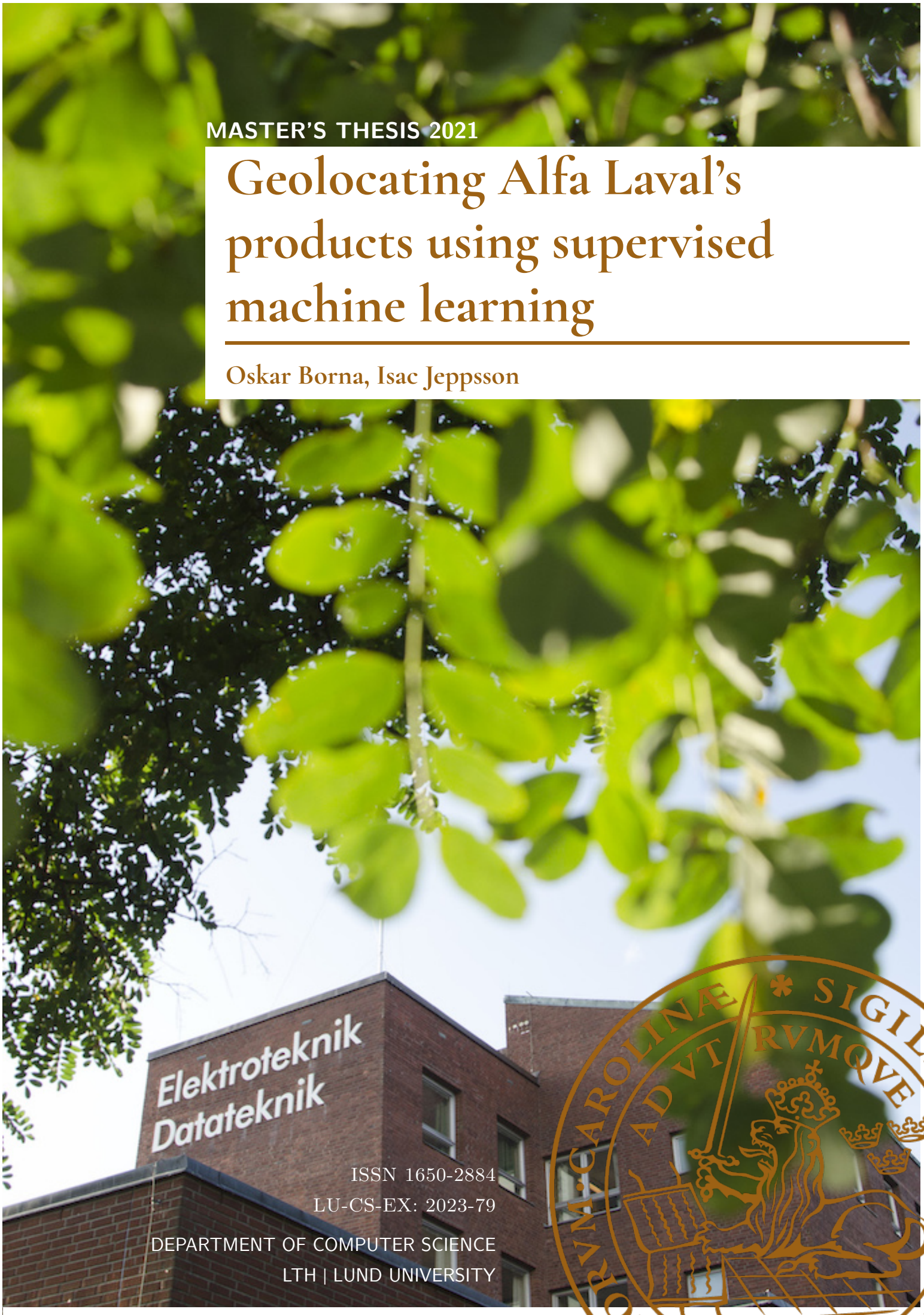


MASTER'S THESIS 2021

Geolocating Alfa Laval's products using supervised machine learning

Oskar Borna, Isac Jeppsson

ISSN 1650-2884
LU-CS-EX: 2023-79

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2023-79

Geolocating Alfa Laval's products using supervised machine learning

Geografisk lokalisering av Alfa Lavals produkter med hjälp av maskininlärning

Oskar Borna, Isac Jeppsson

Geolocating Alfa Laval's products using supervised machine learning

Oskar Borna
os3048bo-s@student.lu.se

Isac Jeppsson
is0435je-s@student.lu.se

June 11, 2021

Master's thesis work carried out at Alfa Laval.

Supervisors: Martin Gunnäng, martin.gunnang@alfalaval.com
Pierre Nugues, pierre.nugues@cs.lth.se

Examiner: Jacek Malec, jacek.malec@cs.lth.se

Abstract

A lot of companies have data that can be used to develop a more successful business. To become more data-driven, it is important to extract valuable information from the raw data. One of the largest challenges for companies, while trying to make this transition, is to ensure a data quality at a high level. In this thesis, we worked with Alfa Laval's database of previously sold products. The main issue with this database was the lack of existing locations, where the products have been installed. In this thesis, we report a solution for the hierarchical prediction of geolocation on three levels: country, city, and coordinates. To build a solution, we examined the three tasks using four different supervised machine learning algorithms. Given our prior knowledge and the available attributes in the database, most tasks proved to yield surprisingly good results. The prediction of countries and cities globally achieved an accuracy of 71% and 57%, respectively. *Random forests* was the overall best performing algorithm for these two tasks. The prediction of coordinates for the United States was a harder task, resulting in a mean error distance of 872 km, which was achieved by an implementation of *artificial neural networks*. Our results showed that a prediction of country and city in fact was an achievable goal, even if the existing input did not have an obvious connection to a location. On the other hand, predicting coordinates did not give a result with a sufficiently small margin of error to be useful for most applications.

Keywords: *k*-nearest neighbors, artificial neural network, random forests, LightGBM, geolocation

Acknowledgements

We would like to firstly thank Alfa Laval and Martin Gunnäng for giving us the opportunity to perform a very exciting and favorable master's thesis in collaboration with them. We want to thank Martin for his help and guidance which has inspired and aligned us along the project.

Furthermore, we would also like to thank Pierre Nugues for his exceptional expertise and devotion throughout the process, always keeping us on the right track.

Contents

Chapter 1
Introduction

1.1 Background

For many years, companies all around the world have been collecting data regarding their business. Whether the data is customer information, sales records or some other type of data, it can be used for multiple purposes.

In more recent years, the search for ways to become more data-driven has become more and more relevant. In the highlight of successful tech companies that thrives on their data, the search for new ways to make earnings using data-driven decisions has become a necessity for many industries. By not embracing and applying new techniques for usage of data, many companies are in danger of losing a possible competitive advantage. The Swedish company Alfa Laval is no way different from these companies.

Alfa Laval was founded in 1883 by Gustaf de Laval and is a leading provider of industrial equipment within heat transfer, separation, and fluid handling on a global scale. The company can be found all over the world and has its headquarters in Lund, Sweden.

Alfa Laval is trying to become more data-driven in all the aspects possible given their data. Unfortunately, in the process of implementing more usage of their data, Alfa Laval has been faced with the issue of deficient data quality. In order for companies to be truly data-driven, their data must be of a certain quality to be valuable. This is the goal of this master's thesis, where we aimed at solving this problem.

The deficient data quality results in that Alfa Laval can not draw substantiated conclusions from the available data. The reason is that a large proportion of the data can not be accounted for due to faulty, or completely missing data. To improve their data quality, Alfa Laval could perform a number of different operations. For example, they could investigate the reason why the faulty data is entered in the database in the first place and set up counteracting methods. This would certainly help improving the data for future entries. Another approach would be to appoint employees to manually update the values that have been entered incorrectly. This approach would help improve the historical data. Unfortunately, this

is only applicable in some rare cases, where the faulty data is obvious. The process would also be very tiresome and expensive for Alfa Laval. There exist countless of other ways of improving the data quality, of which Alfa Laval practice several of them already, however there is still plenty to do.

The solution that we propose is to use different machine learning algorithms that could predict missing or faulty data. In contrast to the process of manually updating the database, this solution would be more efficient and cost effective for Alfa Laval.

1.2 Thesis objectives

Alfa Laval sells industrial equipment of different types, as well as services on these equipment. We will refer to sales of serviced products as *service sales* in this thesis. In some cases, the product specifications are poorly documented in Alfa Laval's database which prevents the possibilities of further service sales.

One of the main areas, where deficient data quality prevents Alfa Laval's service sales is the lack of exact location of some of the products. If the company does not know the location of a product and neither have contact information to the buyer, it is impossible to proactively sell services on the product. Thus, it would be beneficial if the coordinates, the city, or even the country of the product were known. With this information, the unknown products could be divided into sales regions and the respective regions could perform further investigation to locate the exact location of the product.

The goal of our thesis is to solve this issue which corresponds to different types of location predictions. We divided the location prediction of the products into three parts with the purpose of finding a more precise result:

1. Prediction of the country;

2. Prediction of the city; and finally

3. Prediction of the coordinates.

We will use the same approach to predict both the country and city, where the product is located. This task will be solved with classification, where each unique city or country is treated as a category. In contrast to this, when predicting coordinates, we will use regression, where a continuous value is predicted, in this case the latitude and longitude. All tasks will be solved with the use of a few chosen algorithms.

1.3 Previous work

To the best of our knowledge, the amount of work regarding prediction of geolocation is quite limited. Articles regarding trajectory prediction exist, but we argued that those articles were not relevant enough for this thesis. When predicting the trajectory, you simply take previous geolocations, such as coordinates, into account and predict the next coordinates. One example could be to predict in which direction a storm is moving. In contrast to trajectory prediction, where previous coordinates are used as input of utmost importance, we will,

in this thesis, work with input that does not have an obvious connection to a geolocation. In the following section, we will discuss three different articles that we found relevant.

Cheng et al. (2010), in the article *You are where you Tweet: A content-based approach to geolocating Twitter users*, used the text content of tweets to predict the users location on a city level. The algorithm uses several techniques such as naive Bayes and support vector machines to classify and extract so-called local words, which means words that have a strong correlation to a specific geographical location. The occurrences of local words in tweets are then used to classify the city that a specific tweet originates from. In a similar sense, this thesis will relate to finding a way to connect a specific type of product to its potential location.

Verma et al. (2020), in the article *Prediction of residence country of student towards information, communication and mobile technology for real-time: preliminary results* classified students' country of residence from personal data. The experiment is performed using several different algorithms and feature selection methods. The data set contains information collected through a Google form, asking questions towards students in India and Hungary about their perception of the current state of information, communication and mobile technology in their society. The article in question is relevant to this thesis because it aims to predict a geographical location based on information that is not clearly related to geographical locations. It is also highly related in regards to the methods used. Verma et al. (2020) have chosen to use random forests, *k*-nearest neighbors, artificial neural networks and support vector machines as algorithms to solve the problem.

Gaman et al. (2021), in the article *UnibucKernel: Geolocating Swiss German Jodels Using Ensemble Learning*, identified German dialects, where they used a data set consisting of Swiss German Jodels. Jodel is an application, where you can publish posts locally, such a post in the application is called a jodel. It is similar to a Tweet in Twitter. The task is to predict the geolocation, latitude and longitude, of the different jodels. To solve the task, they have performed double regression, where both latitude and longitude approximations are treated as regression tasks. During the process, they have explored a number of different features, where the authors even constructed handcrafted features. Lastly, they used median error distance to evaluate their result.

To summarize the studied articles, each of them touches one of the three hierarchical geolocation levels studied in this thesis. To assist our thesis, we took inspiration mostly in regards to the algorithms that were chosen in the articles.

With this thesis, we have clearly identified a gap in the available research that we hope to bridge. To the best of our knowledge, no other research regarding all hierarchical levels has been performed on a dataset similar to Alfa Laval's. All specified previous works handle a dataset consisting of human text entries originating from the wanted location. Our approach, on the other hand, handles a dataset with no apparent connection to a geographical location. Instead it aims at finding possible patterns in the world's behavior with regards to purchases of industrial machines. For example, it could be that customers in Lund order products with a specific combination of technical attributes that should be used on a certain market. A pattern could then be that this frequently occur in Lund and nowhere else.

Chapter 2

Dataset

2.1 Introduction

As dataset, we used one of Alfa Laval's databases, the *Installed Base* (IB). IB contains information such as technical data, where manufactured and sold products are installed, together with information about the customer who purchased the product. IB consists of 187 different tables. Such a number was intractable for our computing machines. Thus we had to make a specific selection to reduce it.

2.1.1 Scope

Alfa Laval sells a range of different products on a global scale. For this thesis, we considered the *gasketed plate-and-frame heat exchanger* (GPHE), simply for the reason that approximately 53% of the products in IB are GPHEs, making it the most common product.

For the coordinate prediction task, we chose to narrow down the scope by investigating only the products sold in the United States. The prediction of countries and cities were performed on a global scale.

2.1.2 Gasketed plate-and-frame heat exchanger

The gasket plate-and-frame heat exchanger, or the GPHE, can be used in a range of different industries and has the purpose of exchanging heat from one fluid to another without any interference. This is accomplished by making the fluids pass through between every other of the several plates as visualized in Figure 2.1 (Alfa Laval, 2021a).

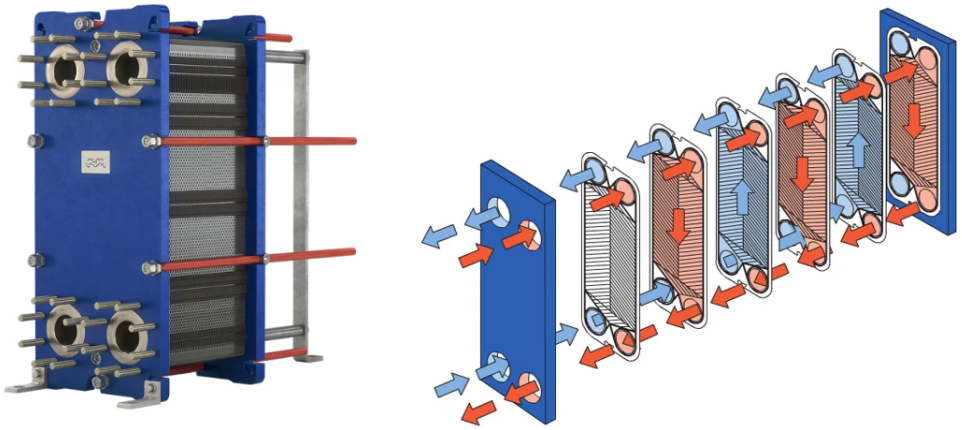


Figure 2.1: Alfa Laval's Gasket plate-and-fram heat exchanger and a visualization of the heat exchange between two fluids (Alfa Laval, 2021b)

2.2 Relevant tables

2.2.1 General data

Within IB, there is a a table called *general data* that consists of one row per sold product. Each row contains information such as the product model, serial number, and several technical attributes. The table has 1,211,337 rows with 220 columns where 55.7% of the cells are missing values. In Figures 2.2, 2.3 and 2.4 we can, in the light blue stacks, see the distribution of the different equipment types, ALSIS application codes and countries for GPHE. ALSIS application code is a financial code, created by Alfa Laval, used to classify the specific market that the product have been sold to.

2.2.2 Customer data

Another table is called *customer data* that contains information about each unique customer who has bought one or several products throughout the years. This information mainly consists of contact information and, most importantly to this thesis, the geographical location of the customer. The location is recorded with a country, a region, a city and an address.

Using the Google API, Alfa Laval has also extracted the geographical coordinates for each address. This enables Alfa Laval to graphically display the customers on a world map. The table has 410,195 rows with 29 columns where 35.2% of the cells are missing values. Figure 2.5 shows all products with existing customer location information in the United States.

2.2.3 Service data

Lastly we used a table called *service data* that consists of a record for all the instances of service that have been performed on the products. Each service contains information about the date of execution, the serial number of the product in question, type of service, and a handful of

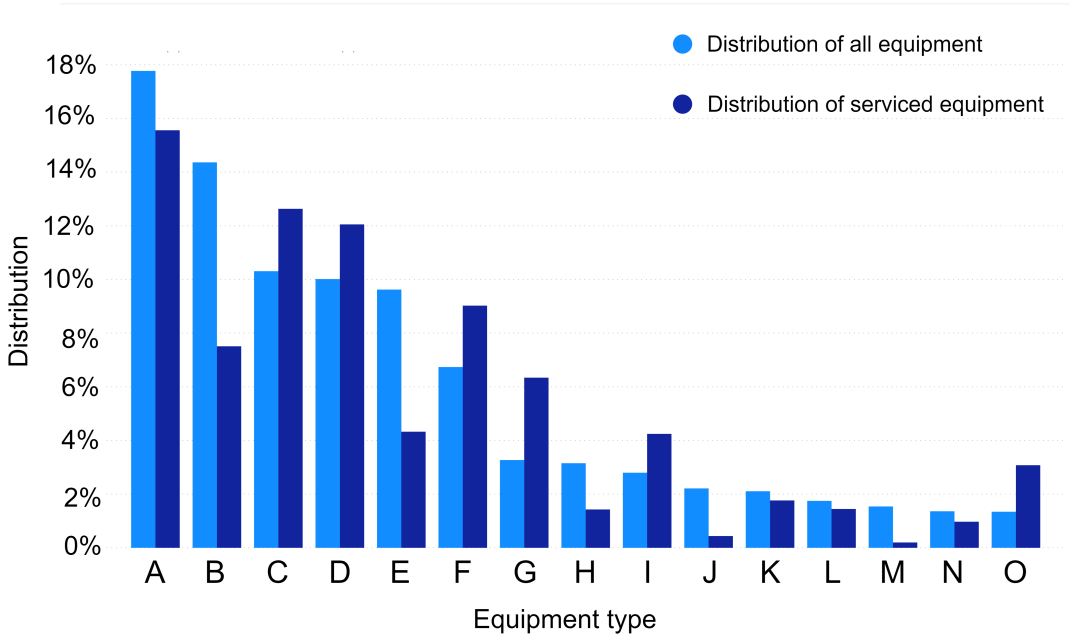


Figure 2.2: Distribution of the number of sold equipment per equipment type compared to the same distribution for serviced equipment in IB.

other attributes. The table has 380,606 rows with 19 columns where 35.7% of the cells are missing values.

In the dark blue stacks in Figures 2.2, 2.3 and 2.4 we can see the distribution of equipment types, ALSIS application codes and countries for the products that have been serviced.

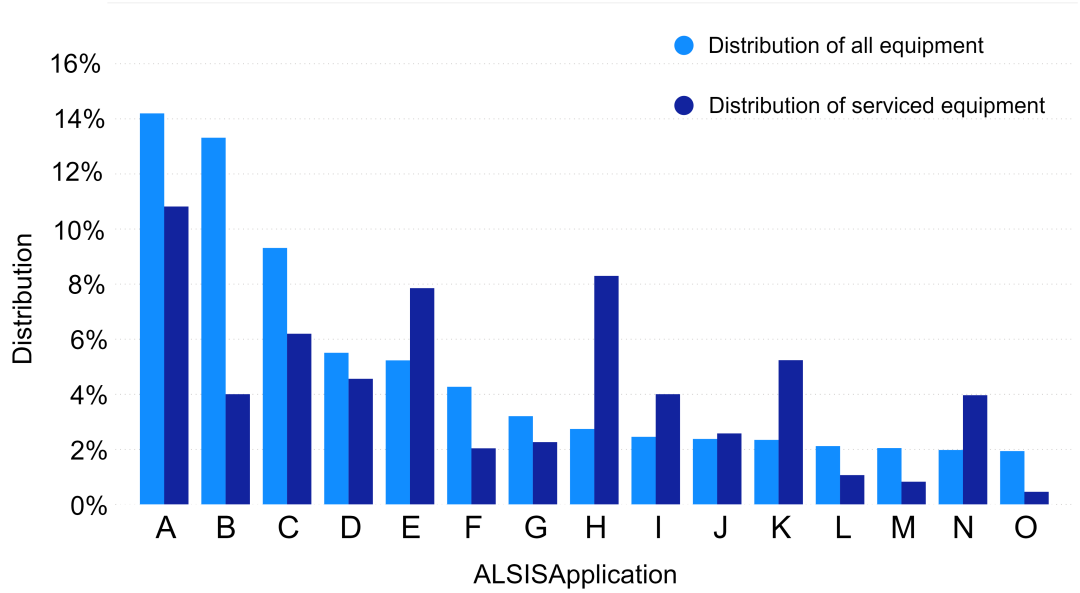


Figure 2.3: Distribution of the number of sold equipment per ALSIS application code compared to the same distribution for serviced equipment in IB.

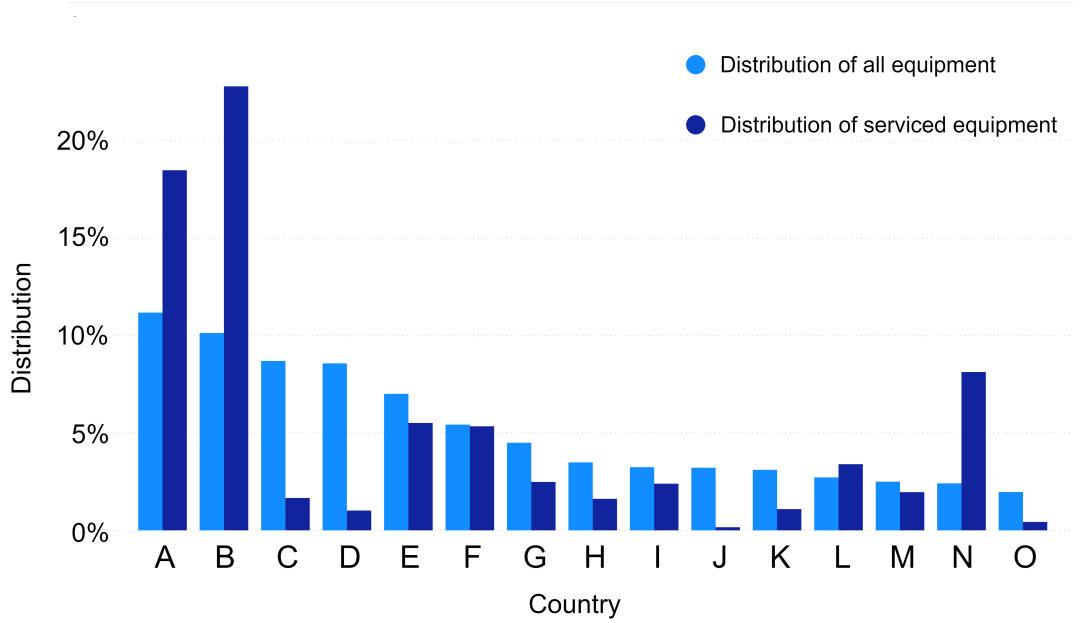


Figure 2.4: Distribution of the number of sold equipment per country compared to the same distribution for serviced equipment in IB.

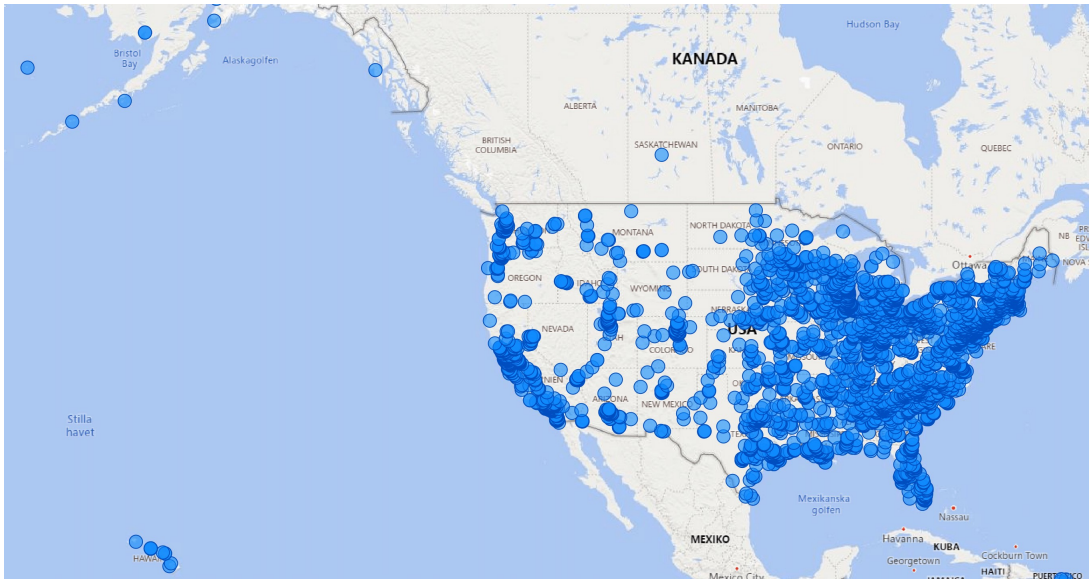


Figure 2.5: All products in the United States with a known position.

Chapter 3

Method

Throughout the project we have followed the *Cross-industry standard process for data mining* (CRISP-DM) (Chapman et al., 2000). CRISP-DM is a framework developed to assist any type of data mining project to provide a more structured and effective approach. The framework is divided into several phases that create the entire life-cycle of the project (Wirth and Hipp, 2000). The phases and how they are connected are displayed in Figure 3.1.

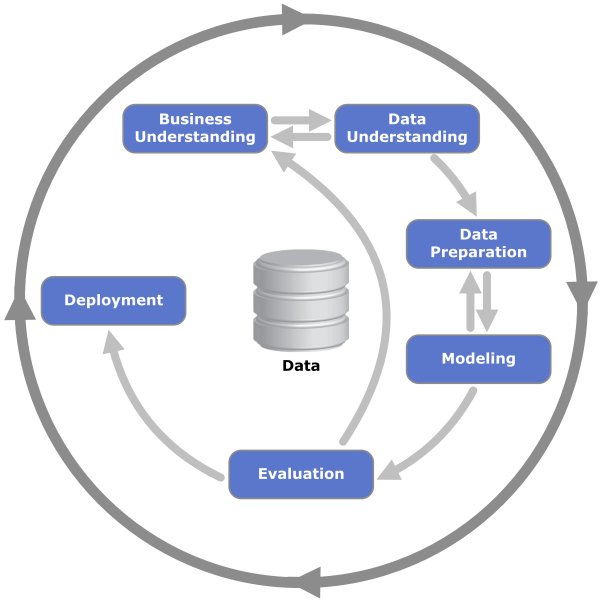


Figure 3.1: The CRSIP-DM life cycle

3.1 Business understanding

To obtain a good business understanding, we had meetings 1-2 times a week with Martin Gunnäng. The purpose of the meetings was to align, validate thoughts and determine the way forward. Additionally, Oskar has been working at Alfa Laval for three years and thus has a general understanding of the company.

3.2 Data understanding

The first step in this project, as when encountering any problem within data science, was to perform an *exploratory data analysis* (EDA) (Tukey, 1977) to some extent. The purpose of performing an EDA was to analyze the data set to discover characteristics, patterns and outliers of the data.

The main goal of the project was to improve the data quality where it contributes the most to Alfa Laval, in terms of service sales. Therefore, we focused the EDA on a comparison between equipment that has and has not been serviced, in combination with the products that could generate the highest sales.

To calculate the service potential, we took multiple attributes into consideration, such as what equipment type, technical attributes, what industry the equipment operated in (ALSIS Application Code) and country of origin. Our findings can be seen in Figures 2.2, 2.3 and 2.4. However we came to the conclusion that even though several trends could be identified, there was not enough data in each category to carry out any successful prediction attempts.

3.3 Data preparation

We had to take a number of factors into consideration when we prepared the data. Firstly we applied all conclusions and focus restrictions from the EDA to the dataset such as correcting spelling errors, deleting outliers and limited our coordinate prediction to the United States.

As mentioned earlier, a fundamental issue for Alfa Laval is the deficient quality of the sales data, more specifically the large percentage of missing data. This problem can be handled in two different ways, imputation or deletion:

- By imputing data, the missing data is replaced by either an approximated or a random value.
- By deleting data, you delete the whole data row for the rows with missing values.

Due to the large amount of data rows that was obtained initially, we concluded that handling the missing values by deletion would not hurt the final model as much as it might do by imputing partly incorrect values. Even though this is a very common practice for many data mining projects, there is a risk of introducing a bias. This highly depends on the reason for the missing data. If the missing values occur randomly throughout the dataset, there is a very low risk of introducing a bias. However, the missing values could for example occur mostly for a specific country or city, which would produce a biased result.

Another large issue with the dataset was the amount of free text entries. For example, several rows contained different formatting of white spaces and dashes making it impossible to categorize those columns in a correct way. To solve this, we performed manual refining, grouping several different values into one and finally removing rows that simply could not be grouped because of their rare occurrences.

Finally, we filtered all numerical columns to remove all potential outliers that could disrupt the model.

3.4 Modeling

3.4.1 Feature selection

In every iteration of the CRISP-DM cycle, feature selection was an important part of the modeling. To ensure a potentially high-prediction accuracy, it was important that the different features that we used were optimized for the actual problem.

The initial goal when performing the feature selection was to find as many valuable columns as possible. Due to deficient data quality of the dataset, the goal was often disrupted by the fact that a large proportion of the data was removed due to the occurrences of empty values. This resulted in that limitations were made, not only based on the information that the column provided, but also the proportion of empty versus filled out values in that specified column.

When we analyzed the data, we decided to initially remove all columns that contained more than 40% empty values. From a total of 220 features, this left us with 92 features. We then removed all columns that contained unique values for every row such as serial number and order number.

Next, we performed a manual selection of the columns. The selection was based on the possibility that the columns could have an impact on the location prediction. We continuously discussed the possible impact of the columns with Alfa Laval to ensure that the decisions that we made were based on reasonable assumptions. However, we only removed columns that certainly did not have the possibility to positively influence the prediction. If an uncertainty occurred, we kept the column and investigated it thoroughly during the ablation study. An ablation study consists of removing and adding features in a systematical fashion to identify the features that potentially improves the score, but equally important, to identify potential features that potentially decreases the score.

Finally, the entire data set was compressed down to 13 features as well as the four targets: longitude, latitude, country and city. Table 3.1 shows the selected features and targets with imaginary example values. The table is flipped to fit in the page, thus rows are columns and vice versa.

3.4.2 Preprocessing

Firstly, we chose to remove all duplicate values in the dataset. We justify this because we believed that the distribution of the two data sets, with and without location, differed. The reason to our beliefs was that customers order multiple products with the same technical attributes and within the same markets. These products would be considered duplicates in

Table 3.1: Selected features and targets with imaginary row values.

	Example row 1	Example row 2	Example row 3
Equipment	M10	M25	T10
ALSIS Application	110	120	114
ALSIS Channel	710	701	710
TotalPlates	15	35	14
SalesPrice	1	2	3
Thickness	0.2	0.5	0.4
CBLength	350	350	250
TempInCold	6	15	22
TempInHot	82	70	100
TempOutCold	63	49	52
TempOutHot	68	55	83
FluidCold	Water	Water	Ethanol
FluidHot	Water	Oil	Water
Latitude	33.744798	55.717209	34.992725
Longitude	-84.408256	37.605203	135.732111
Country	USA	Russia	Japan
City	Atlanta	Moscow	Kyoto

our dataset. We believed that the risk that just one of these products was missing its location was very low when the other products had a location. Thus, any algorithm would probably present a better score than it actually performed when it was used on products where the location was missing.

The dataset consisted of two types of columns: categorical and continuous data. We replaced the categorical data with a one-hot-encoded representation and we normalized the continuous data to zero mean and unit variance. To one-hot-encode categorical data, we replaced each unique category with a numeric representation. If the categorical data consist of two categories, for example *True* or *False* then the numeric representation can either be 0 and 1, or 01 and 10. See Table 3.2 for an example with four colors where three colors are unique.

To normalize data, we first calculated the mean and variance of all values. Then we subtracted every value with the mean and divided with the variance to create a distribution from -1 to 1. The distribution will now have an average mean of zero and variance 1, thus called zero mean and unit variance.

Table 3.2: Example of one-hot-encoding

Color		Red	Blue	Green
Red		1	0	0
Blue	⇒	0	1	0
Green		0	0	1
Red		1	0	0

Lastly, we split the data set into *training set*, *test set* and *validation set*. Firstly we started with dividing the data set in two parts with the proportion 80/20: *split set* and *test set*. Then we

divided *split set* similarly into *training set* and *validation set*, resulting in the following proportions 64/20/16 (train/test/validation). To ensure the same distributions in the three different data sets, i.e. at least one occurrence for each city or country, we applied a filter, before dividing the data sets, that disregarded every category with less than $\frac{1}{0.16} \approx 7$ occurrences.

3.4.3 Training

In this phase of modeling, we calculated a baseline for each problem to obtain a good reference value in how well an algorithm was performing.

To obtain a baseline for prediction of countries, we retrieved the country with most occurrences. This country was then used for all predictions and both accuracy and F_1 score was calculated.

When obtaining the baseline for prediction of cities, we chose not to select the most frequently occurring label as we did for the country prediction. The reason for this was that the country, that the product was located in, was passed on as a feature. Therefore we assumed that any algorithm would at least be able to predict the most frequently occurring city in each country.

Finally, for the coordinate prediction, we calculated the mean value for all the products in the training set and used that as a baseline prediction. This resulted in that every product was classified in the center of all data points. This is for obvious reasons a very bad way of predicting coordinates but a good way to obtain a baseline.

When the baseline was set, we started to train our different algorithms. A thorough explanation of the chosen algorithms will be given in the next chapter. To achieve the best performance, we optimized the hyperparameters of each algorithm, given the prediction result on the validation set, which is called *hyperparameter optimization*. It basically refers to changing parameters for a given algorithm, such as number of iterations, loss function, learning rate, etc to obtain a better score. We then iterated through this process until an optimal score was obtained. When the optimization was finished, we predicted the test set using the models, trained on the split set.

3.5 Evaluation

In the evaluation step of the CRISP-DM model, we performed some assessments of the constructed machine learning models. To evaluate the performance of the models, we used accuracy and F_1 score for classification tasks.

$$\text{accuracy} = \frac{\text{true positive} + \text{true negative}}{\text{total samples}} \tag{3.1}$$

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{3.2}$$

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{3.3}$$

According to Nugues (2014), the F_1-score is obtained through a combination between recall and precision, in fact the harmonic mean of the two performance metrics.

$$F_1\text{-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{3.4}$$

To give an example of the accuracy and the F_1 score, imagine a distribution of 10 products in two different countries. The distribution consist of 9 products from Sweden and 1 from Denmark. If a machine learning algorithm classify every product in Sweden, we would get an accuracy of 90 %. For this case with Sweden in focus we would get a precision of 0.9 and a recall of 1 and thus a F_1 score of $\frac{1.8}{1.9}$. To get the resulting F_1-score, we need to calculate the F_1 score with respect to Denmark as well which in this case is equal to zero. Thus the resulting F_1 score, after computing the mean, is approximately 47.4%.

For all regression tasks, we used mean error distance (MED) and mean squared errror (MSE), seen in Equation 3.5, for evaluation of our results. MED calculates the average distance in kilometers between the true and the predicted coordinates, while MSE on the other hand calculates the average squared difference between the latitude and longitude values. Despite the fact that Gaman et al. (2021) used median squared distance as evaluation metric, we chose to use MED. Generally, using median instead of mean is a good practice when trying to avoid potential large outliers in the data. On the other hand, in our case, there was a risk of achieving good results even if the majority of the predictions are considered as outliers to median. Another reason why we chose to use MED was because coordinates can only assume values in a limited range, making it impossible for huge outliers to occur.

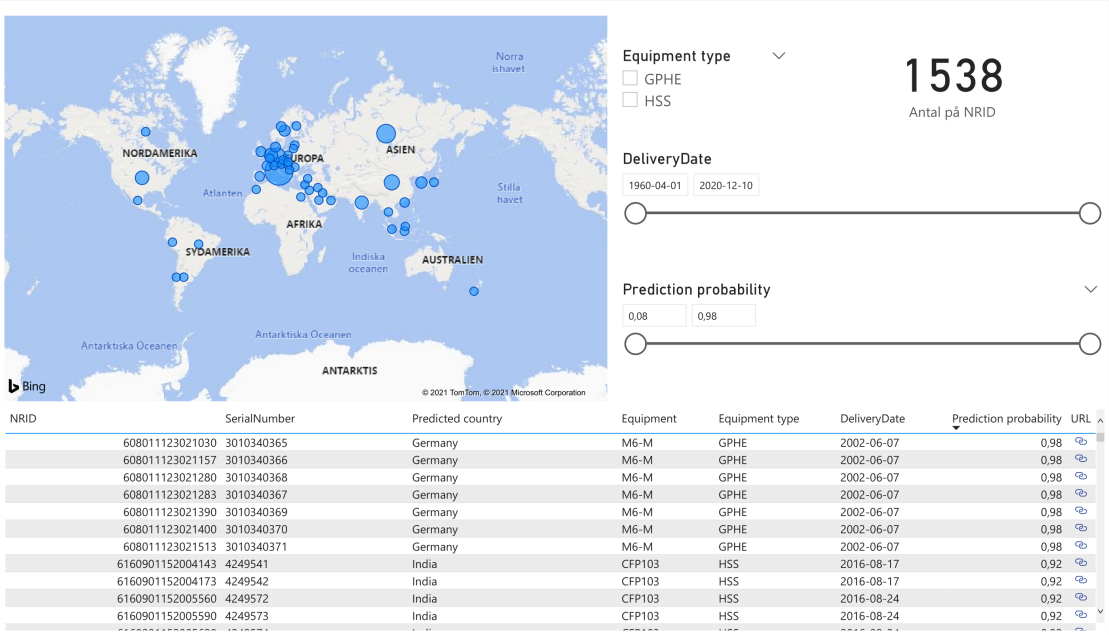
$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_{true} - y_{prediction})^2 \tag{3.5}$$

Based on the different results, we made different conclusions such as consulting with our supervisors, further feature selection, hyperparameter optimization etc. By iterating continuously through the steps of the CRISP-DM model, we got a complete understanding of the problem at hand and could more easily locate the missing factor when we got a poor evaluation result.

3.6 Deployment

Deployment is the last phase of the process in the CRISP-DM framework. In this thesis, the objective in terms of deployment was to construct a Power BI dashboard to visualize our results.

We initially created a map where we could plot the products that had been given a predicted location. Further, we included a table to visualize parts of the technical data. In this table, we included a unique id to enable mapping to the database to retrieve additional attributes that was not accounted in the machine learning model. Beyond a unique id, predicted location and technical data, we also included a probability of how likely each prediction was. This probability was retrieved from the method *predict_proba* used in the RF classification. Lastly, two KPIs and only two filters were visualized since it was possible to apply filters for all attributes used to construct the dashboard. A preview of the main features of the Power BI dashboard can be seen in Figure 3.2.



NRID	SerialNumber	Predicted country	Equipment	Equipment type	DeliveryDate	Prediction probability	URL
608011123021030	3010340365	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021157	3010340366	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021280	3010340368	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021283	3010340367	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021390	3010340369	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021400	3010340370	Germany	M6-M	GPHE	2002-06-07	0,98	
608011123021513	3010340371	Germany	M6-M	GPHE	2002-06-07	0,98	
6160901152004143	4249541	India	CFP103	HSS	2016-08-17	0,92	
6160901152004173	4249542	India	CFP103	HSS	2016-08-17	0,92	
6160901152005560	4249572	India	CFP103	HSS	2016-08-24	0,92	
6160901152005590	4249573	India	CFP103	HSS	2016-08-24	0,92	

Figure 3.2: Power BI report for deployment of predicted counties.

Chapter 4
Algorithms

In the this chapter we will go through the different algorithms that will be used in this thesis. Furthermore, we will introduce the structure of our method and in detail go through every step of that structure.

In this thesis, we used four different algorithms to solve our tasks. Much like Verma et al. (2020), we chose to look at k-nearest neighbors, artificial neural networks and random forests. In addition to these algorithms, we also used an algorithm called LightGBM. We will provide more information about each algorithm in the following sections.

4.1 k-Nearest Neighbors

As an initial experiment, we implemented the simple k-nearest neighbors algorithm (k-NN), created by Fix and Hodges (1985). Since we had little knowledge about the distribution of the used data set, and due to the simplicity of k-NN, we chose to use k-NN for our first experiment. k-NN can be used both for classification and regression, where the main idea of k-NN is to, for a given input, find the nearest, most similar, neighbor in the training set. To find the nearest neighbor, the distance can be calculated using Euclidean distance (Equation 4.1) or Manhattan distance (Equation 4.2).

$$d(x,y)_{Euclidian} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4.1}$$

$$d(x,y)_{Manhattan} = \sum_{i=1}^{n}|x_i - y_i| \tag{4.2}$$

When using a k larger than one, the predicted class is equal to the most frequent true class among the k neighbors found.

As implementation, we used the sklearn.neighbors.KNeighborsClassifier from scikit learn for the prediction of cities and countries. For prediction of coordinates, we used sklearn.neighbors.KNeighborsRegressor. The algorithm contains several hyperparameters that can be tuned for optimal performance.

4.2 Artificial Neural Networks

Artificial neural networks (ANN) are a supervised machine learning algorithms that were first introduced by Mcculloch and Pitts (1943) and originate from neural networks in the human brain. A neural network consists of multiple neurons which receives signals through dendrites that then are forwarded through axons. Similarly, ANNs have nodes that corresponds to the dendrites and edges that corresponds to axons. The nodes are organized in layers, with one layer's output serving as input to the next. Each node is connected to all or a subset of all nodes in the following layer. Each edge in the network has a weight that determines the impact of the signals that are transported from the nodes through that edge.

In this way the input to each node can be calculated by Equation 4.3. If a weight is set to zero the inputs that are passed through that edge will not have an impact at all on the final output. The larger value the weight has, either positive or negative, the more impact that edge has.

The most simple architecture is called a *perceptron*, consisting of only one input layer and one output layer. The output is calculated using an activation function that is applied to the sum of the weighted inputs (Equation 4.3). An activation function is a type of threshold function that determines if a signal should be fired or not depending on the value of the weighted sum.

Most problems cannot be properly solved using a perceptron which leads us to the *multilayer perceptron* (MLP). An MLP is an architecture that consists of layered perceptrons, creating one or several hidden layers between the input and output layer. A visualization of this can be seen in Figure 4.1.

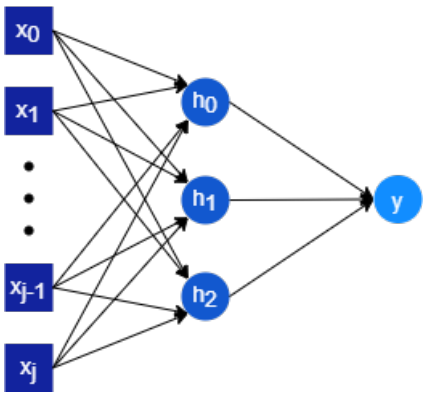


Figure 4.1: Visualization of an arbitrary MLP

When training an ANN, each weight in the network is adjusted over several iterations to give the best final result. This is performed initially with a so called *forward pass* or in other words a prediction of the inputted training data. The error between the prediction and the

labels in the training data is then calculated, and the weights are updated accordingly using gradient descent.

Backpropagation is performed to determine which of the weights that should be used and how they should be updated. Basically, backpropagation means stepping through the network backwards, identifying each connection that resulted in an increased prediction error.

$$y_k = \sum_{j=1} w_{kj} x_j \tag{4.3}$$

As ANN implementation, we used the Tensorflow and Keras libraries. To tackle the tasks at hand with this implementation, different adjustments were made depending on whether the problem was concerning regression or classification. Most importantly, the output activation function was set to linear (Equation 4.4) for the regression task. A linear function basically retrieves an input and sends it forwards as is.

For the classification task, on the other hand, we used the softmax (Equation 4.5) as the output activation function. The softmax function retrieves input without a specific distribution and can thus contain positive and negative values in an undetermined range. When the softmax function is applied, the input is modified and distributed in the interval (0,1). The sum of all the components adds up to 1 and can thus be transformed to probabilities. This way the output can be seen as a list of probabilities of that the input belonging to each class (Géron, 2017).

$$\text{linear}(x) = x \tag{4.4}$$

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{4.5}$$

4.3 Random Forests

The concept of decision trees is the first step required to understand the random forests (RF) learning method. Decision trees are a popular machine learning method that can be used for both regression and classification tasks. Decision trees use a divide-and-conquer approach to construct a tree structure. Figure 4.2 shows a trained decision tree for an arbitrary classification task (Quinlan, 1986). One of the benefits with decision trees is that they are descriptive. If you want to understand why a certain classification was made, such as why a penguin was the outcome in Figure 4.2, you can simply walk through the decision tree backwards to examine what actions that were taken.

Decision trees have proven relatively inaccurate in relation to other methods and a modified model, such as RF, is therefore often used instead. RF is an ensemble method that combines several decision trees and trains them on randomly selected subsets of the original training set. The results are then combined to form the overall result.

In this way, RF can counteract potential overfitting and reduce the variance between results based on different input data (Svetnik et al., 2003). According to Verma et al. (2020), RF outperforms all the other methods evaluated in the article for two out of three cases and achieved the overall highest accuracy score of 92.8% when predicting the country of residence.

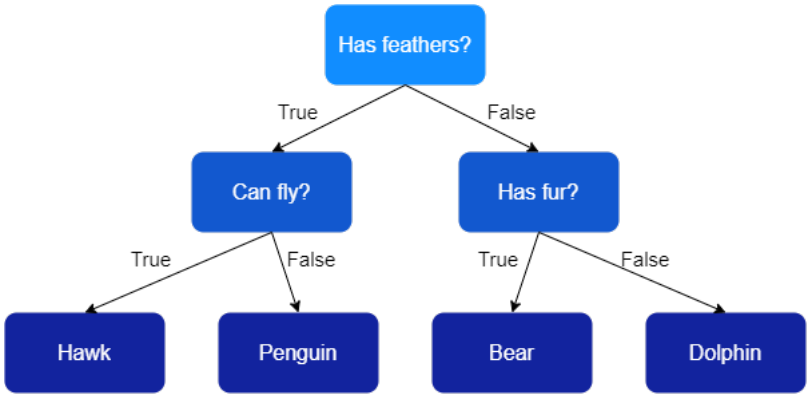


Figure 4.2: Visualization of an imaginary trained decision tree predicting types of animals.

The implementation that we used is the sklearn.ensemble.RandomForestClassifier from scikit learn.

4.4 Light gradient boosting machine (LightGBM)

There are two ways of constructing a model for predicting purposes. Either, one algorithm is used that can solve the task or a combination of weaker algorithms are combined to form an ensemble. An example of an ensemble is the previously mentioned RF method. The most basic and frequently used way of constructing these ensembles is to, after all models have been created and applied to the dataset, calculate the average of all the models results. This has proven to be a rewarding practice historically. In contrast, a boosting algorithm sequentially combines the models, i.e. after each iteration a new model is added to the ensemble based on the current error obtained from the ensemble (Natekin and Knoll, 2013).

In this thesis, we have chosen to use a boosting algorithm called *gradient boosting machines* (GBM) (Friedman, 2001), more specifically, an implementation called LightGBM (Ke et al., 2017). This algorithm is closely related to XGBoost that was used for a regression task in Gaman et al. (2021).

LightGBM is a supervised machine learning algorithm that can be used for both regression and classification tasks and originates from GBM. Since the algorithm is supervised, we need a data set of format $(\mathbf{x}, y)_{i=1}^{N}$ where $\mathbf{x} = (x_1, ..., x_d)$ are the input variables and y is the label that correspond to the input variables. The objective of the algorithm is to find a function $f(\mathbf{x}) = y$ that minimizes a loss function $L(y, \gamma)$ (Equation 4.6) Natekin and Knoll (2013).

$$f(x) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_{true}, y_{predicted}) \tag{4.6}$$

Much like for the other methods, we used LightGBM to solve both of our tasks: regression and multi-classification. For the regression task, while training, we used the L2 Loss (Equation 4.7) as our loss function.

For the multi-classification task we used categorical cross-entropy error, CCEE (Equation 4.8), when we trained the model (Microsoft, 2021).

$$L(y_{true}, y_{predicted})_{L2Loss} = \sum_{i=1}^{N} (y_{true} - y_{predicted})^2 \tag{4.7}$$

$$L(y_{true}, y_{predicted})_{CCEE} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{i} y_{true} \ln(y_{predicted}) \tag{4.8}$$

To set the number of trees used in the training phase, you can change a parameter called `num_iterations`. If the number of trees is set too low, the model will be underfitted and if the number of trees is too high the model will be overfitted. To prevent this from happening, you simply set a high number of trees and use a parameter called `early_stopping_round`, which terminates the training of the model when it has not improved for a given number of iterations.

Chapter 5
Evaluation

5.1 Experimental setup

In this section, we present all relevant settings and hyperparameters used for the final model for each of the algorithms. With the help of this information, our results will be possible to reproduce with the same dataset, with some reservation regarding different default random variables.

5.1.1 Country prediction

The same datasets were used for all models for the prediction of countries. Table 5.1 shows the size of each dataset.

Table 5.1: Sizes of every data set used when predicting countries.

Data set	Size
Train	132,136
Validation	33,034
Test	41,293

After we performed hyperparameter optimization on the validation set, we ended up using the specified parameters shown in Table 5.3. We chose only to list the parameters that we changed from their default values. Thus, the additional parameters that are not listed in the tables assume their default value.

When running the ANN algorithm, beyond the hyperparameters in Table 5.3b, the layer structure used can be seen in Table 5.2.

Table 5.2: Layer structure for ANN used when predicting countries.

Layer type	Activation function	Output size
Dense	relu	100
Dense	relu	25
Dense	softmax	131

Table 5.3: Optimized hyperparameters for all algorithms when predicting countries.

(a) *k*-NN

Parameter	Value
n_neighbors	1
weights	'uniform'
algorithm	'auto'
leaf_size	15
p	1
metric	'minkowski'
n_jobs	None

(b) ANN

Parameter	Value
batch_size	32
epochs	100
optimizer	nadam
loss	categorical_crossentropy

(c) RF

Parameter	Value
n_estimators	500
max_features	sqrt
min_impurity_split	0.01
bootstrap	False

(d) LightGBM

Parameter	Value
objective	multiclass
num_class	131
num_leaves	512
learning_rate	0.03
boosting_type	dart
feature_fraction	0.6
num_iterations	600

5.1.2 City prediction

For the prediction of cities, Table 5.4 shows the size of each dataset.

Table 5.4: Sizes of every data set used when predicting cities.

Data set	Size
Train	80 032
Validation	20 008
Test	25 011

The hyperparameters used for each algorithm, obtained by hyperparameter optimization, can be found in Table 5.6. The layer structure used when running the ANN algorithm is shown in Table 5.5.

Table 5.5: Layer structure for ANN used when predicting cities.

Layer type	Activation function	Output size
Dense	relu	100
Dense	relu	100
Dense	softmax	3001

Table 5.6: Optimized hyperparameters for all algorithms when predicting cities.

(a) *k*-NN

Parameter	Value
n_neighbors	1
p	1

(b) ANN

Parameter	Value
batch_size	16
epochs	10
optimizer	nadam
loss	categorical_crossentropy

(c) RF

Parameter	Value
n_estimators	100
max_features	sqrt
oob_score	True

(d) LightGBM

Parameter	Value
objective	multiclass
num_leaves	200
max_depth	15
learning_rate	0.05
feature_fraction	0.6
max_bin	50
num_iterations	450

5.1.3 Coordinate prediction

Finally, Table 5.7 shows the size of each dataset, used for coordinate prediction.

Table 5.7: Sizes of every data set used when predicting coordinates.

Data set	Size
Train	4948
Validation	1237
Test	1547

The optimized hyperparameters for each algorithm can be found in Table 5.9 and the layer structure for the ANN algorithm is shown in Table 5.8.

Table 5.8: Layer structure for neural network used when predicting coordinates.

Layer type	Activation function	Output size
Dense	relu	500
Dense	relu	300
Dense	relu	200
Dense	linear	2

Table 5.9: Optimized hyperparameters for all algorithms when predicting coordinates.

(a) *k*-NN

Parameter	Value
n_neighbors	19
p	1
leaf_size	15

(b) ANN

Parameter	Value
batch_size	10
epochs	50
optimizer	nadam
loss	mae

(c) RF

Parameter	Value
n_estimators	500
max_features	sqrt
min_impurity_split	0.01
bootstrap	False
num_iterations	3000

(d) LightGBM

Parameter	Value
num_leaves	700
learning_rate	0.002
feature_fraction	0.6
metric	l1
max_bin	200
num_rounds	3000

5.2 Results

In the following sections, we will present the results for the different regression and classifications problems that we have encountered. The result will be presented for the prediction of countries, cities and coordinates. The same algorithms have been used for all problems.

5.2.1 Country prediction

Table 5.10 shows the algorithm we used, the number of predicted countries, accuracy and F_1 score. There is a total of 131 predictable countries. The boldface numbers show the highest number for that column.

Table 5.10: Number of predicted countries, accuracy and F_1 score for all methods when predicting country.

Method	# predicted countries	Accuracy	F_1-score
Baseline	1	0.122	0.002
k-NN	129	0.627	0.294
ANN	25	0.233	0.024
RF	119	0.700	**0.341**
LightGBM	**131**	**0.709**	0.285

Figure 5.1 shows results for ANN where we see the training and validation accuracy in the left figure. We clearly see that the training accuracy, dotted line, constantly improves throughout the process. However, we see that it converges around 100 epochs or slightly thereafter. The validating accuracy improves in the beginning and follows the training accuracy. However it quite quickly falls of and converges slightly over 23% with small changes. In the right figure we can instead see the training and validation loss. Similarly to the left figure, we can see that the training loss converges around 100 epochs. The validation loss quickly assumes its lowest value and then increases.

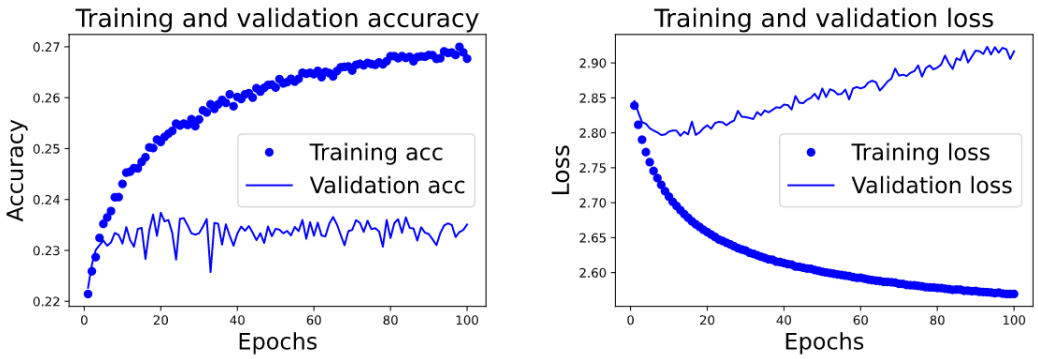


Figure 5.1: Graphs over training and validation loss and accuracy during training of the ANN for country prediction over 100 epochs.

Figure 5.2 shows a confusion matrix from predictions with the LightGBM algorithm. Below the confusion matrix, we see five red dots with the purpose of highlighting the five most occurring countries. We clearly see that the corresponding vertical lines are more distinctive than other vertical lines in the confusion matrix.

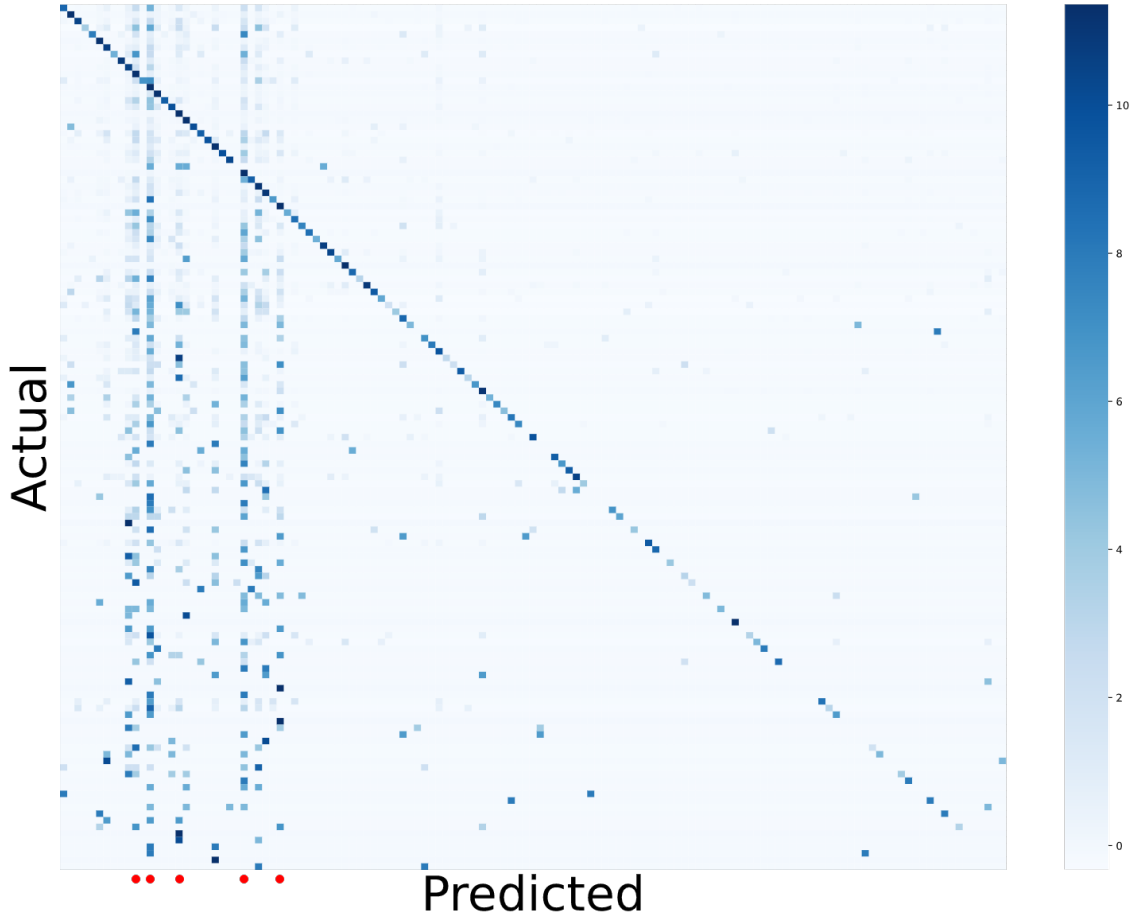


Figure 5.2: Confusion matrix over all predicted countries using the LightGBM algorithm. Below the vertical lines, we marked the five most frequent countries with red dots.

5.2.2 City prediction

In the same way as for countries, Table 5.11 shows what algorithm that was used, number of predicted cities, accuracy and F_1 score. There was a total of 3001 predictable cities. The boldface numbers show the highest number for that column.

Table 5.11: Number of predicted cities, accuracy and F_1 score for all methods when predicting cities.

Method	# predicted cities	Accuracy	F_1-score
Baseline	101	0.192	0.019
k-NN	**2789**	0.486	0.381
ANN	192	0.080	0.010
RF	2709	**0.571**	**0.447**
LightGBM	2490	0.534	0.360

Figure 5.3 shows results for ANN prediction of cities. In the left figure we see both train-

ing and validation accuracy. Here we clearly see that the training accuracy continuously increases and converges around 50 epochs. The validation accuracy decreases and also converges. In the right figure, we see both training and validation loss. The right figure is similar to the left one but with the opposite data set, training and validation.

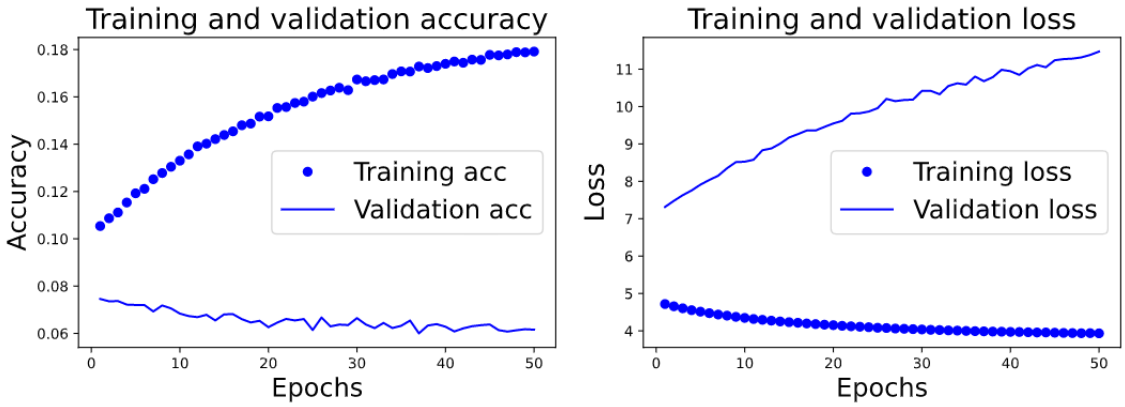


Figure 5.3: Graphs over training and validation loss and accuracy during training of the ANN for city prediction over 50 epochs.

Figure 5.4 illustrates a three-dimensional confusion matrix where the underlying confusion matrix contains values for every city. The cities in the confusion matrix were sorted according to what country they belong to. Thus, one could see squares along the diagonal showing all cities for a specific country. The confusion matrix is then partly highlighted. The highlighted part only contains values from cities in Russia.

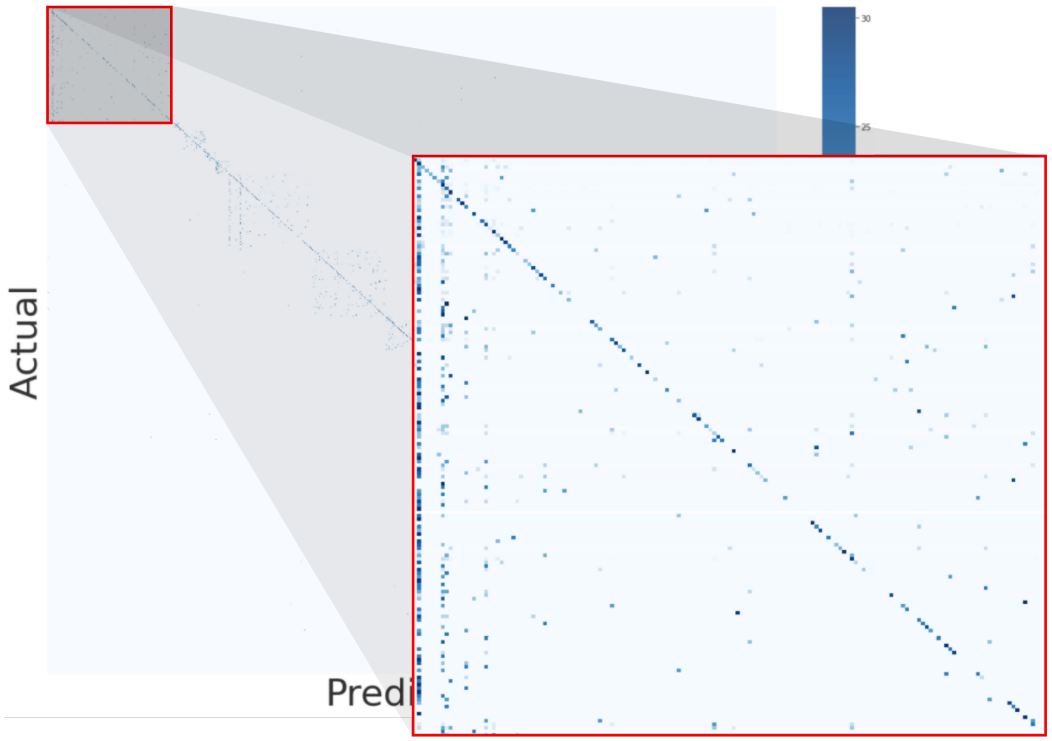


Figure 5.4: A zoomed in section of the confusion matrix of predicted cities, using RF, showing only the cities located in Russia.

In Figure 5.5, we see a similar figure as 5.4 but for the United States. It follows the same principles but highlights the cities located in the United States instead of in Russia. The underlying confusion matrix is the same in both 5.5 and 5.4.

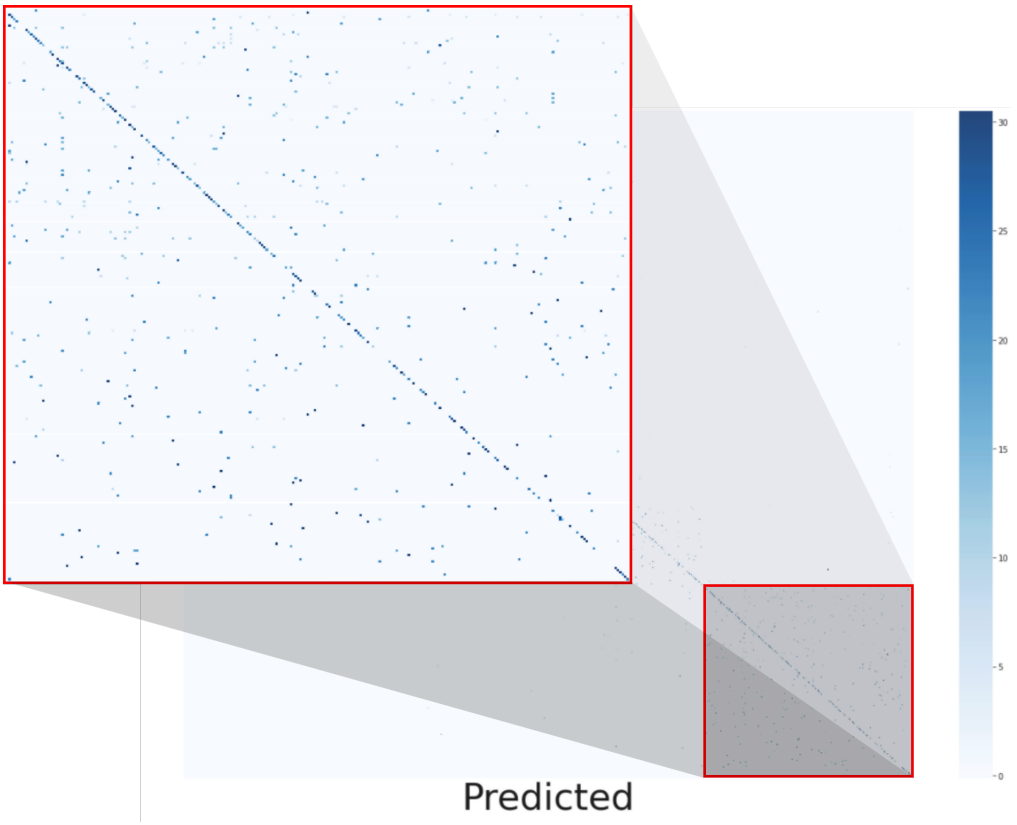


Figure 5.5: A zoomed in section of the confusion matrix of predicted cities, using RF, showing only the cities located in the United States.

5.2.3 Coordinate prediction

Table 5.12 shows what algorithm that was used together with the different error functions. In contrast to above tables, Tables 5.10 and 5.11, the boldface numbers are the lowest numbers in the respective column.

Table 5.12: MSE and MED in kilometers for all methods when predicting coordinates.

Method	MSE	MED (km)
Baseline	134.16	1230.76
k-NN	118.92	1091.73
ANN	**105.83**	**872.01**
RF	107.25	992.73
LightGBM	108.79	1027

In Figure 5.6, we can see the result for ANN prediction of coordinates. We see training loss in contrast to the validation loss. The reason that we do not have a graph for training and

validation accuracy is that coordinate prediction is a regression task and accuracy can not be calculated. We clearly see that the training loss continuously decreases but the validation accuracy converges early around a loss of 7.

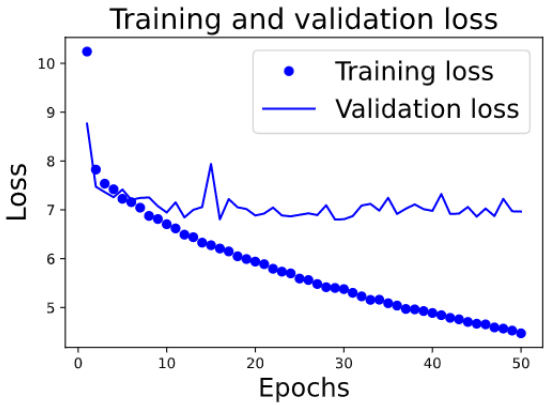


Figure 5.6: Graph over training and validation loss and accuracy during training of the ANN for coordinate prediction over 50 epochs.

In Figure 5.7, we can see all labels from the test set. By labels, we are referring to the correct coordinates that the predictions relate to. In Figure 5.8, the prediction for each algorithm is presented. We clearly see that every algorithm originate at the same spot but more or less drawn out horizontally.

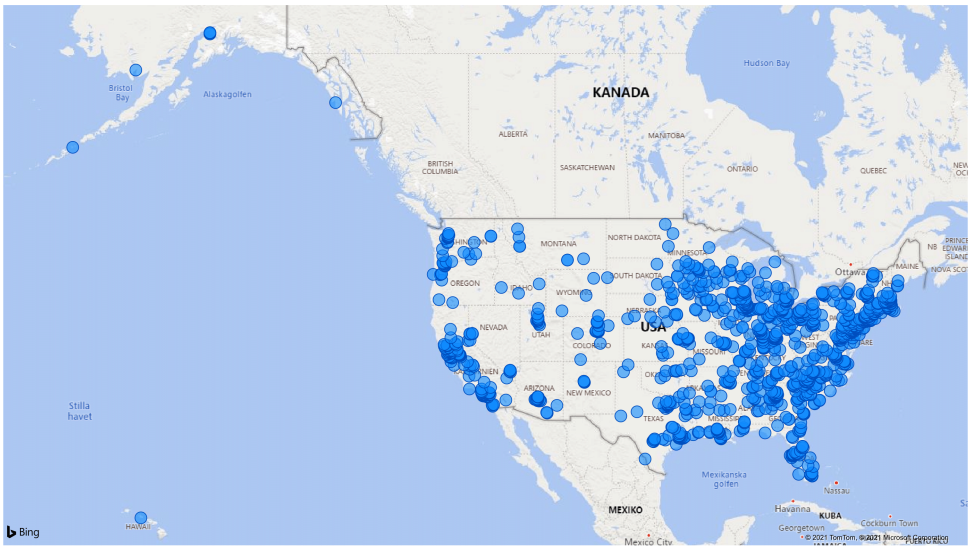
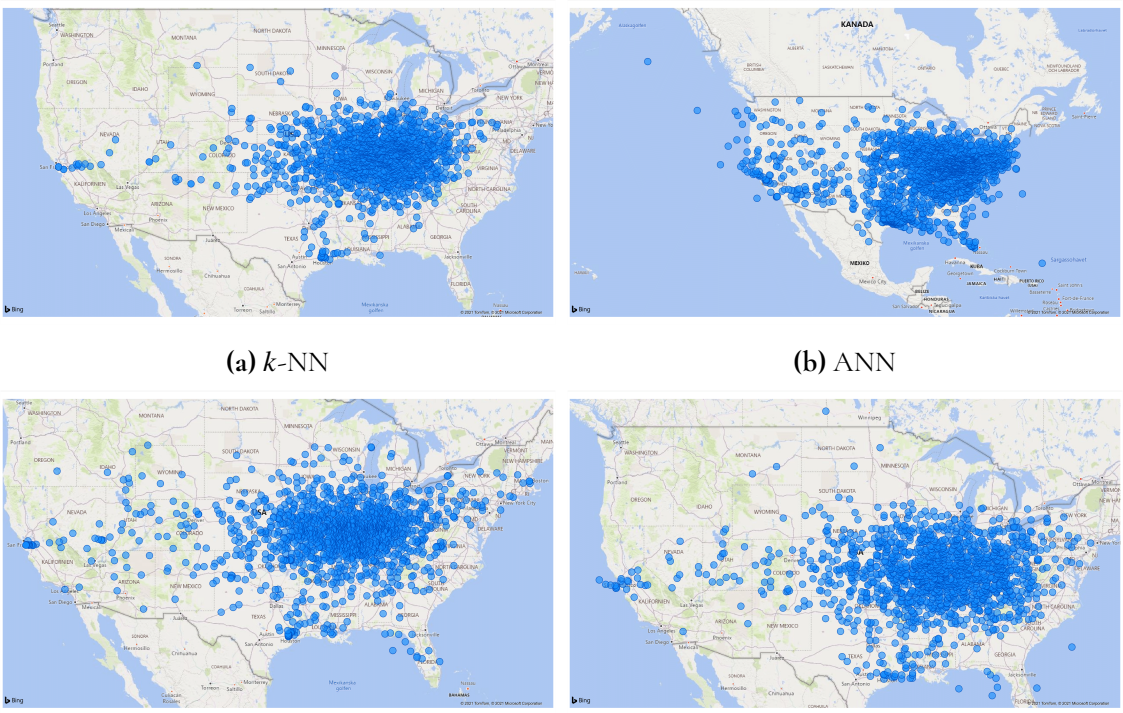


Figure 5.7: All labels in the test set.



(a) *k*-NN **(b)** ANN

(c) RF **(d)** LightGBM

Figure 5.8: Predicted coordinates from the test set using all algorithms.

Chapter 6
Discussion and Conclusion

6.1 Discussion

One of the goals with this master's thesis was to answer the question whether is is possible to predict location of products or not. To answer this question, we obtained a simple baseline and quickly realized that our algorithms performed significantly better. It was surprising to both us and Alfa Laval that we were able to predict the location with an accuracy as high as we got, given the type of data and the data quality that we had.

6.1.1 Countries

In Table 5.10, we can see that our baseline obtained an accuracy of 12%. This directly correlates to that Russia is the most frequent country with 12% of the products. Further, when studying Table 5.10, we can see that ANN performed better than our baseline but was still very bad. Generally when you are using ANN, you want much data, which we argue was the scenario in our case, but perhaps we did not have enough data or that the given data was too narrowed. As can be seen in Figure 5.1, the model seemed to get overfitted, almost immediately, in respect to the validation loss but the accuracy remained more or less the same. We noticed that the accuracy did not improve much depending on any of the hyperparameters and did thereby assume that the ANN could not solve our task with the data set at hand.

The next algorithm we tried was k-NN. k-NN generated a surprisingly good accuracy in relation to its simplicity. The resulting F_1 score was significantly better than the F_1 score for both the baseline and ANN. However, the F_1 score was somewhat disappointing in comparison with the accuracy.

When we tried RF, we realized that without hyperparameter optimization we obtained a good result. We predicted less countries than we managed to predict with k-NN. However, the F_1 score was higher for RF than for k-NN which implies that k-NN predicts more countries, but the countries were predicted incorrectly. The number of predicted countries is not

a good measure of how well the algorithm was performing. It was used as a validation to ensure that the algorithm did not predict only a few countries and that way obtained a good accuracy. The F_1 score served the same purpose and generated a similar kind of validation.

Lastly we used LightGBM to predict the countries. We obtained the best accuracy with LightGBM of 70.9%, about one percentage point better than RF. LightGBM managed to predict all countries and thus a logical adoption would be that the F_1 score also was high. However this was not the case, the F_1 score was in fact lower than both of the previously tried algorithms.

Looking at the confusion matrix, Figure 5.2, we can see that the majority of the countries was partly predicted as the five biggest countries, marked with red dots. The middle red point in the confusion matrix represents Russia which was the country with the most occurrences, as mentioned previously. Studying the vertical lines the least prominent was Russia which indicates that the algorithm was fairly good at predicting Russia compared to the other four countries with many occurrences.

6.1.2 Cities

When studying Table 5.11, we can see that when selecting the most frequent city for each country as the prediction, the baseline achieved a score of approximately 19% and a F_1-score of 1.9%. In comparison with our highest scoring algorithms, we clearly see that they perform significantly better with a score of 57.1% in accuracy and a F_1 score of 44.7%.

What we can see is also that the F_1 score is much closer to the achieved accuracy for all methods, in comparison to country prediction, which is very promising. The reason for this could be that the distribution of cities across the data set was more even. In comparison to the country prediction, it did not exist a few classes containing the majority of all data points to the same extent.

Just like for the country prediction, the k-NN algorithm produced a decent result but was outperformed by RF and LightGBM once more in regards to both accuracy and F_1 score. Although, k-NN manages to achieve the best score when it comes to number of predicted cities.

Finally, we can see that this was the only experiment, where any of the algorithms performed worse than the baseline, which in this case was the ANN. As discussed for the country prediction, an important requisite for most types of problems solved with ANN is to have a data set of extensive size. This could be the issue here as well, or possibly that the data simply did not contain the information needed to solve the problem with this method. Much like for the country prediction, we saw signs of clear overfitting when analyzing Figure 5.3. In contrast to the country prediction, in this case, we can even see signs of overfitting for the validation accuracy. Because of this, the chosen number of epochs was kept low as can be seen in Table 5.6b.

Looking at the zoomed in sections of the confusion matrix in Figures 5.4 and 5.5 we can immediately notice large differences. In Figure 5.4, showing the cities that were located in Russia, we can identify at least two very distinct vertical lines. The leftmost and also the most significant line belongs to Moscow, which naturally was the city with the most occurrences among the cities within the data containing product located in Russia.

Comparing this to the submatrix showing the cities in the United States, Figure 5.5, we can see that the United States shows no similar trends. The missclassified products do not

seem to all be predicted as the most frequent city.

The conclusion that we draw from this is that the result differs a lot between different countries, which could depend on several reasons. One reason could be that the products sold to Russia are often very similar to each other. Another reason could be that the sales representatives in charge of filling out the data for the sales to Russia make consistent errors in important columns that other countries, such as the United States, do not. Both reasons could make it very difficult for our algorithms to predict the location properly and thereby gain more in accuracy by predicting the locations as the most frequent city. Obviously, trends such as the one for Russia can be identified for several other countries but we choose to highlight Russia and the United States because of their large share of the dataset.

6.1.3 Coordinates

When we obtained our baseline for coordinate prediction, as mentioned previously, we placed every product at the location of the mean coordinate. Then we calculated the MED and got 1231 km, see Table 5.12. Our initial experiments of the different algorithms with default parameters showed the same trend as for the baseline. The predictions originated around the mean coordinate with some adjustments.

Even after we performed the hyperparameter optimization, the results showed quite poor distribution of the predicted coordinates, as can be seen in Figure 5.8. Only the ANN proves to predict coordinates all across the country and started to resemble the true coordinates in Figure 5.7. On the other hand, we can also see that there is an increase of clearly faulty predictions, such as the ones placed outside the coast line. Investigating Table 5.12 seems to prove the theory that the more distributed the predictions are across the country, the lower is the MED.

ANN achieved the overall best score in comparison to all other methods with an MED of approximately 872 km. The result was quite disappointing in relation to the purpose of this thesis. After meetings with our supervisor, Martin Gunnäng, we came to the conclusion that, for it to be valuable to Alfa Laval, we had to narrow the error to somewhere below 10 km in MED. We tried to reduce the error with some hyperparmeter optimization but it only resulted in small changes.

The high MED does not seem to be a result of overfitting either, as can be seen in Figure 5.6. The graph shows that no matter the increase in epochs, the validation loss does not increase or decrease. Instead it drops to a threshold and stays there throughout the training. Thus, given the initially large error we realized that, we would never obtain an error small enough to satisfy the needs from Alfa Laval for this task.

6.1.4 Improvements

Unbiased data set

To obtain a completely unbiased result, we would need to predict on a subset of the dataset that has an unknown location. For that to be possible, someone would have to manually validate the predicted values. Further, to implement a completely unbiased model, one would need the manually located subset as a part of the training, test and validation set.

The reason that we did not include a completely unbiased dataset was due to two reasons. The main reason was that, given our dataset and the specific columns selected, it is next to impossible to make this manual prediction, which is also why our master thesis was originated. The second reason was that we could not see any obvious differences between the datasets and as far as we knew, they had similar distribution.

Improve the dataset

The dataset that we used for training is generally based on technical attributes in combination with a few others such as to what market the product was sold, how it was sold, etc. To improve the dataset, we could have done further feature selection. The reason why we did not select additional columns from the database was that they had too many empty values. We could have tried to include those columns and filled them out using some form of data imputation. For example using the mean value or the most common value of that column. However, there is a risk that these columns instead have a negative impact of the model. Thus, an ablation study would have been of great importance especially in this case.

To further improve our dataset, we could have investigated different columns that were not in the database. This could have been done through studies and meetings with sales companies within Alfa Laval to investigate what considerations are of importance when they manually determine the location of products.

Another idea could be to perform feature engineering. Feature engineering is often referred to manual creation of columns. This can be done by combining two columns into a third one in a way that gives the algorithm a correlation between them, not found otherwise.

Improve the model

One thing that we could have investigated further was to put a threshold on *predict_proba* when predicting countries and cities using RF. Then we could have, using the training and validation phase, investigated the outcome of predictions that had a probability higher than a certain value, a threshold. If the accuracy and F_1 score improved, we could have either chosen to discard the predictions below the threshold or segment the predictions into different certainties.

6.2 Conclusion

To summarize the solution that we created, we can clearly state that when it comes to exact coordinate prediction, the algorithms can't properly solve the task for the given data2set. On the other hand, the other two tasks: country and city prediction, show great promise. With an accuracy top score of 70.9% for country prediction and 57.1% for city prediction, this project exceeded all of ours and Alfa Laval's expectations.

References

Alfa Laval (2021a). Gasketed plate-and-frame heat exchangers. https://www.alfalaval.com/products/heat-transfer/plate-heat-exchangers/gasketed-plate-and-frame-heat-exchangers/.

Alfa Laval (2021b). Gphe selection guide. https://www.alfalaval.com/microsites/gphe/tools/selectionguide/.

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., and Wirth, R. (2000). Crisp-dm 1.0 step-by-step data mining guide. Technical report, The CRISP-DM consortium.

Cheng, Z., Caverlee, J., and Lee, K. (2010). You are where you tweet: A content-based approach to geo-locating twitter users. In *International Conference on Information and Knowledge Management, Proceedings*, pages 759–768.

Fix, E. and Hodges, J. (1985). *Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties*. Number v. 1-2 in Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties. USAF School of Aviation Medicine.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Gaman, M., Cojocariu, S., and Ionescu, R. T. (2021). Unibuckernel: Geolocating swiss german jodels using ensemble learning.

Géron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Sebastopol, CA.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Mcculloch, W. and Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147.

Microsoft (2021). Parameters. `https://lightgbm.readthedocs.io/en/latest/Parameters.html`.

Natekin, A. and Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, 7:21.

Nugues, P. (2014). *Language Processing with Perl and Prolog: Theories, Implementation, and Application*. Cognitive Technologies. Springer Berlin Heidelberg.

Quinlan, J. R. (1986). Induction of decision trees. *Mach. Learn.*, 1(1):81–106.

Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P., and Feuston, B. P. (2003). Random forest: A classification and regression tool for compound classification and qsar modeling. *The Journal for Chemical Information and Computer scientists*.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

Verma, C., Stoffová, V., and Illés, Z. (2020). Prediction of residence country of student towards information, communication and mobile technology for real-time: preliminary results. *Procedia Computer Science*, 167:224–234. International Conference on Computational Intelligence and Data Science.

Wirth, R. and Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*.