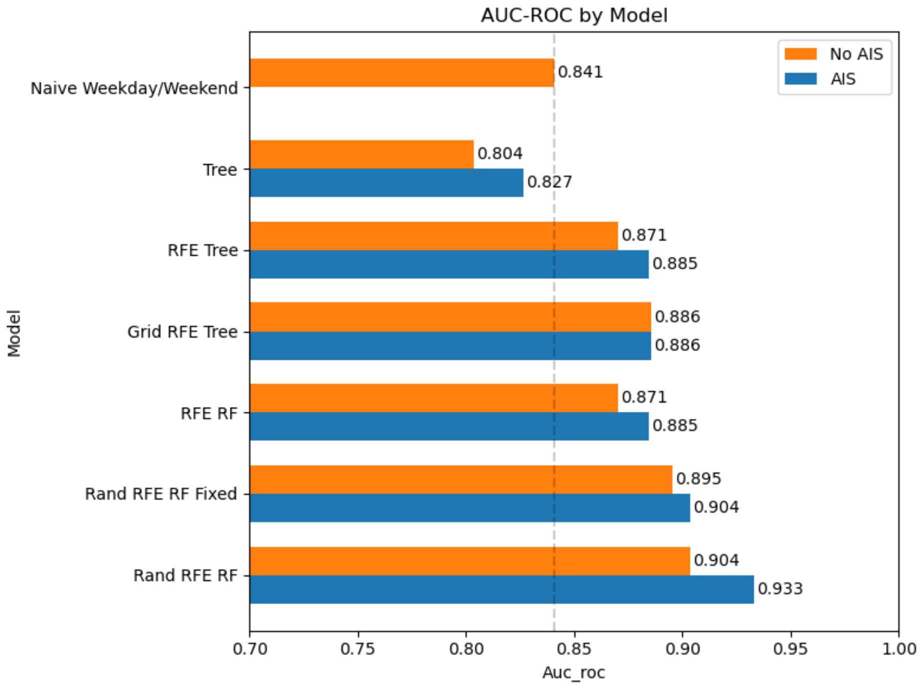


Figure 8.1: *AUC-ROC* by Model versus baseline.

For the models using the *Labeled Days* dataset, we observe a progressive improvement in the *AUC-ROC* score as the applied models become increasingly sophisticated. The simplest model, 'Tree', is the only model that does not beat the baseline's *AUC-ROC* score. Even though it does not beat the baseline, the version that uses the AIS inclusive feature set outperforms the AIS non-inclusive version. The implementation of this model does not use Recursive Feature Elimination. Thus, we know that the visit* feature is not eliminated and contributes towards improving the model's score. The fact that the 'Tree' model is outperformed by the baseline signifies that the baseline is a rather good predictor of Stjernelaks' activity status and that simple models such as 'Tree' becomes ineffective.

8.1.1 RFE Tree

The 'RFE Tree' model that uses Recursive Feature Elimination scored even higher for both instances of feature sets and did outperform the baseline. It scored the highest when trained on the AIS inclusive feature set. However, even though it scored the highest for the AIS inclusive feature set, the RFE did not deem the visit* feature important and it was not included in the selected features.

Feature	Importance
day	1.0

Table 8.1: Feature Importance RFE Tree, AIS non-inclusive feature set.

For the AIS non-inclusive feature set, the default RFE implementation selects the floored square root of available features. $\sqrt{3} \approx 1.732$, so $\lfloor\sqrt{3}\rfloor = 1$. Resulting in an *AUC-ROC* score of `0.871` based on only the 'day' feature.

Feature	Importance
day	0.624954
weekday	0.375046

Table 8.2: Feature Importance RFE Tree, AIS inclusive feature set.

For the AIS inclusive feature set, it was allowed to select $\sqrt{4}$, which is 2. Resulting in an *AUC-ROC* score of `0.885` based on the 'day' and 'weekend' features leading to an increase of `1.6`% in terms of *AUC-ROC* score.

8.1.2 Grid RFE Tree

For the 'Grid RFE Tree,' we observe that the scores are identical when the model is applied to the feature sets. Unlike the 'RFE Tree', the 'Grid RFE Tree' does not use the default RFE hyperparameters but is fixed to select exactly three features (see Section 6.7.3). The model yields identical results for both feature sets because the RFE has eliminated the AIS features from the AIS inclusive feature set, ultimately resulting in two identical feature sets and *AUC-ROC* scores of *0.886*.

Feature	**Importance**
weekday	0.613557
week	0.340415
day	0.040628

Table 8.3: Feature Importance Grid RFE Tree, both AIS inclusive and non-inclusive feature sets.

8.1.3 RFE RF

For the 'RFE RF' model, we make the same observation as for the 'RFE Tree' model. The number of features is set to use the default square root method. Therefore, when the model is applied the AIS non-inclusive feature set is only allowed to select one feature, while the model that is applied to the AIS inclusive feature set is only allowed to select two features. Consequently, the model trained on the AIS inclusive feature set scores better than the model with access to fewer features resulting in an increase to the *AUC-ROC* score of `1.6%` from `0.871` (one feature) to `0.885` (two features).

Feature	Importance
day	1.0

Table 8.4: Feature Importance RFE RF, AIS non-inclusive feature set.

Feature	Importance
day	0.622062
weekday	0.377938

Table 8.5: Feature Importance RFE RF, AIS inclusive feature set.

8.1.4 Rand RFE RF Fixed

Similar to what was the case for 'Grid RFE Tree' above, the 'Rand RFE RF' does not use the default RFE hyperparameters but is fixed to only select three features. However, it do not provide the same scores for both feature sets. When the AIS inclusive feature set was applied the *AUC-ROC* score increased by `1% from 0.895` to `0.904`. Despite this small increase the same features is selected by the RFE with slightly different importances, see tables 8.6 and 8.7 below.

Feature	Importance
weekday	0.601356
week	0.327484
day	0.071160

Table 8.6: Feature Importance, AIS non-inclusive feature set.

Feature	Importance
weekday	0.654537
week	0.304403
day	0.041060

Table 8.7: Feature Importance, AIS inclusive feature set, (*) indicating AIS feature.

The RF model creates slightly different trees in the forest every time it is trained. We can see that this is the case for the two forests using 'Rand RFE RF Fixed' because they are trained on exactly the same features, yet end up getting different feature importances and scores. Given the arbitrary nature of the *AUC-ROC* score change, it could just as likely have been an increase or a decrease, and the absence of any selected AIS feature does not indicate that AIS brings any value.

8.1.5 Rand RFE RF

The results of the best performing model 'Rand RFE RF' has already been extensively described in Section 7.1.2.

This model does not use the default RFE hyperparameters but is allowed to select the optimal number of features between 2, 3, or 4 (see Section 6.8.1). This means that the final random forest model is trained on the number of features that provides the best *AUC-ROC* score. For the AIS non-inclusive feature set the model yields the best *AUC-ROC* score when it only selects two features, see table 8.9 below.

Feature	Importance
day	0.726299
weekday	0.273701

Table 8.8: Feature Importance, AIS non-inclusive feature set.

For the AIS inclusive feature set the model yields the best *AUC-ROC* score when it selects four features, see table 8.7 below.

Feature	Importance
weekday	0.719927
week	0.174250
visit*	0.090585
day	0.015237

Table 8.9: Feature Importance, AIS non-inclusive feature set.

This is the first time we see the selection of the AIS feature visit*. Why do we see this now and not for the other models? Due to the arbitrary nature of the trees selecting different features every time it is trained, visit* has been observed as a selected feature in other training runs than what is presented in the thesis results. The random forests model handles the arbitrary nature by applying the 'wisdom of the crowd', using the mean of all the created trees. From all the trees in the forest, some of the trees select the AIS feature visit*, and some do not.

In total, we see that the AIS feature visit* actually has a meaningful impact on the results because it is not eliminated by the 'Rand RFE RF' model.

The importance of visit* is not very high. Which might be the reason for it to only be selected when the model is allowed to select the optimal number of features compared to the 'Rand RFE RF Fixed' which is fixed to selecting three features.

When comparing model performance using the AIS inclusive and non-inclusive feature set we see the biggest increase in *AUC-ROC* score. Additionally we see the highest *AUC-ROC* score when visit* is included, supporting the hypothesis that AIS data add value to the models predictive performance.

8.2 Sub-RQ 2: Labeled Time Series Dataset

Recall the figure 7.20 from the previous Chapter (Results), highlighted again in figure 8.2 below. The figure illustrates the resulting *AUC-ROC* scores for all the visited models to the baseline: 'Naive Weekend'.

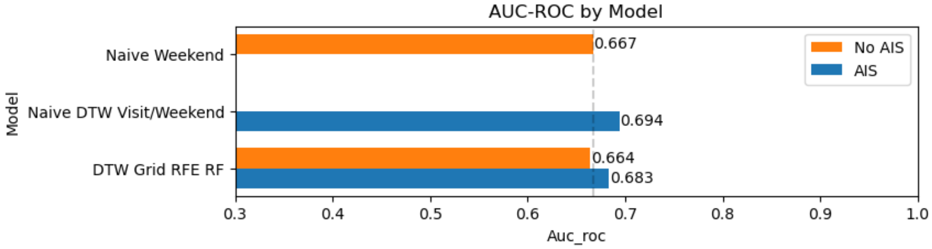


Figure 8.2: *AUC-ROC* DTW Grid RFE RF vs baseline

8.2.1 Labeled Time Series Baseline: Naive method

The 'Naive Weekend' method, serving as the baseline for the *Labeled Time Series* dataset, provides some interesting results that are worth taking a closer look at. This naive simplistic model, which predicts $Direct$ = True for all weekdays and $Direct$ = False for all weekends, yields an *accuracy* of 0.654, meaning that it correctly predicts 65.4% of the outcomes in the dataset. This highlights a basic but important pattern in the data: the tendency for the $Direct$ label to be True on weekdays and False on weekends.
The method has a *sensitivity* (true positive rate) of 1.000, meaning that the method perfectly identified all True instances of the $Direct$ label in the dataset. This means there exists no $Direct$ = True on the weekends. On the other hand, with the *specificity* of 0.333, the method only correctly identifies 33.3% of the instances where the $Direct$ label = False. This means that the method has trouble distinguishing days when fish are not directly received from a vessel, resulting in false positives.

8.2.2 Naive DTW Visit/Weekend

The highest *AUC-ROC* score is observed in figure 8.2 for the 'Naive DTW Visit/Weekend' method, outperforming the baseline 'Naive Weekend' method which was the best naive method we were able to create. When this was initially observed, it gave an indication that the use of AIS features could provide some predictive value also when predicting the $Direct$ label. However, this is not a very comprehensive method which might miss the complex nature and patterns in the data. It is therefore expected that a more comprehensive model such as the 'DTW Grid RFE RF' should outperform both the baseline and the 'Naive DTW Visit/Weekend'. This was not the case and is further discussed later in this chapter.

This model yields the highest *AUC-ROC* score while applying the AIS inclusive feature set. It outperforms the more complex model 'DTW Grid RFE RF', implying that AIS provides some predictive value. However, this *AUC-ROC* score of `0.694` is below what is considered *acceptable* (*AUC-ROC*<0.7) highlighting the uncertainty related to this result.

8.2.3 DTW Grid RFE RF

This model does not use the default RFE hyperparameters but is allowed to select the optimal number of features between 2, 3 or 4 (see Section 6.8.1). This means that the final random forest model is trained on the number of features that provides the best *AUC-ROC* score. For both the AIS inclusive and non-inclusive feature set the model yields the best *AUC-ROC* score when it only selects three features. The same features are selected but their importance differs between the sets, see tables 8.10 and 8.11 below.

Feature	Importance
weekday	0.589941
day	0.205575
week	0.204484

Table 8.10: Feature Importance, AIS non-inclusive feature set.

Feature	Importance
day	0.454243
week	0.438085
weekday	0.107672

Table 8.11: Feature Importance, AIS inclusive feature set.

The inclusion of AIS improved the model performance in terms of *AUC-ROC* score by 2.86% from 0.664 to *0.683*. In contrast to the similar model applied to the *Labeled Days* dataset, this model did not choose the *Labeled Time Series* dataset's AIS feature direct_distance_sum_norm*. Despite the improved *AUC-ROC* score, this is a random result and it does not support the hypothesis that AIS data adds value to the model's predictive power. To understand the randomness in the results we should compare the *AUC-ROC* score to the other methods.

The 'DTW Grid RFE RF' model applied to the AIS non-inclusive feature set is outperformed by the baseline 'Naive Weekend'. We also observe that when the 'DTW Grid RFE RF' model is applied to the AIS inclusive feature set the model gets outperformed by the 'Naive DTW Visit/Weekend' method.

This is an unexpected result because random forests are generally a robust and efficient model capable of capturing complex patterns and interactions between features in the data. The random forest model's high variance property makes it a good fit for large and complex datasets, and it typically excels over simpler models when the relationship between the predictors and the label is non-linear and intricate. However, this is not observed in our results. Given the relatively small size of our dataset, the 'DTW Grid RFE RF' model may have overfitted the training data, explaining the randomness we observe.

8.3 Limitations

In examining the thesis research question, we identified two key limitations that have influenced our inability to arrive at a convincing conclusion. These two limitations are related to Kystdatahuset's API endpoints and insufficient label data.

8.3.1 Kystdatahuset's API Endpoints

A few things undermine the quality of the AIS data. We discovered that the Kystdatahuset's API endpoints referred to in Chapter 5, for some reason, do not provide all the tracks it's supposed to. We discovered this when explicitly looking at a single vessel's movement through Kystdatahuset's website, where we could not find its corresponding tracks in the response from the **Kystdatahuset's API 1: POST /api/tracks/within-area** endpoint. The extent of the missing tracks is unknown, but we know at least 41 is missing. However, if we successfully retrieve all the missing tracks, the models that employ AIS inclusive feature sets are expected only to improve. This is due to the fact that for the missing tracks, all AIS features that the ML models consume contain missing data.

Labeled Days Implications

Currently, with all the missing tracks, the models are incorrectly taught to believe that for some $Active$ days in the *Labeled Days* dataset, visit* equals False. An implication of enriching our *Labeled Days* dataset with more instances where visit* equals True, is an expected enhancement of our model's discriminative ability, which is measured by the *AUC-ROC* score. Since *AUC-ROC* represents the model's capacity to correctly classify $Active$ and $Inactive$ states at various threshold settings. If these additional instances lead to more accurate predictions, they could potentially enhance the *AUC-ROC* score, signifying an improvement in the model's predictive capacity. Consequently, highlighting that the AIS feature visit*, adds value to the model's predictive performance.

The current best-performing model, the 'Rand RFE RF,' has an *AUC-ROC* score of `0.933`. We believe that with more visit* = True instances in the *Labeled Days* dataset, our model's capacity to correctly distinguish between $Active$ and $Inactive$ states, could potentially be improved further, leading to an even higher *AUC-ROC* score.

Labeled Time Series Implications

For the *Labeled Time Series* dataset, retrieving the missing tracks would lead to an increased number of time series samples. We will get one time series from the same day and vessel as the retrieved track. This time series will likely be labeled $Direct$ = True, since it would be describing a relevant vessel within the geofencing of Stjernelaks. Additionally, we will get corresponding time series for the day before, and the day after for the same vessel. The label for these time series is harder to hypothesise whether it should be $Direct$ = True or False because they could either be inside or outside of the geofencing.

Since the *Labeled Time Series* dataset already is quite small, only containing about 422 time series, retrieving the missing tracks would lead to a significant increase to the number of time series. After some investigation, we know that we are missing at least 41 tracks. Retrieving these leads to `41 * 3 = 123` `(+31.5%)` additional time series. 3 = number of time series we fetch for each track (day before, current day, and day after).

An increase in the amount of data would likely lead to better model performance. This is especially true for time series, where the availability of more data points allows for a better understanding of trends, seasonalities, and other temporal dynamics. With a larger sample size, the importance of the AIS feature direct_distance_sum_norm* in predicting the $Direct$ label might become more apparent.
Additionally, when we have more data, the *AUC-ROC* will be calculated based on a larger set of instances, making it more reliable and less susceptible to random variations in the small *Labeled Time Series* dataset and hence reduce the uncertainty related to the *AUC-ROC* score.

8.3.2 Insufficient Label Data

Recall the figure 5.4 from Chapter 5, highlighted again in figure 8.3 below. This figure describes the periods we collect labels from in the dataset. When addressing a research question as complicated as ours, it requires an equally complex dataset to ensure accurate predictions. This figure illustrates the fact that for both the labels $Active$ and especially $Direct$, the data is not complex enough for our models to adapt to all the complexities inherent in our research problem's domain.

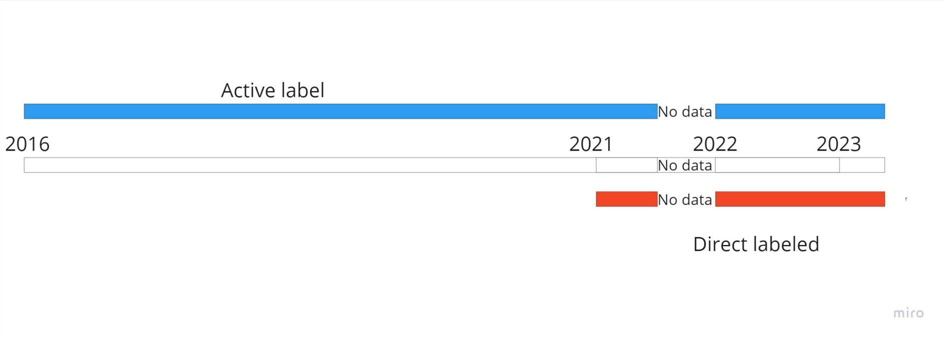


Figure 8.3: Time periods where we can extract $Active$ and $Direct$ labeled data, resulting in 2007 $Active$ labels, and 547 $Direct$ labels.

The shortfall in the label data is potentially hampering the models performance, by restricting their ability to accurately discern the patterns and correlations within the Stjernelaks fish processing dataset.

However, we observe a promising result from the analysis performed on the *Labeled Days* dataset that uses the $Active$ label, with an *AUC-ROC* score of *0.933* which is considered as *excellent* (*AUC-ROC*>0.9). We interpret this result as a confirmation that the AIS data provides extra value to the model's predictive performance.

To reach a definite conclusion regarding if the AIS does provide value to the model's predictive performance, we must also look at the results from the *Labeled Time Series* dataset to see if the results coincide. From the *Labeled Days* results the best performing model was the 'Rand RFE RF'. When we observe the equivalent result for that model for the *Labeled Time Series* dataset, the *AUC-ROC* score is *0.633*. This is less than what is considered *acceptable* (*AUC-ROC*<0.7). Consequently meaning that the we are unable to arrive at a definite conclusion that AIS data adds value. However, it is likely, as we have argued above that the limited $Direct$ labels may be the cause of this.

Future research would greatly benefit from obtaining a larger dataset, with a greater representation of both $Active$ and $Direct$ labels. This would allow for a more comprehensive understanding of the research question space, facilitating the development of other more sophisticated models that can more accurately capture its complexity and thereby provide more reliable predictions.

Chapter 9

Conclusion

9.1 Can AIS Data be used to Predict Fish Processing at Grieg Seafood Stjernelaks?

This research aimed to identify if AIS data could be used to predict fish processing at Grieg Seafood Stjernelaks.

In order to reach a profound answer to the research question, it was further split into two sub-questions:
Sub-RQ 1: Can AIS data be used to predict if Stjernelaks is processing fish on any given day, regardless of the source being a waiting cage or direct vessel delivery?
Sub-RQ 2: Can AIS data be specifically used to predict of Stjernelaks is processing fish that has been directly delivered by a vessel on any given day?

The rationale was to ensure a comprehensive exploration of the utility of AIS data in predicting fish processing at Stjernelaks - both in general and specific contexts - effectively covering the full extent of the research question.

Regarding **Sub-RQ 1**, by analyzing the *Labeled Days* dataset and the $Active$ label, we found promising results that indicated that AIS data could indeed be used to predict fish processing on any given day, regardless of the source being a waiting cage or direct vessel delivery. We found that the visit* of a vessel can give valuable information. When a visit* is recorded of a relevant vessel, it will, based on our investigation of model trees, increase the probability of fish processing that day. This is because it will either deliver fish directly to Stjernelaks - guaranteeing that the activity status for Stjernelaks is $Active$, or to the waiting cage.

For **Sub-RQ 2**, we further investigated if a visiting relevant vessel delivers fish directly by analyzing the *Labeled Time Series* dataset and the $Direct$ label. We know that this has a conditional relationship with Stjernelaks' activity status. Being able to predict this using AIS, would further strengthen our capabilities to answer the main research question. However, we were not able to prove this with our analysis. As extensively debated, we believe this is due to the limitations of our label data and the faulty Kystdatahuset's API tracks endpoint.

Thus, based on our inability to establish satisfactory results for **Sub-RQ 2**, we can not convincingly conclude that AIS can be used to predict fish processing at Grieg Seafood Stjernelaks. However, when addressing **Sub-RQ 1**, the results suggest that AIS data can be used to predict if Stjernelaks is processing fish on any given day. Furthermore, our answer to the main research question is that while AIS data shows promise in predicting if Stjernelaks is processing fish on any given day, it does not conclusively prove that AIS can be used to predict fish processing at Grieg Seafood Stjernelaks.

9.2 Further Work

Building upon the findings of this thesis, future research could potentially explore the following areas:

- Acquiring more label data from multiple processing facilities.
  - This would improve predictive effectiveness and robustness.
- Gather AIS data from multiple sources to cross-examine the results to improve the integrity of the data.
  - This would improve predictive effectiveness and robustness.
- Utilize other data types such as fish sickness reports and holidays.
  - This would give the models a chance to learn new patterns possibly improving the predictive effectiveness and robustness.

- If the points above have been addressed. Look into other closely related research questions, such as predicting precise volumes of processed fish, or look into predictions elsewhere in the life cycle of the Atlantic salmon, such as the transporting of smolt to fish farms.
  - This could provide value towards ensuring the sustainable and responsible management of the Norwegian fish farming industry.

- This thesis did not consider the time aspect of a vessel's path. This could be done by imputing the time series, making all time series have the same frequency. For example a point for every minute. These points could then be clustered to describe which areas the vessels spend shorter or longer periods of time.
  - Can be used to make better-performing models.

9.3 Contributions to the Applications of AIS Data Within the Norwegian Fishing Farm Industry

This research consists of a thorough investigation of how AIS may be used to describe a part of the Norwegian fish farming industry, specifically the fish processing facility Grieg Seafood Stjernelaks. Other lessons can be learned from the findings in this thesis.

Firstly, the results clearly state that predictions using temporal patterns to predict the activity status for Stjernelaks perform *excellently* concerning *AUC-ROC* scores.
Secondly, we have created a generic method of collecting AIS data, that can be followed by others when investigating research questions that require AIS data.
Thirdly, as addressed in the literature review in Chapter 3. There is little existing research into the advanced applications of AIS data. Thus, this thesis itself can be considered as pioneering within the application of AIS toward describing the Norwegian fish farming industry.
Lastly, we have processed and structured the Stjernelaks fish processing data making it consumable for ML models. Other ML models, analysis, and business intelligence may be built around our proposed data structure. This data structure could possibly be expanded to be utilized by all fish processing facilities within the Grieg Seafood corporation.

Bibliography

[Azdy and Darnis, 2019] Azdy, R. A. and Darnis, F. (2019). Use of haversine formula in finding distance between temporary shelter and waste end processing sites. In *Journal of Physics: Conference Series*, volume 1500 of *3rd Forum in Research, Science, and Technology (FIRST 2019) International Conference*, page 012104. IOP Publishing Ltd.

[Barchard and Pace, 2011] Barchard, K. and Pace, L. (2011). Preventing human error: The impact of data entry methods on data accuracy and statistical results. *Computers in Human Behavior*, 27:1834–1839.

[BarentsWatch, 2023] BarentsWatch (2023). BarentsWatch. Accessed: 28-05-2023.

[Blue Planet, 2023a] Blue Planet (2023a). Blue Planet. Accessed: 28-05-2023.

[Blue Planet, 2023b] Blue Planet (2023b). Meetings and emails. Personal communication with [Blue Planet], 2023.

[Box et al., 2015] Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

[Brownlee, 2019] Brownlee, J. (2019). *Master Machine Learning Algorithms*. Machine Learning Mastery Pty. Ltd.

[Cawley and Talbot, 2010] Cawley, G. C. and Talbot, N. L. (2010). On overfitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(Jul):2079–2107.

[Domingos, 1999] Domingos, P. (1999). The role of occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery*, 3.

[Efron, 1979] Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26.

[Fawcett, 2006] Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

[Fiskeridirektoratet, 2023] Fiskeridirektoratet (2023). Tildelingsprosessen. `https://www.fiskeridir.no/Akvakultur/Tildeling-og-tillatelser/Tildelingsprosessen`. Accessed: 01-03-2023.

[Gareth James, 2013] Gareth James, Daniela Witten, T. H. R. T. (2013). *An Introduction to Statistical Learning*, volume 112. Springer, New York.

[GeoJSON, 2023] GeoJSON (2023). Geojson. `https://geojson.org/`. Accessed: 28-05-2023.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

[Grieg Seafood ASA, 2023] Grieg Seafood ASA (2023). Grieg seafood. `https://griegseafood.com/`. Accessed: 28-05-2023.

[Guyon et al., 2002] Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422.

[Harmon, 2011] Harmon, F. (2011). Life cycle of atlantic salmon for new sea grant poster. `http://harmon-murals.blogspot.com/2011/01/life-cycle-of-atlantic-salmon-for-new.html`. Accessed: 28-05-2023.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

[Hyndman and Athanasopoulos, 2018] Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: Principles and Practice*. OTexts.

[IBM, 2023a] IBM (2023a). What is a decision tree? `https://www.ibm.com/topics/decision-trees`.

[IBM, 2023b] IBM (2023b). What is random forest?

[Investopedia, 2023] Investopedia (2023). Variance inflation factor (vif). Accessed: 11-06-2023.

[Jia et al., 2017] Jia, H., Daae Lampe, O., Šoltészová, V., and Strandenes, S. P. (2017). Norwegian port connectivity and its policy implications. https://openaccess.nhh.no/nhh-xmlui/handle/11250/2489492.

[Jia et al., 2019] Jia, H., Prakash, V., and Smith, T. (2019). Estimating vessel payloads in bulk shipping using ais data. *International Journal of Shipping and Transport Logistics*, 11(1):25–40.

[Jupyter Development Team, 2023] Jupyter Development Team (2001 - 2023). Project jupyter.

[Kendig, 2015] Kendig, C. E. (2015). What is proof of concept research, and how does it generate epistemic and ethical categories for future scientific practice?

[Keogh et al., 2001] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3(3):263–286.

[Kjerstad, 2022] Kjerstad, N. (2022). Ais. https://snl.no/AIS.

[Kruskall and Liberman, 1983] Kruskall, J. and Liberman, M. (1983). The symmetric time warping problem: From continuous to discrete. In *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*, pages 125–161. Addison-Wesley Publishing Co., Reading, Massachusetts.

[Kuhn and Johnson, 2013] Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*. Springer.

[Kystdatahuset, 2023] Kystdatahuset (2023). Kystdatahuset webservices. https://kystdatahuset.no/webservices/swagger/ui/index. Accessed: 28-05-2023.

[Kystverket, 2023] Kystverket (2023). Kystdatahuset. Accessed: 14-06-2023.

[Lerøy Seafood Group, 2023] Lerøy Seafood Group (2023). Lerøy seafood. https://www.leroyseafood.com/no/?gclid=CjwKCAjw67ajBhAVEiwA2g_jELfgReUscHIiU-iWT7pUVuA6uo-_293rXzSe6sT2UW8SV0G3CD2B4RoCrvoQAvD_BwE. Accessed: 28-05-2023.

[Lin et al., 2007] Lin, J., Keogh, E., Lonardi, S., and Chiu, B. (2007). Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144.

[MarineTraffic, 2023] MarineTraffic (2023). Ronja polaris. Accessed: 28-05-2023.

[Mavridis and Moustaki, 2008] Mavridis, D. and Moustaki, I. (2008). Detecting outliers in factor analysis using the forward search algorithm. *Multivariate Behavioral Research*, 43(3):453–475. PMID: 26741205.

[Meling, 2021] Meling, J. (2021). Grieg Seafood Rogaland AS Avd Stjernelaks - Høyringsbrev. `https://www.statsforvalteren.no/contentassets/9e0bdeb08da442979b5e22a09c15bec1/grieg-seafood-rogaland-as-avd-stjernelaks---hoyringsbrev-040621-stavanger.pdf`. Accessed: 23-05-2023.

[Misund, 2023] Misund, B. (2023). Fiskeoppdrett. `https://snl.no/fiskeoppdrett`. Accessed: 04-03-2023.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

[Mowi, 2023] Mowi (2023). Mowi. `https://mowi.com/`. Accessed: 28-05-2023.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Peter Bull, 2023] Peter Bull, Isaac Slavitt, e. a. (2023). Cookiecutter data science – a logical, reasonably standardized, but flexible project structure for doing data science projects. Accessed: 25-05-2023.

[Project, 2023] Project, F. (2023). fastdtw: Approximate dynamic time warping (dtw) algorithm with an o(n) complexity. `https://pypi.org/project/fastdtw/`. Accessed: 01-05-2023.

[Python Software Foundation, 2023] Python Software Foundation (2023). Python.org. Accessed: 28-05-2023.

[Raschka and Mirjalili, 2017] Raschka, S. and Mirjalili, V. (2017). *Python Machine Learning*. Packt Publishing Ltd.

[Regjeringen.no, 2022] Regjeringen.no (2022). Fargeleggingen i trafikklyssystemet i havbruk er klar. https://www.regjeringen.no/no/aktuelt/fargelegging-i-trafikklyssystemet-i-havbruk/id2917698/.

[Roar Os et al., 2017] Roar Os et al., (2017). Are ais-based trade volume estimates reliable? the case of crude oil exports. https://openaccess.nhh.no/nhh-xmlui/handle/11250/2492199.

[Robert M., 2007] Robert M., O. (2007). A caution regarding rules of thumb for variance inflation factors. https://www.researchgate.net/publication/226005307_A_Caution_Regarding_Rules_of_Thumb_for_Variance_Inflation_Factors.

[SalMar ASA, 2023] SalMar ASA (2023). Salmar. https://www.salmar.no/. Accessed: 28-05-2023.

[Scikit-Learn Developers, 2023] Scikit-Learn Developers (2023). Decisiontreeclassifier. Accessed: 28-05-2023.

[Scikit-learn Developers, 2023a] Scikit-learn Developers (2023a). Gridsearchcv: Exhaustive search over specified parameter values for an estimator. Accessed: 28-05-2023.

[Scikit-learn Developers, 2023b] Scikit-learn Developers (2023b). sklearn.ensemble.RandomForestClassifier - Scikit-learn. Accessed: 28-05-2023.

[SSB, 2023] SSB (2023). Statistisk sentralbyrå - aquaculture. Accessed: 09-06-2023.

[Statsforvalteren, 2022] Statsforvalteren, R. (2022). Tillatelse til virksomhet etter forurensningsloven - grieg seafood rogaland as avd stjernelaks. Accessed: 23-05-2023.

[Stein W. et al., 2018] Stein W. et al., W. (2018). The value of foresight in the drybulk freight market. https://openaccess.nhh.no/nhh-xmlui/handle/11250/2482578.

[Tableau, 2023] Tableau (2023). What is data cleaning? - Tableau. Accessed: 28-05-2023.

[TensorFlow, 2023] TensorFlow (2023). Tensorflow. https://www.tensorflow.org/. Accessed: 28-05-2023.

[Towers, 2010] Towers, L. (2010). How to farm common carp. https://thefishsite.com/articles/cultured-aquatic-species-common-carp. Accessed: 04-03-2023.

[tslearn Contributors, 2023] tslearn Contributors (2023). tslearn documentation: Dynamic time warping. https://tslearn.readthedocs.io/en/stable/user_guide/dtw.html#dtw.

[Tukey et al., 1977] Tukey, J. W. et al. (1977). *Exploratory data analysis*, volume 2. Reading, MA.

[United Nations Statistics Division, 2023] United Nations Statistics Division (2023). Overview of ais dataset. https://unstats.un.org/wiki/display/AIS/Overview+of+AIS+dataset. Accessed: 28-05-2023.

[UnixTimestamp, 2023] UnixTimestamp (2023). Unix Time Stamp - Epoch Converter. Accessed: 28-05-2023.

[Yamane, 1967] Yamane, T. (1967). *Statistics: An Introductory Analysis*. Harper and Row.

[Yang, 2019] Yang (2019). How big data enriches maritime research - a critical review of automatic identification system (ais) data applications. https://www.tandfonline.com/doi/full/10.1080/01441647.2019.1649315?casa_token=vInhq1I4kbsAAAAA%3AR3B7GExyxnffFibm_T1vPggYykP8b78jCp0mbwevov9rs0ZIcdOlHIsbv3bdlVXkmn_5UTQHmDUD3w. Accessed: 01-03-2023.

[Zheng and Casari, 2018] Zheng, A. and Casari, A. (2018). *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media.

List of Figures

List of Tables