UPPSALA
UNIVERSITET

UPTEC STS 21019

Examensarbete 30 hp

Juni 2021

Customer segmentation using machine learning

Axel Johansson
Jonas Wikström

Civilingenjörsprogrammet i system i teknik och samhälle

UPPSALA UNIVERSITET

Customer segmentation using machine learning

Axel Johansson
Jonas Wikström

Abstract

In this thesis, the process of developing an application for segmenting customers with the use of machine learning is described. The project was carried out at a company which provides a booking platform for beauty and health services. Data about customers were analyzed and processed in order to train two classification models able to segment customers into three different customer groups. The performance of the two models, a Logistic Regression model and a Support Vector Classifier, were evaluated with different numbers of features and compared to classifications made by human experts working at the company. The results shows that the logistic regression model achieved an accuracy of 71% when classifying users into the three groups, which was more accurate than the experts manual classification. A web API where the model is provided has been developed and presented to the company. The results of the study showed that machine learning is a useful technique for performing customer segmentation based on behavioral data. Even in the case where the classes are not naturally divisible, the application provides valuable insights on user behaviour that can help the company become more data-driven.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala

Handledare: Anton Aderum Ämnesgranskare: David Sumpter

Examinator: Elisabet Andrésdóttir

Populärvetenskaplig sammanfattning

Sedan internets födelse har världen blivit allt mer digitaliserad, där fler och fler processer sköts online. Internet och digitala tjänster har blivit en del av vardagen och många av oss kan inte tänka oss ett liv utan de tjänster som internet ligger till grund för. I och med den digitala utvecklingen och att fler människor använder sig av internet, har företag och organisationer tvingas anamma nya tekniker och metoder för att förbli konkurrenskraftiga.

Personalisering har vuxit till att bli en viktig process för företag, där de skräddarsyr sina tjänster och webbsidor för varje unik individ. Allt från innehåll till produktrekommendationer och e-post personaliseras för att förbättra kundupplevelsen, och på så sätt öka antalet kunder som genomför ett köp. Idag har personaliserade hemsidor blivit normen och något som förväntas av många kunder; företag som bedriver verksamhet online måste därför arbeta med personalisering om de vill förbli konkurrenskraftiga på en allt mer digitaliserad marknad. Till grund för personaliserat innehåll ligger stora mängder kunddata och kunskap om användarbeteende. Webbsidors innehåll kan personaliseras utifrån olika faktorer, exempelvis baserat på individens webbaktivitet, personliga drag eller demografisk data. Personalisering kan ske på individnivå men också på gruppnivå, där individen associeras med ett visst kundsegment som innehållet anpassas för.

Att analysera stora mängder data och segmentera kunder är en svår och kostsam process som många företag tampas med, men de senaste årens utveckling inom området Maskininlärning och Artificiell Intelligens har gett nya möjligheter att analysera och hitta mönster i data. Denna studie har därför undersökt hur metoder inom maskininlärning kan användas för att kategorisera användare till specifika kundgrupper. Projektet utfördes hos företaget Bokadirekt, som tillhandahåller en onlineplattform för bokning av hälso- och skönhetstjänster. För att uppnå målet med studien tränades och testades två maskininlärningsmodeller för klassificering. Deras resultat jämfördes sedan med två experter från företaget, som manuellt klassificerat ett antal användare. Den slutliga modellen kunde kategorisera användare med en träffsäkerhet på 71 procent, vilket var avsevärt bättre än experternas resultat. Arbetet resulterade i en tjänst som, med användning av maskininlärning, kunde kategorisera Bokadirekts användare i tre kundgrupper. Resultaten visade på att maskininlärning är en lämplig metod att använda för att kategorisera användare.

Acknowledgements

This thesis project was conducted at Bokadirekt, a platform for booking beauty and health services. We want to thank Bokadirekt for providing all material and data necessary to complete this thesis and especially our supervisor Anton Anderum, for all the help and support throughout the project. The collaboration worked excellent without any problems.

We also want to thank our subject reviewer, David Sumpter at Uppsala university, for his guidance and tips that were provided to us during the project, it would not have been possible without your knowledge in the subject.

Distribution of Work

This thesis project has been carried out by Jonas Wikström and Axel Johansson in close collaboration. Most of the code has been written separately, but in two copies carrying out the same task. Some code has been developed using a remote pair-programming technique via a shared desktop, where one person writes and modifies the code while the other person is monitoring and making suggestions. These techniques have made it easy to spot mistakes, partly because of having one person monitoring, but also due to the fact of having two versions of each function in the code. This approach also ensured that both authors covered all parts of the project. The pair-programming method also allowed for more creative solutions, since the problems at hand were discussed and analyzed. All parts of the thesis report were written in close collaboration.

Table of Contents

1 Introduction

The internet has since its beginning taken a larger place in our lives each passing day. Today most things in our society depend on the internet, and most people use it not only for work or school, but for everyday tasks such as shopping, banking, ordering food and making appointments at the hairdresser (Data reportal, 2021). As we use these online services, we also provide a lot of data about ourselves to the companies providing the services. The companies can then use the data to improve their platform, and in turn profit from it. This is something most people are aware of, and have come to terms with; in order to use the service they accept that their data can be used by the company. How companies use data about their customers to improve their service is an area that has seen a lot of change in recent years, much thanks to the advancements and subsequent cost reduction in storage that has allowed companies to store a lot more data (Mayer-Schönberger and Kukier, 2013).

There are of course multiple ways to analyze user data, but fundamentally the goal is to try to identify characteristics that can be found in many of the users. It could be characteristics such as user actions or user attributes of different kinds that, when identified, can be used to create groups that represent a collection of users (Amplitude, 2021). This process goes by many names: customer segmentation, user profiling, user behaviour analysis to name a few, but the end goal is to gain insight about user behaviour: who the users are, what they are doing and how they are doing it. That insight can then be used in a variety of ways, but most of the time it is used to customize and personalize the product or service to specific groups of users, so that their experience is improved (Mobasher and Anand, 2005).

This is becoming more common by the day, and it has evolved into an important process within the field of marketing (Kotras, 2020). It is no longer only the big tech-companies such as Netflix and Spotify that tailor their services based on the user, but a lot of smaller companies also use some type of customized content. Customers have grown accustomed to content and ads that have been tailored for them, and many expect the services they use to be personalized. An implication of this is that businesses that operate online must work with customer segmentation and personalization, if they want to remain competitive in an increasingly digitized and user-tailored market (Brownlow et al., 2015).

But achieving effective customer segmentation and analyzing large amounts of data, is a difficult and costly process that businesses often struggle with. Many large tech companies use machine learning technologies to analyze data with the intent of gaining insights about users and predicting future behaviour (Netflix, 2021) (Amazon, 2021) (Alemany Oliver and Vayre, 2015). More companies have gained interest in machine learning technology during the last couple of years, and it is therefore a subject worth studying.

This study was conducted at Bokadirekt, which is a company that provides an online platform for booking health and beauty services. Over the years they have gathered a lot of data about their users, which they currently do not fully utilize. They want to change that, and therefore this thesis project arose, where machine learning methods are explored as a tool for gaining insight about users, from their data.

1.1 Study goal

The purpose of this study is to explore the possibilities of using machine learning as a tool for identifying and understanding user behaviour on a booking platform for beauty and health services. The aim is to develop an application that can group and classify users based on their booking data, which then could be used to improve the company's platform.

In order to achieve the purpose and goal of the study, the following research question has been formulated.

1. Is it possible to use machine learning to identify certain user groups, based on users' booking data?

1.2 Disposition

This thesis consists of eight different chapters, in the first chapter there is an introduction of the subject and the project, with its goal and purpose, as well as delimitations. Following that is a chapter which describes the company the project was carried out at, some background to their problem and an introduction to the different users they have. Chapter 3 then covers the theoretical parts needed to complete this project, and how the different methods used work in theory. The following chapter starts with describing how the company's problem was redefined as a machine learning problem that could be solved by a machine learning approach. After that the process of collecting and processing the data is described, as well as how the model was implemented and evaluated. Thereafter the results are presented, with a description of the final model and its performance, which is also compared to human experts' performance. In chapter 6 the results are then discussed and the conclusions are then covered in chapter 7, followed by some ideas for future research in the area.

1.3 Terminology

Robustness - A model's sensitivity to changes in the input data

Decision boundary - The boundary that divides the vector space into two rooms, one for each class

Linearly separable - A dataset that can have a linear decision boundary is linearly separable

Feature space - The collection of all features used to characterize the data

Model parameter - A variable of the model (used interchangeably with "weight" and "coefficient")

User - A person who uses the company's platform (used interchangeably with "customer")

Persona - A number of predefined characteristics that describes a user group

Raw data - Raw data is data that has been collected from a database and that has not been processed

Record - A record is a "row" in a dataset consisting of a collection of attributes/features.

Attribute - A field/column in the original dataset that describes something

Feature - A field/column in the dataset that has been created from the original attributes

Feature vector - A vector of features that represents an object, in our case a user

Observation - A machine learning term for a record consisting of features. (used interchangeably with "datapoint", "sample" and "record")

2 Background

This project was conducted at Bokadirekt, the provider of Sweden's largest platform for booking health and beauty services. The platform allows users to book haircuts, massages and other beauty treatments at thousands of salons all over the country (Bokadirekt, 2020). Bokadirekt, which will interchangeably be referred to as the “company” in this thesis, has more than four million visits and processes around a million bookings each month (Bokadirekt, 2020). With all bookings and traffic on the site, the company generates a lot of user data, which they currently do not take full advantage of. User data can be very useful for companies as it often contains valuable information about the users’ behaviour and their preferences. If analyzed and interpreted correctly, it can provide a good understanding of the customer-base, which helps companies make well substantiated decisions. Therefore it is of interest for Bokadirekt to analyze and use their data to learn more about their users.

In many of the company’s processes, such as software design and marketing, they use a few so-called user personas as hypothesized groups of users that use their platform in a similar way. A persona is an imaginary person that represents a segment of real people within a defined population (Cooper, 2004; Pruitt & Grudin, 2003). A user persona represents the goals and behaviours of a user group and the properties that characterize that group.

The company has, during their many years in business observed three types of users that have been translated into personas. These are *impulsives, regulars* and *browsers.*

Impulsives as the name suggests, are impulsive users who use the platform when they have an urgent need. For example, a user who one day wakes up with lumbago and therefore wants to see a chiropractor as fast as possible. These users do not necessarily plan ahead or care about which salon they go to. They also have a tendency to make quick decisions.

Regulars are users that often visit the same salon and often book the same service on a regular basis. The purpose of the website visit is clear, and the user uses Bokadirekt to schedule an appointment at the salon he or she wants to visit. These users tend to be a bit more structured than *impulsives*; they often visit the same salons and plan their bookings some time ahead.

Browsers are users who like to try new things and browse the website a lot. The ultimate goal of the visit to Bokadirekt is not entirely clear. The person uses Bokadirekt to find something fun to do but does not necessarily know what service or salon to book on beforehand.

The company knows that these personas, or groups, exist amongst their users, but they do not currently know which users belong to which group. Therefore, the goal of this project has been to create an application that can categorize the company's users into the different personas, based on their data.

3 Theory

3.1 Machine learning

Machine learning is a branch of artificial intelligence (AI) that can be described as the study of algorithms and statistical models that are based on data (Nationalencyklopedin, 2021) (IBM, 2020). In machine learning, algorithms are often trained on big amounts of data with the purpose of identifying patterns or to make predictions on new data. Machine learning and AI are fairly new disciplines that were introduced during the 20th century, however many of the statistical concepts used in various algorithms date back much longer (James et al., 2013, pp. 6). Machine learning has now made its way into most fields where there are large amounts of data and complex problems that need to be solved.

Machine learning problems can generally be divided into two types of problems, namely supervised problems and unsupervised problems. Supervised machine learning is where there is training data characterized by a collection of tuples (x_i, y_i), where the input variable x_i is the set of attributes for record i and y_i is its output label or class. The input variable x is related to the output variable y and the goal is to learn a model that can predict the output y for a new record, for which only x is known. Depending on whether the output is numerical or categorical, the supervised problem is further recognized as a regression or classification problem (Lindholm et al., 2021, pp. 7).

In unsupervised learning there is no output variable y in training data, but one still tries to find some pattern or extract some type of information from the data. What type of information this could be is bound to the domain where the problem lies. Unsupervised learning could for example involve clustering, where records with similar attributes are grouped up together in so-called clusters (Lindholm et al., 2021, pp. 204) (James et al., 2013, pp. 27) (Roman, 2019).

In this thesis the focus is on supervised learning and specifically classification problems, which is explained in more detail in the coming section.

3.2 Classification

Classification tasks' main goal is to decide which label(class) an observation belongs to; in this case the classes are the three different personas. Classification problems are usually divided into binary or multiclass classification, depending on how many different classes there are. The concept of binary classification is rather simple, where there are two classes to decide between. A classic example is the spam filter, where new emails are to be classified as either spam or non-spam. Multiclass classification on the other hand, as the name suggests, involves multiple classes to choose from (Geron, 2019).

There are lots of algorithms available for solving classification problems. Some of them can handle multiclass problems, while others can not. There are, however, some strategies that make it possible to use binary classifiers for multiclass problems. One-vs-rest and One-vs-one are two of the most common ones (Geron, 2019) (Band, 2020).

One-vs-rest, OvR for short, involves breaking down the multiclass problem into as many binary classifiers as there are classes. For each class C_i of the K classes, a classifier is trained with the complete dataset. All observations that do not belong to class C_i are re-labeled to a negative class with a label (-1). For each of the K classes, there is a classifier trained to classify the observations as either class C_i or (-1). The biggest drawback with this method is that each classifier is trained on all training data, and the class-weights are very unbalanced since there are a lot more observations belonging to the negative class (Geron, 2019) (Ng et al., 2014).

One-vs-One, OvO for short, consists of classifiers trained on pairs of the K classes. Each classifier is trained with a subset of the data, and the final prediction is based on the output of all the $\frac{K*(K-1)}{2}$ classifiers. The strength of this technique is that the training set for each classifier is rather small, and the class weights are a lot more balanced. It does need a lot more classifiers in total however (Ng et al., 2014) (Band 2020).

There are multiple methods and algorithms that use the techniques described above, and the choice of algorithm depends on the problem at hand and especially on what the desired form of output is (Nelli, 2018). If for example the problem has multiple classes that are hard to distinguish between, it can be useful to get the probability estimates of an observation belonging to a specific class, which logistic regression can provide naturally (Lindholm et al., 2021, pp. 134-140). In the following section, a couple of the most commonly used methods for classification problems will be described in more detail.

3.3 Classification algorithms

3.3.1 Logistic Regression

Logistic regression is a popular machine learning classification method that is also used in various fields like traditional statistics, social science and medicine fields (Tolles and Meurer, 2016) (Osborne, 2012). The technique, in its basic form, is used for binary classification problems where there are two classes, but it can also be extended to solve multiclass problems. For the binary classification problem the model basically combines linear regression with a sigmoid function, that maps the output variable z from the regression model (1) to a defined interval. In logistic regression this function is called the logistic function (2) which squeezes the output variable z to the interval [0,1]. Therefore the output can be interpreted as a probability that the input belongs to a certain class (Lindholm et al., 2021, pp. 134-140) (James et al., 2013, pp. 127-138).

$$z = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n \tag{1}$$

$$P(z) = \frac{e^z}{e^z + 1} \tag{2}$$

The logistic regression model parameters β_i i.e weights for each input variable, are trained with maximum likelihood estimation. This is an iterative method which seeks estimates for the model parameters by maximising the likelihood of observing the training data (James et al., 2013, pp. 127-138) (Lindholm et al., 2021, pp. 134-140).

For the multiclass problem the logistic function is replaced with a generalization of the logistic function: a softmax function which outputs something that can be interpreted as probabilities for all classes in vector-format. All probabilities are non-negative and sum to one. One individual class probability can be written as:

$$P_{y=m} = \frac{e^{z_m}}{\sum_{j=1}^{M} e^{z_j}} \tag{3}$$

In multiclass logistic regression, each class m has one set of model parameters and cannot be trained with maximum likelihood. Instead, all parameters are trained with cross-entropy loss. (Pedregosa et al., 2011) (Lindholm et al. 2021 pp. 139-140).

3.3.2 Support Vector Machines

Support Vector Machine is a term used as a collective name for a few different classification methods, namely the Maximal Margin Classifier, the Support Vector Classifier (SVC) and the Support Vector Machine (SVM). They are all based on the same basic principle, to find a decision boundary that divides the different classes in the data. The Maximal Margin Classifier is the most basic, it tries to find a linear decision boundary that divides the data but also maximizes the margin, which is the distance from any point to the decision boundary. It is intuitive and easy to implement, but not very robust (James et al., 2013, pp. 336-356). An illustration of the Maximal Margin Classifier for two classes can be seen in figure 1.

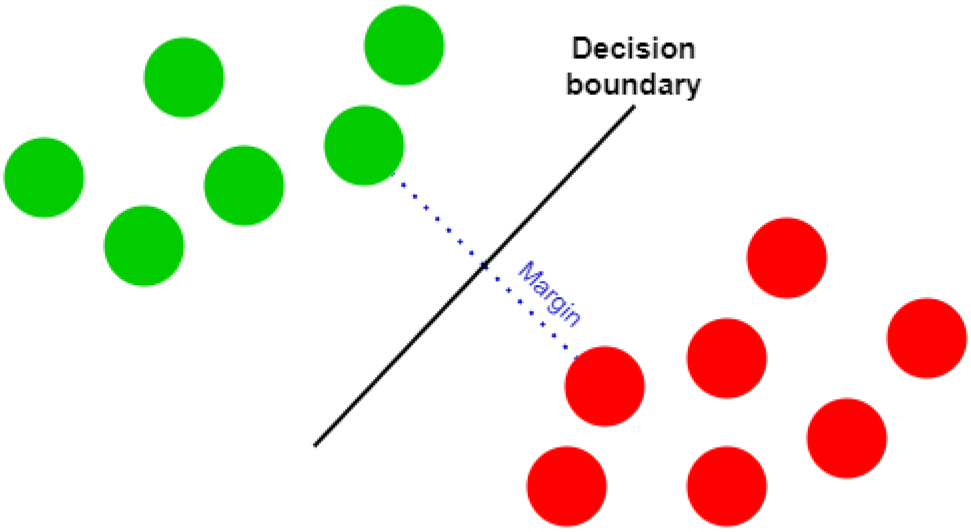


Figure 1: Illustration of the Maximal Margin Classifier with its decision boundary and margin

The Support Vector Classifier is an extension to the method, which specifically improves the robustness. Instead of finding a linear decision boundary that maximizes the margin to any datapoint, it finds a linear decision boundary that may have some observations within the margin or even on the "wrong" side of the boundary. This approach allows for some observations to be misclassified, which in turn makes the algorithm more robust. It often yields a better classification accuracy of the remaining observations (James et al., 2013, pp. 336-356).

Both of these two methods work only with linear decision boundaries, but far from all datasets in reality are linearly separable. For those that need a non-linear decision boundary, there is the support vector machine, which is an extension of SVC. It works by essentially enlarging the feature space, by using features of higher dimensions when fitting the classifier. For example instead of using n features when fitting the classifier, it uses 2n features, where all features have also been squared. This might seem simple, but it does have some drawbacks that makes it a bit more complicated however. If a higher polynomial degree is used, there will be a large amount of features created, which in some cases can lead to cumbersome calculations. This will make the model very slow, and in some cases it might not even be practically possible to make the calculations. This is where SVM has its advantages as it uses kernels for enlarging the feature space, which is a method of achieving a higher dimension feature space without actually having to do the heavy calculations (Geron, 2019).

3.4 Data preprocessing

Raw data that is extracted from a database comes in different shapes and quality. Data objects or attributes may be of different types: some are quantitative, represented by a numerical value, whilst others are qualitative, for example represented by a string of words. The raw data is unprocessed and often contains noise and outliers. Sometimes there are duplicate records and some records might be missing values in certain fields. In order to deal with these types of issues, the data needs to be preprocessed into a fitting format that can be used to train a machine learning model (Larose, 2005, pp. 27-39) (Tan et al., 2014, pp. 3). Tan et al. (2014) states that most machine learning algorithms can handle some deficiencies in the input data. However, the authors argue that spending time refining the data often is worth it, as it usually has an impact on the final model's performance as well.

Data preprocessing is one of the most time-consuming and important phases of a machine learning project, since algorithms require meaningful and manageable data to provide useful knowledge and predictions (Garcia et al., 2015, pp. 1-15) (Tan et al., 2014). The preprocessing phase can include many steps such as data cleaning, integration, transformation of data, feature extraction, feature subset selection etc (Garcia et al., 2015, pp. 1-15) (Tan et al., 2014). The choice of techniques are however tightly bound to the structure of data, the specific problem and its domain.

In the following sections the theory and reasoning for using some techniques that were found useful for this specific problem is covered.

3.4.1 Data cleaning

The data cleaning process refers to a handful of techniques for dealing with various errors in data. As mentioned in the previous section; unprocessed data often contains some sort of errors, for example missing values, duplicate records or outliers. These issues need to be addressed to avoid training a faulty and biased machine learning model. Chu and Ilyas, in the book "Data Cleaning" (2019) expresses that data cleaning often consists of two phases; the error identification phase and the error repair phase. The first phase may involve statistical or qualitative techniques to detect corrupted records such as outliers, duplicate records and violations, while the repair phase focuses on correcting or removing these records. The authors further state that human experts in the area are often consulted during the cleaning process to define what is considered “normal” and what is abnormal (Chu and Ilyas, 2019, pp. 1-5).

3.4.1 Feature extraction

The construction of new features from raw data is called feature extraction. The general idea behind feature extraction is to merge or map data to a new space where it captures information much more effectively (Tan et al., 2014, pp. 53). According to Tan et al. (2014) merging attributes can gain the benefits of dimensionality reduction which can increase the performance of machine learning algorithms. However, aggregating multiple records into one object has the potential disadvantage of losing important information. Tan et al. (2014) also argues that feature extraction is highly domain-specific and that the development of new features is a core task when applying machine learning to a new area. Before new fruitful features can be created, a good understanding of the application area as well as of the raw data at hand is needed (Nelli, 2018, pp. 1-10).

3.4.2 Standardization

The values of different data features often lie within different ranges, which can cause problems for various machine learning algorithms. Features with large values may dominate the result of a model, even though features with smaller values may be of greater significance (Tan et al., 2014, pp. 64) (Theodoridis and Koutroumbas, 2009, pp. 263). This problem is overcome by standardizing the data so that all features’ values lie within similar ranges (Theodoridis and Koutroumbas, 2009, pp. 64). There are several methods for standardizing the data, but one of the most common techniques transforms the feature to a new variable that has the mean of zero and standard deviation of one (Tan et al., 2014, pp. 64) (Theodoridis and Koutroumbas, 2009, pp. 263). Tan et al. (2014) also states that the mean and standard deviation are strongly affected by outliers, which affects the standardization.

3.4.3 Feature selection

Not all features are necessary or contain useful information in order to implement a successful machine learning algorithm. Sometimes features contain much of the same information which makes them redundant (Tan et al., 2014, pp.52). Not only are there redundant features, but also irrelevant features which contain no information that is useful for the specific problem (Tan et al., 2014, pp.52). By selecting a subset of features, the dimensionality of the data can be reduced, which can increase the performance of algorithms in terms of accuracy, i.e. its ability to classify correctly, but also has advantages in terms of computational time. Tan et al. (2014) states that some features can be removed straight away, just by using common sense or domain knowledge whilst the selection of others require a more systematic approach (Tan et al., 2014, pp.52).

Tan et al. (2014) states that the ideal way to select a subset of features is to test all combinations of features and choose the subset which yields the best result from the classification algorithm. However, this approach is very costly and in most cases other approaches are needed. There is an abundance of different techniques, but most are experimental and involve comparing results of different feature subsets (Tan et al., 2014, pp.54). After finding the optimal subset of features, one should also contemplate if it's beneficial to remove features and lose accuracy for the reason of reducing the dimensionality.

3.4.4 Sampling

The number of samples belonging to each class is most of the time uneven, where one class has more samples than the others. The class in the dataset which holds the most samples is referred to as the *majority class* and the class consisting of relatively less data samples is referred to as the *minority class*. Training a classification algorithm with a skewed dataset can affect the performance, causing it to produce false predictions (Mishra et al., 2020). López et al. (2013) states that class imbalance problems often cause classification algorithms to be biased towards the majority class. This problem often prompts a higher misclassification rate among the minority classes. There are several methods for addressing this issue and the method of choice can be categorized into one of two groups: internal approaches that create and modify algorithms to handle skewed class distributions and external approaches that preprocess the data to diminish the imbalance (López et al., 2013). The latter approach often involves resampling techniques such as under-sampling or over-sampling, where samples are removed or added to the dataset (Mishra et al., 2020). Resampling techniques have shown great success in empirical studies, and have the advantage of being unconstrained by the choice of classifier (López et al., 2013).

3.5 Model evaluation

A classification model needs to be evaluated before being used in practice. The classification model is fitted on the training data, but it also needs to accurately predict the labels of new, unseen data (Lindholm et al., 2021, pp. 47) (Tan et al., 2014, pp. 148). Therefore it is important to evaluate the performance of the model on test data which it has not been fitted on. The validation process also helps with choosing between different classification algorithms and setting hyperparameters for each model (Lindholm et al., 2021, pp. 47). Mohri et al. (2012) explains that a model may commit no error on the records of training data but still have a bad performance on test data. This is known as *overfitting* and occurs for complex classification algorithms when the training set is too small.

There are several techniques for evaluating the performance of a model. The most basic method is called *hold-out validation,* where the data is divided into three subsets: one training set, one validation set and one test set. The training set is used to learn the classification model. The validation set is used for estimating the performance of the model in the training phase, and as a platform for tuning hyperparameters and finding the optimal subset of features. The test set is used to evaluate the model's expected future performance. Since the model(s) are tuned with respect to the performance on the validation set, there is a risk of overfitting to the validation data. Therefore, the independent test set is needed as a final estimator of performance (Lindholm et al., 2021, pp. 52) (Mohri et al., 2012, pp. 4) (Mutuvi, 2019) .

However, if the amount of data is limited, this technique could leave an insufficient amount of data for training. If this is the case, another widely used method: *k-fold cross validation* can be used for evaluating the performance. In k-fold cross validation the data is split into training data and test data, where the training data is used to calculate the k-fold cross-validation error. The training data is partitioned into *k* folds - typically 5 or 10 folds of equal size. One of the folds is *held-out* as validation data and the model is trained with the other k-1 folds. This procedure is repeated until all folds have been used as validation data once. The cross-validation error is calculated by computing the mean error of all iterations (Lindholm et al., 2021, pp. 52) (Mohri et al., 2012, pp. 4) (Mutuvi, 2019). A visual explanation of the technique is shown in figure 2.

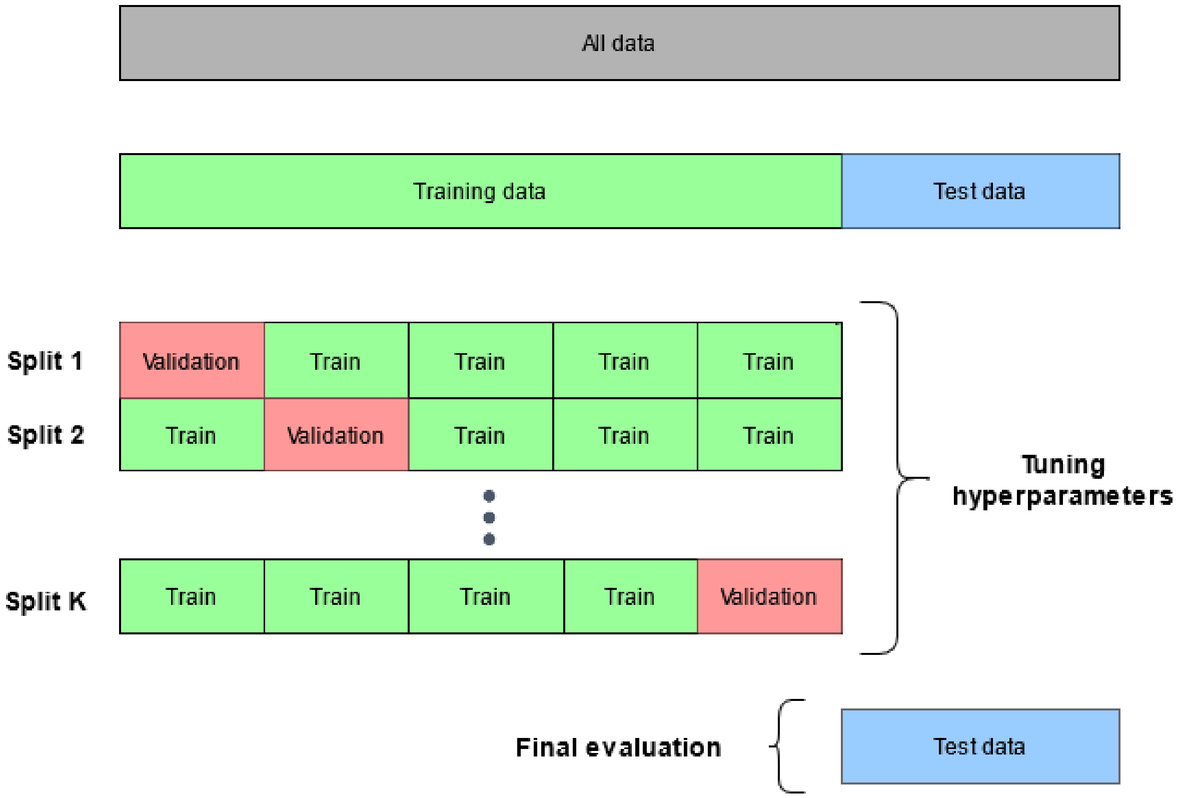


Figure 2: K-fold cross validation. The data is initially split into training data and test data. The training data is split into k folds, where each fold is used as validation data once and the model is trained on the remaining k-1 folds. The final model is trained on all training data and the performance of the model is evaluated on the test data.

Since all of the intermediate models are trained on the same data except one 1/k fraction - the cross validation error is considered a good enough approximation of the final model trained on all available training data (Lindholm et al., 2021, pp. 52). The final performance is evaluated on the test data, which is the same as for hold-out validation.

3.5.1 Evaluation metrics

In supervised learning, different evaluation metrics are used to evaluate the performance of a model. It is important to understand what these metrics are telling us about our classification model in order to improve it. The evaluation of a classification model is based on the number of correctly and incorrectly predicted test records summarized in a number of metrics (Tan et al., 2014, pp. 149). The different metrics for evaluating a model is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = true\,positive\,rate = \frac{TP}{TP + FN} \tag{6}$$

$$F1\,score = 2 \times \frac{precision \times recall}{precision + recall} \tag{7}$$

The notation *TP* is the number of *true positives (TP)* i.e. the number of instances where the model *correctly* predicts the *positive* class. Similarly, *true negative (TN)* is the number of instances where the model *correctly* predicts the *negative* class. *False positive (FP)* is the number of false predictions of the positive class and *false negative (FP)* is where the model incorrectly predicts the negative class (Fawcett, 2006). *Accuracy* can be described as the proportion between the amount of correct predictions and total number of predictions. Lindholm et al. (2021) describes *recall* as "how large proportion among the positive data points that are correctly predicted as positive" and *precision* as "what the ratio of true positive points are among the ones predicted as positive". The *F1 score* is an aggregated measure for precision and recall, calculated by the harmonic mean (Lindholm et al., 2021, pp. 65).

4 Method

As previously mentioned, the goal of the project was to produce an application that could categorize the company's users based on their previous behaviour on the platform. Since the data has been the most central keystone for this problem we start by giving the reader an overview of the company's data and the different data sources. Thereafter, we present the problem and how we transformed it into a machine learning problem that could be solved by developing a classification model. In section 4.2 the tools and software used to develop the application is described. The following two sections cover the preliminary investigation of data, as well as a more detailed description of the preprocessing steps taken in order to transform the data into a suitable format for the classification algorithm. In section 4.5, the training and validation of the model is described. The last section covers how the model was evaluated.

A summary of the different steps in the process can be seen in figure 3.

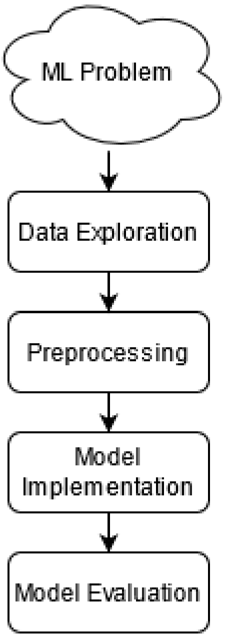


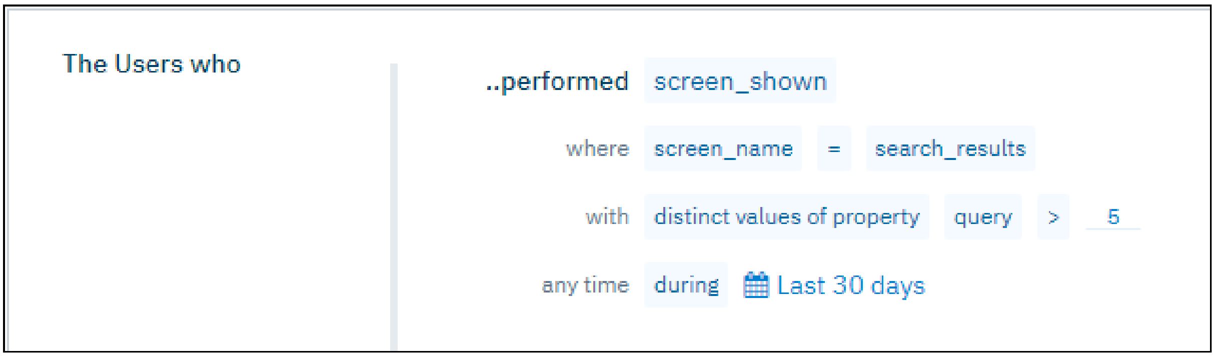
Figure 3: Illustration of the workflow after the machine learning problem was defined, starting with data exploration and ending with the final evaluation of the models.

4.1 The machine learning problem

4.1.1 Data

There have been two sources of data used during the project: a cloud-based relational database where all bookings made on the company's platform is stored, and Amplitude, a tool for data analysis where all event-data from the company's website is stored. The relational database contained a lot of attributes connected to a booking, fields such as; the users who made the booking, the time of the appointment, the appointment type, price, duration, salon id, location etc. The event-data from the website on the other hand consisted of events triggered during a user's sessions on the site, events such as searches, viewed salons and services etcetera. The company has used Amplitude for approximately a year, which means that data older than that is not available.

Specific data from the relational database could be accessed by the company via certain queries. This data was thereafter provided in comma-separated files (.csv format). The event-data on the other hand was not stored so that it could be accessible in the same way, but the software Amplitude provides a tool where filters can be set up. The output from these filters were lists of users that matched the specific criterias in the filter. Figure 4 shows an example of how a filter can be set up, this specific filter returns a list of all users who made more than 5 unique searches on the website during the last 30 days.



The Users who

..performed screen_shown

where screen_name = search_results

with distinct values of property query > 5

any time during Last 30 days

Figure 4: A filter in Amplitude which returns a list of users which made more than 5 unique searches on the website in the last 30 days.

The two data sources are very different and capture different dimensions of user behaviour; since it is only possible to set up filters for the event data, it is not possible to extract any attribute values from that datasource. As the filter in figure 4 shows, it is only possible to extract the user IDs of those who meet the criteria in the filter; it is not possible to retrieve a specific users' data, e.g. how many searches the user has made. Therefore it is impossible to know whether a user did 6 searches or 50 searches without setting up filters for all natural integers. However, it is possible to find users' who have displayed a certain behaviour on the website, which is not possible from the booking database.

4.1.2 Classification

The company has been using three different personas in previous development processes, but they were lacking a way to identify which users belonged in which persona-group. That knowledge would be useful for personalizing the website content, and to tailor marketing for the users.

As the three personas: browsers, impulsives and regulars describe the user groups' behaviour with the service, it is possible to identify these personas by setting up criterias for each of them in Amplitude. This is a good way of finding users that match with the behaviours covered by the persona, but because of its limitations it was not feasible to use only Amplitude for the segmentation.

First of all, Amplitude is not Bokadirekt's own system, which means that they have little or no control over it. Secondly, the data that Amplitude generates only exists for users that have an account, and that are logged in; users that are using an adblock software or plugin are not captured by the system for example, which further reduces the number of users. Lastly Bokadirekt has only used Amplitude for about a year, so no data older than that exists. The number of users captured by Amplitude is thus only a fraction of Bokadirekt's customer base. Because of all these limitations, using the event-data was not deemed possible on a large scale. Therefore the booking data seemed to be a solution, as it was a more complete datasource.

Since Amplitude worked great for identifying people that matched with a persona, we decided to use it to extract usernames that could then be cross referenced with the booking database. The result of this was a set of users with complete booking data **and** a persona label. The problem was then recognized as a supervised machine learning problem, specifically a classification problem, where the goal was to predict the persona-label of users using their booking data.

4.1.3 Labelling

As described in the section above, the event-data in Amplitude was a great tool for extracting users that had shown behaviours that matched with the three user groups'. The criterias used were developed together with the company and as the company has observed these personas during many years in the industry, their knowledge about their customers' behaviours was integral in defining the criterias.

The criterias for the **impulsive** persona was that the user had:

- viewed 10 or less salons during the last 365 days
- made 5 or more bookings scheduled within 3 days during the last 365 days
- made 5 or less searches with one of the 50 most common keywords during the last 60 days

The criterias for the **browser** persona was that the user had:

- viewed 20 or more salons during the last 60 days
- made 3 or more bookings during the last 60 days
- made 9 or more unique searches during the last 60 days

The criterias for the **regular** persona was that the user had:

- viewed 2 or less salons during the last 60 days
- made 3 or more bookings scheduled more than 7 days ahead during the last 365 days
- made 2 or less searches with **none** of the 50 most common keywords during the last 60 days

For the impulsive persona, the reasoning behind the criterias was that *impulsive* users have an urgent need and know exactly what service they want to book; therefore they do not search a lot and schedule their appointments within a short time. They also tend to make quick decisions and do not view plenty of salons before deciding on which one to book.

The reasoning behind the criterias for the browser persona is that a *browser* does not know what service or salon he or she wants to book prior to visiting the website; and therefore tend to view a lot of salons and make many unique searches, for example of different services.

For the regular persona, the reasoning is that a *regular* has a clear goal with the visit on the website; he or she often knows which service and salon to book beforehand and therefore does not view many salons. *Regulars* also tend to be a bit more structured than *impulsives* and plan their appointments some time ahead. The last criteria was set up because they have a tendency to search for the salon name and not a common keyword such as: “haircut” or “massage” for example.

These criterias might seem somewhat arbitrary, but they are based on the company’s knowledge about their customers. The criterias were discussed and adjusted over a period of time, where the company provided the foundation for each persona. Together we analyzed typical behaviour and discussed simple user stories for each persona. In this process, our own experiences and behaviour with the platform were also useful, and together with the company we came up with values that seemed reasonable for each criteria. In addition, analysis of behaviours is very complex, a user rarely fits distinctly into one persona only, but often shows behaviours encapsulated by other personas as well. Due to the structure of filters and limitations with Amplitude, not all users received a label; only users who were logged in during their sessions and met all criterias in a filter were labelled. The three user groups filtered out in Amplitude were also not disjoint, meaning that a user could be caught in multiple filters and thus receive multiple labels. To cope with this problem, any user that had been labeled with more than one class was simply removed from the data set. An illustration of the labelling process can be seen in figure 5.

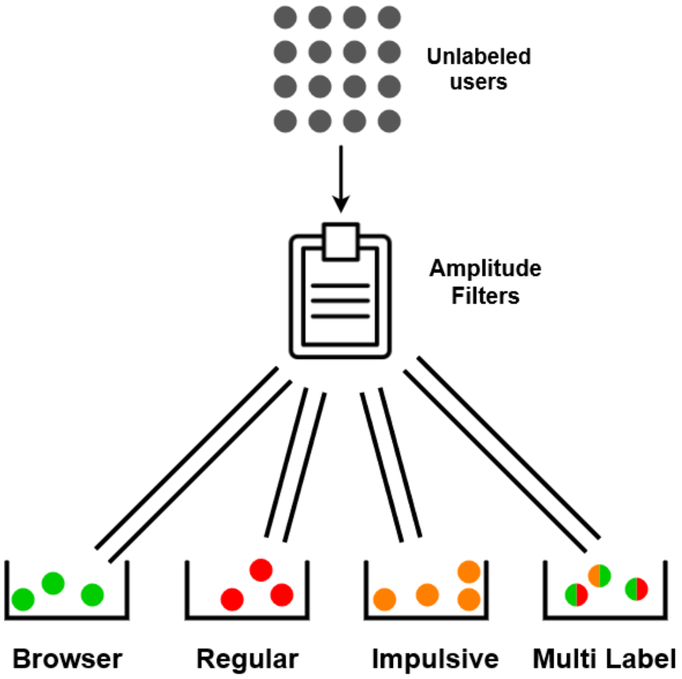


Figure 5: An illustration of the labelling process. A subset of all users are caught in the three Amplitude filters and labelled accordingly.

4.2 Tools

All code in this project was developed in Python, using various software libraries such as Pandas, NumPy, Scikit-learn, Matplotlib and Flask. The tools Pandas and Numpy were used for data analysis and preprocessing, Scikit-learn for implementing and testing machine learning algorithms, and Matplotlib as a visualization tool. Finally, Flask was used to create a web API that was a part of the final product. All data from the company was provided in CSV text files, which were converted to Pandas dataframes in order to have access to the library's built-in functions for data analysis and modification. Personal information of the users such as email address and name were anonymized and replaced with a user id.

4.3 Data exploration

This section covers the preliminary investigation of the data before choosing appropriate techniques for preprocessing. Tan et al. (2014) explains that data can be represented in many different ways and be of different quality, and that its characteristics is the determining factor when it comes to which techniques might be suitable. Tan et al. (2014) further states that data exploration and visualisation can help selecting relevant methods for preprocessing the data, but are also useful tools to interpret results.

During the data exploration phase, a lot of different attributes from the raw data were closely inspected and analysed. Many different combinations of attributes were plotted and visualized in order to find patterns and to get a deeper understanding of which attributes that might be relevant for the problem at hand. The data was for example divided on the service category, to identify deviations in attribute values for different categories. During this process, a lot of new summary statistics such as mean, median, variance, frequency etc. were put together.

This phase provided valuable insights on the data, as well as useful knowledge about the beauty and health industry. A few examples of insights that were gained about the data, that were especially useful for the project, are given below.

Upon inspection it was found that one user had booked 15 covid-19 tests in one day. That seemed peculiar, and it was likely someone booking tests for employees at a company. Another user had booked two haircuts on the same day and at the same time. After closer investigation, it was found that one of the appointments was for a child haircut and the other a regular haircut - probably a parent having a haircut at the same time as their child.

These are normal instances, but they cause problems when a user is interpreted as one person, and when a model to classify personas is being developed. How instances like these were handled is elaborated in the data cleaning section.

Due to the large number of attributes connected to each booking in the relational database, only the most useful ones such as date, salon, price etc. were selected for further analysis. These attributes were extracted from the database by the company and provided as CSV files.

4.4 Data preprocessing

It is important that the classification algorithm is provided with useful data to generate good and reliable predictions. This section describes the preprocessing steps and techniques that were used before training the classification algorithms. This step in the process of implementing the model was by far the most time consuming. It involved the following steps: Data cleaning, Feature extraction, Undersampling and Standardization, as can be seen in figure 6.

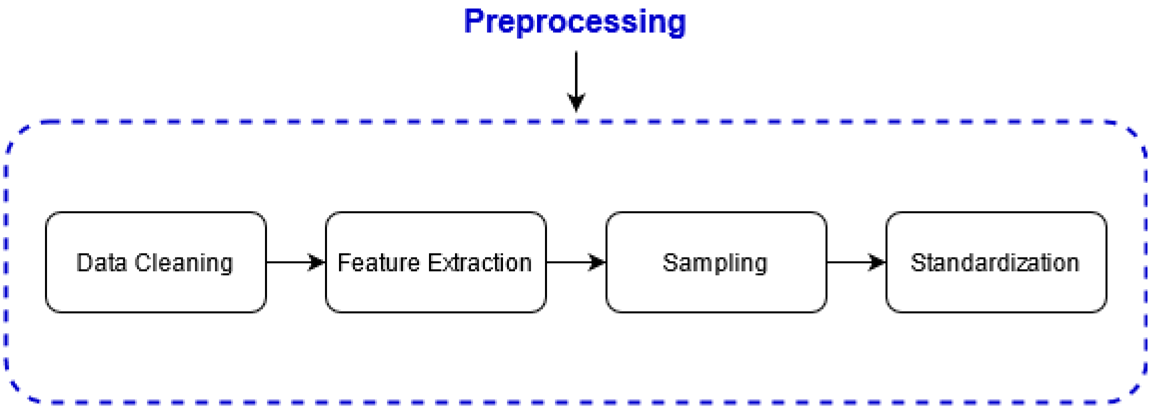


Figure 6: Illustration of the preprocessing steps and workflow.

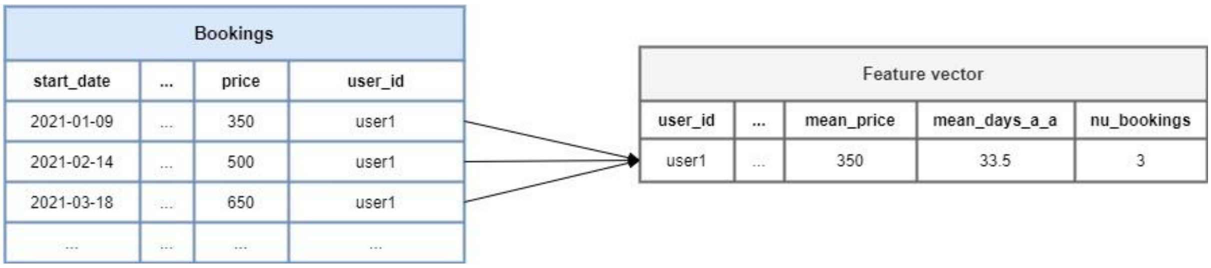
4.4.1 Data cleaning

The data exploration phase gave a good understanding of the data, its characteristics and what type of information it contained. The findings were discussed with the company that explained what was normal behaviour and what was not. One thing the company had to explain was why some records were missing values for the service price attribute; these records were often some types of consultations, so instead of removing them, the value was changed to 0. The dataset also contained some duplicates; these were pruned to avoid bias. Other than that, some users had unusually many bookings - these were considered anomalous, since they most likely were not actual users but instead companies or salons that used the platform. These anomalies were removed by setting a limit on how many bookings a user was allowed to have - any user with more than 15 bookings the last 365 days were removed from the dataset. Users that had less than three historical bookings were also removed from the dataset, as one or two bookings do not provide sufficient data to say something about the user's behaviour. This is elaborated further in the next section.

4.4.2 Feature Extraction

This section covers the process of creating new features, which involved combining attributes, computing similarities and distances between records and introducing statistical measures. The approach for creating new features was to create as many as possible that could be useful. This was the approach because many of the new features that were not useful for the classification model, the company had other uses for, in marketing for example.

In the raw data, each user had a number of rows, representing their historical bookings. Each booking is referred to as a record, with a set of attributes. During the feature extraction stage, a user's records were summarized and aggregated to a feature vector (see figure 7).



Bookings

start_date	...	price	user_id
2021-01-09	...	350	user1
2021-02-14	...	500	user1
2021-03-18	...	650	user1
...	...	...	...

Feature vector

user_id	...	mean_price	mean_days_a_a	nu_bookings
user1	...	350	33.5	3

Figure 7: Feature extraction. Each user's historical booking records are aggregated to a feature vector.

In general, data with timestamps often contains a lot of useful information that can be used to train machine learning models. In this case, the booking data contained two attributes with dates, namely *created_date* and *start_date*. The attribute *created_date* refers to the date and time when the person scheduled the appointment and *start_date* refers to the date and time of the appointment. Some typical questions that could be answered by inspecting these two attributes are: "how many days does the person book in advance?", "how much time is there between appointments?", "what weekday does the person prefer to go on appointments?" etc. The two attributes were used to create about 15 new features of the feature vector. However, not all of them were valuable for a classification algorithm, but many of them were useful for the company's marketing team.

One of the new features created was *recency,* which describes how recently a user booked an appointment. It is measured in days since the last booking, and can be used to identify customers that have not made a booking in a long time, as well as customers that are susceptible to advertising. The new feature *mean_days_b_a* is the average time between *created_date* and *start_date*. This tells us how many days a person books in advance on average. However, the mean measurement is sensitive to outliers and for these problems, the median is a more robust statistic for the middle value (Tan et al., 2014). Therefore, the feature *median_days_b_a* was created as well. The feature *mean_days_a_a* is the average time between appointments. There is also a corresponding *median_days_a_a* feature. The features *variance_days_a_a* and *variance_days_b_a* were also created to capture the spread of the values. These features together with the recency feature could be used as a basis for making marketing decisions, for example when to send a reminder to customers who have not made a booking in a long time.

Features for each day of the week were also created: *bookings_monday_perc*, *bookings_tuesday_perc* and so on. Each represents the fraction of appointments scheduled on that specific weekday. These features were not very important for the model, but could be used by someone in marketing to decide which day of the week to send out personalized marketing. Lastly the feature *pay_day_perc* was created, which describes the fraction of the users bookings that were scheduled within ± 4days of the 25th every month. This feature was created because salary in Sweden is normally paid out on the 25th every month.

Two other new features were *nu_bookings* and *monetary*, where *nu_bookings* is the total number of bookings and *monetary* is the total amount of money spent across all of a user's bookings. *Mean_price* is the mean price of the user's bookings and *median_price* the median price of the bookings. *Nu_salons* is the number of different salons a user booked. *Nu_services* is the number of different services a user booked. *Top_service_1* is the user's most frequently booked service. *Top_service_2* is the 2nd most frequent service and *top_service_3* consequently the 3rd top service. All of the new features created can be seen in table 1.

Table 1: New features. Description of features and unit of measurement.

Feature name	**Description**	**Unit**
Recency	Number of days since the most recent booking	Days

Mean_days_b_a	Mean days between booking date and appointment date	Days
Median_days_b_a	Median days between booking date and appointment date	Days
Mean_days_a_a	Mean days between appointments	Days
Median_days_a_a	Median days between appointments	Days
Variance_days_b_a	Spread of days between booking date and appointment date	Days
Variance_days_a_a	Spread of days between appointments	Days
Pay_day_perc	Percentage of bookings within ±4 days of the 25th	Percentage
Nu_bookings	Total number of bookings	Integer
Monetary	Total booking value	SEK
Mean_price	Mean price of the bookings	SEK
Median_price	Median price of the bookings	SEK
Nu_salons	Number of bookings at different salons (booked)	Integer
Nu_services	Number of different services (booked)	Integer
Top_service_1	The product ID of the user’s most frequently booked service	Integer
Top_service_1_perc	Percentage of bookings of the user’s most frequently booked service	Percentage
Top_service_2	The product ID of the user’s 2nd most frequently booked service	Integer
Top_service_2_perc	Percentage of bookings of the user’s 2nd most frequently booked service	Percentage
Top_service_3	The product ID of the user’s 3rd most frequently booked service	Integer

Top_service_3_perc	Percentage of bookings of the user’s 3rd most frequently booked service	Percentage
Bookings_Monday_perc	The percentage of the user’s bookings scheduled on a Monday	Percentage
Bookings_Tuesday_perc	The percentage of the user’s bookings scheduled on a Tuesday	Percentage

●

●

Bookings_Sunday_perc	The percentage of the user’s bookings scheduled on a Sunday	Percentage

4.4.3 Under-sampling

After the labelling process, described in section 4.1.3, there was a set of around 6500 users that had been labeled. The distribution of the labels was not very balanced however, as the majority class (regulars) contained more than 4500 samples and the minority class (impulsive) contained only 557 samples. In order to deal with the unbalanced classes, undersampling techniques were used to balance the ratio between them to a 1:1:1 ratio. Since the impulsive class was the smallest, 557 samples were randomly selected from the browser class and the regular class, which all together formed a new dataset consisting of 1671 samples that was used thereafter. A visual illustration of under-sampling is presented in figure 8.

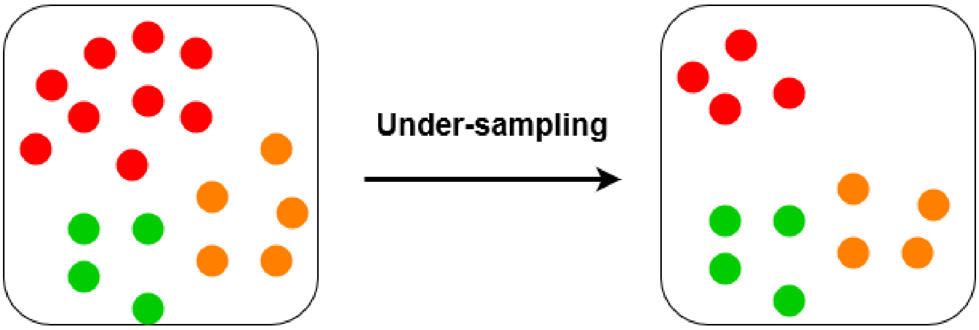


Figure 8: Illustration of under-sampling. The dots represent data points of the three classes.

4.4.4 Standardization

Many of the new features created in the feature extraction process had values in different ranges. For example, every value with the unit *percent* ranged between 0 and 1 whereas the feature *monetary* ranged from 0 to 60 000. The data was therefore standardized to avoid creating a model biased towards features with greater values. The features were standardized by removing the mean and scaling to unit variance (Pedregosa et al., 2011).

$$k = \frac{(x - \mu)}{s} \tag{8}$$

In (8), x is the sample, μ the mean and s the standard deviation which together makes up the new standardized feature value k. The classification models were also evaluated with data that wasn't standardized.

4.5 Model implementation

From the beginning of the project, the goal was to find a way to categorize the company's users based on their historical data. The goal was then transformed into a classification problem, described in section 4.1, and following that a training set of 1671 users' feature vectors and class labels had been created. Each record i (user) in the dataset was characterized by a collection of tuples (x_i, y_i), where input variable x_i is the set of features in the feature vector and $y_i$is the class; *regular, impulsive* or *browser*.

This dataset was used to train the final model that would be used in production, but before the model was finalised some decision had to be made. For example, which algorithm to use, which subset of features to include and which hyperparameters to use. The workflow of the model implementation phase can be seen in figure 9.

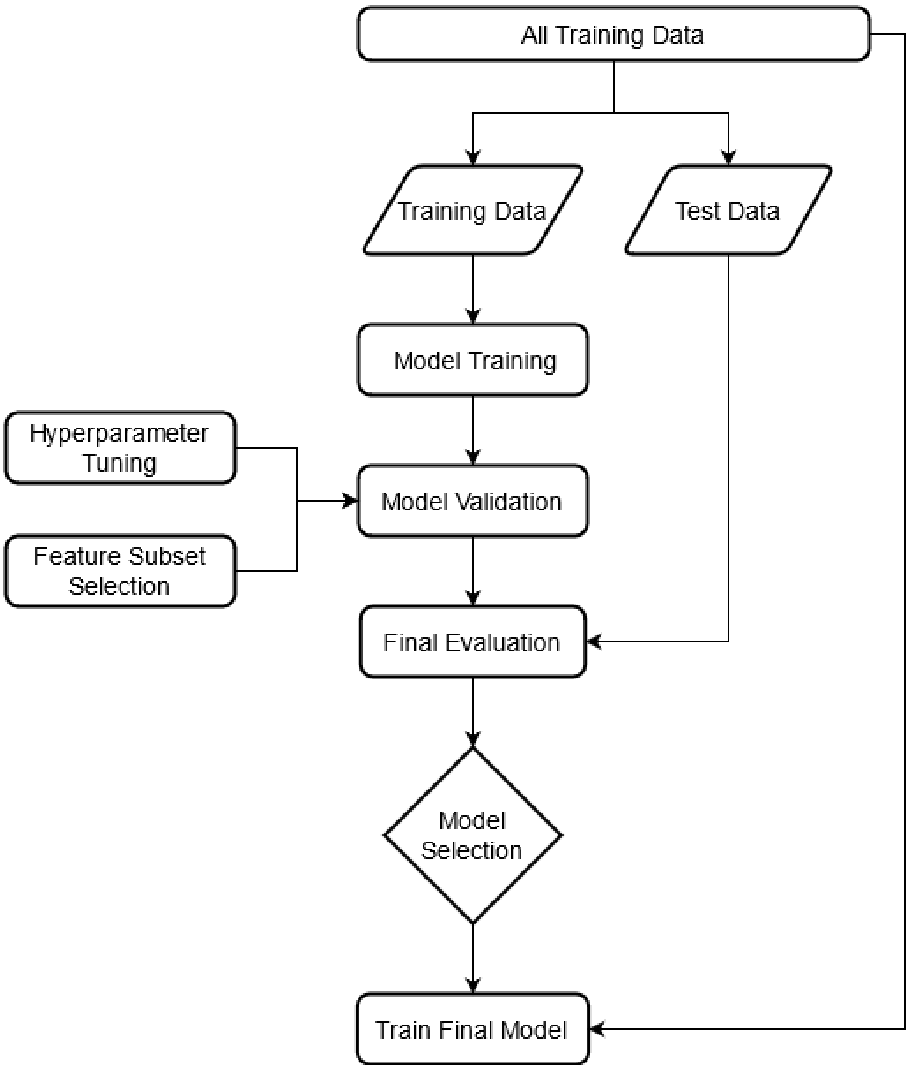


Figure 9: Illustration of the model implementation phase.

4.5.1 Model training & validation

In order to create the best possible model, a larger set of about 10 classification algorithms were tested. The 10 algorithms' performance was estimated using the 10-fold cross validation technique, where the cross validation error was used to compare the models. The support vector classifier and the logistic regression model yielded the best result and were therefore chosen as appropriate models for further calibration.

These two models were then tuned in an iterative process, with the purpose of creating the best possible model to be used in the final product. The two models' cross validation error was used as a measurement of performance, and in this iterative process, different subsets of features and hyperparameters were tested together. How the subset of features was selected is described in the next section.

4.5.2 Feature selection

After the feature extraction phase, each user had a feature vector with 26 features. Many of these were considered irrelevant for the classification task, so they were removed by hand. After these features were removed, a subset of 13 features were considered potentially useful for the classification models. As Tan et al. (2014) explains, the ideal way to select features is to try out all combinations of features and see which performs the best. That is a very costly process however, and a more reasonable method is to use a technique such as Recursive Feature Elimination (RFE).

In order to select an appropriate subset of features, RFE was implemented in python with the scikit-learn library. The technique allowed for a specific number of features to be set and it then recursively removed the least important features. It uses an estimator such as logistic regression to assign weights to each feature and then removes the least important feature from the set. This procedure is repeated until the desired number of features are left (López et al., 2013).

Since it was not known what number of features that gave the best results, the performance of the two classification models were evaluated with different numbers of features in a for-loop. The result of the feature selection process is presented in section 5.1 Features.

4.6 Model Evaluation

The two model's expected future performance was evaluated on an independent dataset that was not used in any sort of training. As Lindholm et al. (2021) mentions, this is good practice, since there is a risk of overfitting the models to the validation data and therefore creating a bad generalization.

In order to compare the two models overall performance, both the evaluation metrics described in the theory chapter, as well as the computational time of the models were taken into consideration. To measure the computational time, a timer function was implemented in the code that measured the time it took to classify 100 users, that time was then multiplied to get the corresponding time it would take to classify 100 000 users. The reason for training two classifiers was that it is favourable to be able to compare their performance, and then select the one with the best overall performance for the final implementation. For example, an algorithm can have high accuracy due to its ability to identify only one of the classes well, but perform badly on the others. Therefore, it is important to base the selection on the overall performance since the accuracy measure can be misleading.

It is often hard to interpret the different evaluation metrics and what is considered a good score in terms of accuracy, precision etcetera. Therefore it is good practice to have some other performance measures to compare with, which gives some perspective to the result. One way of putting the model's performance in perspective is by comparing it with other options for classifiers. A random classifier would have an accuracy of 33%. However, that is not very interesting to compare with since the goal is not to be slightly better than a random classifier, but rather to be as good as, or better than the best alternative. Several studies within the machine learning field have compared the performance of machine learning algorithms with human expertise. In some cases, humans achieved a higher accuracy, but in others the machine learning model outperformed humans (Goh et al., 2020) (Dressel and Farid, 2018).

This was found to be an interesting method of evaluating the models performance, and therefore two employees at the company, who are experts in the field, were asked to manually classify 30 users from the test dataset. They were asked to look at the user's bookings, provided in an excel file, and rate on a scale from 0 to 10 how well the user fit into the three personas. The sum across all three personas was supposed to be 10, so that it could be interpreted as probabilities of belonging to specific classes, just like the model's output. The final model, as well as the result of the comparison with experts is presented in chapter 5 Results.

5 Results

This section covers the result of this study. The first part covers the features that were most important for the models and their respective weights in the logistic regression model. The second part includes the performance of the machine learning models on test data and an explanation of the model choice based on the result. The third part contains a comparison of the models performance with human classifiers at Bokadirekt, where two experts at the company were given the task of classifying users. The last part describes how the model was implemented as a final product for the company.

5.1 Features

When the performance of the models had been evaluated with all possible numbers of features it was found that 4 features was the most appropriate number for both models. Both models had an accuracy between 70 and 73 percent when the number of features ranged between 4 and 13. The accuracy varied each run depending on the split of data and showed no consistent better performance for a specific number of features. However, when less than 4 features were used in the models, they performed worse.

In table 2, the four most important features are shown. These were the features that showed the best result during the validation phase and therefore they were used in the final implementation of the model.

Table 2: The four most important features together with description and unit.

Feature name	**Description**	**Unit**
Mean_price	Mean price of all bookings	SEK
Nu_bookings	Total number of bookings	Integer
Median_days_b_a	Median days between booking date and appointment date	Days
Nu_salons	Number of bookings at different salons (booked)	Integer

The decision function for the logistic regression model is represented by three binary classifiers; one for each class. The weights for each feature represents the importance of the feature where a positive weight corresponds to the positive outcome of the class and a negative weight vice versa (Pedregosa et al., 2011). The weights for each class can be seen in table 3. The feature *mean_price* was the least important feature of the four since it had the lowest weights. The three other features had a greater importance for the model as seen in table 3.

Table 3: Feature weights for each class. A positive weight corresponds to a positive outcome (True) and negative weight to a negative outcome (False).

CLASS	**Mean_price**	**Nu_bookings**	**Median_days_b_a**	**Nu_salons**
Regular	0.09	0.12	1.58	-0.31
Browser	0.20	-0.47	0.64	0.9
Impulsive	-0.29	0.35	-2.22	-0.59

The first row in table 3 indicates that the regular class had the most positive correlation with the median number of days between booking and appointment and had a negative correlation with the number of different salons. The second row in table 3 indicates that the browser class had the biggest positive correlation with the number of salons but a negative correlation with the total number of bookings. The last row indicates that the impulsive class had a large negative correlation with the median number of days between booking and appointment but a positive correlation with the total number of bookings.

5.2 Models

The two classification models used the same four features as described in the previous section. Table 4 shows that the logistic regression model yielded an accuracy of 71%, where a random prediction would in theory give an accuracy of 33%. Its performance was equally good on test data as on the training data during the validation phase. The model parameters were trained with cross-entropy loss using the 'lbfgs'-solver for the optimization problem (Pedregosa et al., 2011). The model gave similar results for all classes, but had a little harder time classifying the browser class. The impulsive class had a F1 score of 74%, the highest between the three classes, which means that the model was best at classifying the impulsive persona.

Table 4: Evaluation metrics for the logistic regression model with 'lbfgs'-solver and standardized data.

	Precision	Recall	F1 score	Number of samples	Accuracy
Regular	0.72	0.70	0.71	101	0.71
Browser	0.72	0.61	0.66	117	
Impulsive	0.68	0.80	0.74	117	

The support vector classifier showed a similar result to the logistic regression model with an accuracy of 71% (see table 5). The classifier was trained with the One-vs-One strategy for multi-class problems, and used a 'linear' kernel for the decision boundary (Pedregosa et al., 2011). Table 5 shows no significant difference in performance between classes, but the model had the most difficulty in predicting the browser class and the easiest time with impulsive class.

Table 5: Evaluation metrics for the Support Vector Classifier with linear kernel and standardized data.

	Precision	Recall	F1 score	Number of samples	Accuracy
Regular	0.70	0.69	0.70	101	0.71
Browser	0.70	0.67	0.68	117	
Impulsive	0.73	0.77	0.75	117	

The models' performance, in terms of computational time, was also evaluated by calculating the time it would have taken the model to predict 100 000 samples. For the logistic regression model it was found to be 6 seconds, and for the support vector classifier it was 12 seconds.

5.2.1 Final model choice

Based on the overall performance of the models, a final model was chosen. Both the computational time as well as the evaluation metrics were taken into consideration when deciding on the final model. The results show that the two models performed equally in terms of accuracy, with no major difference in their ability to predict different classes. Both models had a harder time to predict the browser class but showed no difference between them. The logistic regression model was however a bit faster in terms of computational time, as it took 6 seconds to classify 100 000 users whereas the support vector classifier did it in 12 seconds. The computational time was never of great importance however, since the final product does not perform the predictions in real-time. But as the two models performed equally in terms of accuracy, the logistic regression model was chosen since it was faster and provides a natural interpretation of the output as probability estimates for each class.

5.3 Model vs experts

In table 6, 7 and 8, the results of the two experts manual classification of the 30 users is presented, together with the machine learning model's classification results. The 30 users' class distribution was as follows: 9 *regulars*, 10 *browsers* and 11 *impulsives*.

Table 6: Confusion matrix of Expert 1's manual classification

	Precision	Recall	Number of samples	Accuracy
Regular	0.45	1.0	20	0.57
Browser	0.86	0.60	7	
Impulsive	0.67	0.18	3	

Table 7: Confusion matrix of Expert 2's manual classification

	Precision	Recall	Number of samples	Accuracy
Regular	0.35	1.0	26	0.40
Browser	1.0	0.10	1	
Impulsive	0.67	0.18	3	

Table 8: Confusion matrix of the model's classification

	Precision	Recall	Number of samples	Accuracy
Regular	0.78	0.78	9	0.70
Browser	0.75	0.60	8	

Impulsive	0.62	0.73	13	

The results were very interesting, especially that the classification made by the two experts had an accuracy of 40% and 57% respectively, while the model on the other hand had an accuracy of 70%. Noteworthy was also that the experts seemed inclined to classify the regular class, as they both had more than two thirds of the total classifications as that class. This led them to have a perfect (1.0) recall of the regular class, which means that they correctly classified all actual *regulars*. However, they both had very low precision (0.35 and 0.45) as they misclassified a lot of users as *regulars* that actually belonged to other classes. Also worth noting is that the results from the model are based on its output with the largest probability. In other words if the output was [0.40, 0.30, 0.30] for the three classes, the class which had 0.40 probability would be interpreted as the model's class prediction. There was no threshold which the probability had to be greater than to be accepted as a prediction. After inspecting the probability estimates of the model, it was found that 7 of the 30 predictions had no class-probability larger than 0.50, which means that the three probabilities were similar.

5.4 Product implementation

As the end-goal of the project was to develop an application that the company could "plug in" to their current system, the scope of the project was greater than just developing the machine learning model. In order to create an application that could categorize customers based on their booking data, an API for a web server was developed with use of the Flask framework. The web server is currently hosted on a local machine at Bokadirekt, and its services can be used by calling its endpoints, defined by a request–response message system. The API has two endpoints that can be called; one endpoint which takes a user's historical bookings as input and returns a feature vector and another endpoint takes a feature vector as input and returns the probability estimates for each persona. The API was constructed in such a way that an agent could work as a middleman, controlling the communication between the web server and the company's database. The agent would fetch data from the database, make a POST request to the web server through the API and insert/update the database with the response. The communication between the components can be visualized as shown in figure 10.

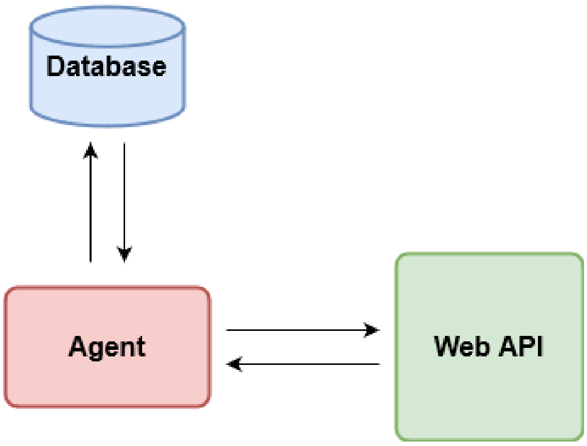
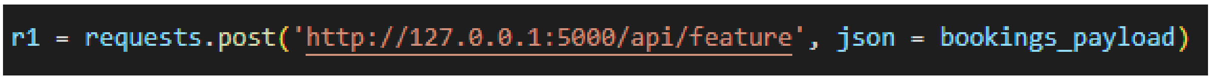


Figure 10: Architecture diagram of the components in the system.

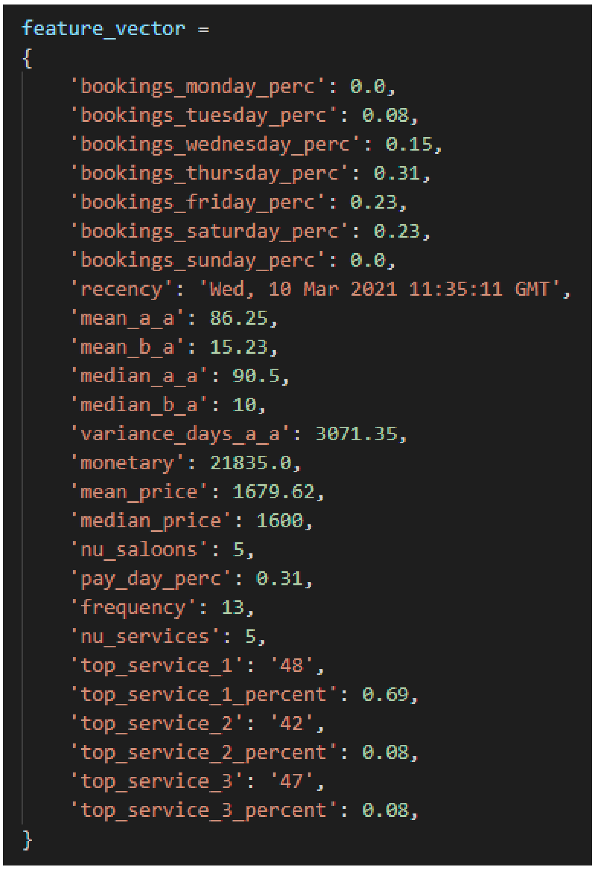
To clarify, the agent was not created by us, but we designed the web API bearing in mind that an agent would control the flow of data.

The first endpoint was created in such a way that an agent program can make a POST request formatted in curl with a user's booking data as a parameter. The web API would then read that data, preprocess it to a feature vector and return it. The features which are returned are described in section 4.4.2 Feature Extraction and shown in figure 12. An example of a curl call to the endpoint is shown in figure 11.



```
r1 = requests.post('http://127.0.0.1:5000/api/feature', json = bookings_payload)
```

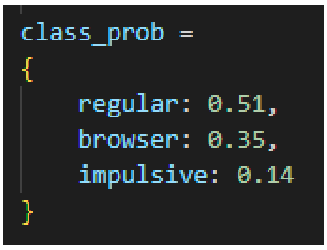
Figure 11. Example of a HTTP POST request to the web server, which contains the booking data in JSON format.



```
feature_vector =
{
    'bookings_monday_perc': 0.0,
    'bookings_tuesday_perc': 0.08,
    'bookings_wednesday_perc': 0.15,
    'bookings_thursday_perc': 0.31,
    'bookings_friday_perc': 0.23,
    'bookings_saturday_perc': 0.23,
    'bookings_sunday_perc': 0.0,
    'recency': 'Wed, 10 Mar 2021 11:35:11 GMT',
    'mean_a_a': 86.25,
    'mean_b_a': 15.23,
    'median_a_a': 90.5,
    'median_b_a': 10,
    'variance_days_a_a': 3071.35,
    'monetary': 21835.0,
    'mean_price': 1679.62,
    'median_price': 1600,
    'nu_saloons': 5,
    'pay_day_perc': 0.31,
    'frequency': 13,
    'nu_services': 5,
    'top_service_1': '48',
    'top_service_1_percent': 0.69,
    'top_service_2': '42',
    'top_service_2_percent': 0.08,
    'top_service_3': '47',
    'top_service_3_percent': 0.08,
}
```

Figure 12: Example of a response from the first endpoint, a JSON object of a feature vector.

For the second endpoint the agent can make a POST request with a user's feature vector as a parameter. The API reads the data, selects the 4 most important features for the classification model, standardizes the data and feeds it as input to the logistic regression model. The model estimates the probability for each persona and returns them as a JSON object, see figure 13.



```
class_prob =
{
    regular: 0.51,
    browser: 0.35,
    impulsive: 0.14
}
```

Figure 13: Example of a response from the second endpoint. A JSON object with the class probabilities.

6 Discussion

In this chapter the result of the project is discussed, and reflections on some strengths and weaknesses in the study are given. First we discuss the features used in the final model, and which of them were most important. Then we move on to the results of the final model and why we chose it, followed by a couple of potential explanations of the results. After that the model's performance compared to the experts is discussed, and we also review the methods used critically before lastly stating the implications of the project.

6.1 Features and weights

The results from this study showed that the classification models worked just as well with only four features as they did with more features, but when the number of features was reduced to less than four, the models' accuracy got worse. This indicates that the four features: *median_days_b_a, nu_bookings, mean_price* and *nu_salons* were the most important for predicting the persona label. The relative magnitude of the coefficients or weights presented in table 3 gives an implication of their importance for each class. The results showed that the regular class had the most positive correlation with *median_days_b_a* and a negative correlation with *nu_salons.* The browser class had a great positive correlation with *nu_salons* and a negative correlation with *nu_bookings*. Lastly, the impulsive class had a positive correlation with *nu_bookings* and three negative correlations of which *median_days_b_a* was the biggest. A plausible explanation for the weights is that some of the criterias that were used in the labelling process have a direct representation in some of the features, for example the number of days booked ahead is represented by *median_days_b_a*.

6.2 Model and results

During the project a lot of different classification algorithms were considered, but in the end a logistic regression model was chosen for the final implementation. There were a number of reasons for this, first of all it achieved an accuracy of 71%, which was equal or better than the other, more complex models managed to achieve. The fact that the model performed equally well on the test data as on the training data indicates that it was not overfitted to the training data and that the model was not too complex for the amount of training data. It was also considerably faster than every other model in terms of computational time although the computational time was not an important factor in this project. In the end the choice came down to either the logistic regression model or the support vector classifier, and since the other factors in the evaluation were very similar, the logistic regression model was chosen because of its low time complexity.

One of the advantages with the final model being a logistic regression model is that it allows for a natural probabilistic interpretation of the output instead of just the predicted class. Having the probabilities allows the company to be more versatile in how they approach the information provided by the model. One way they could potentially use this is by setting a threshold that the class-probability has to be higher than in order for them to take the classification into consideration. Let's say a user is classified with the probabilities [0.45, 0.30, 0.25] of belonging to the different classes, if the threshold is set to 0.50, then this user could be deemed as "unable to classify". With this in place, the company would be sure that the users who actually got classified had shown behaviours that more distinctly match a certain persona. Whether it is better from a business perspective to not be able to classify some users rather than to misclassify them is hard to say, and is something the company will have to decide for themselves.

After inspecting the model's classifications manually it was clear that often the largest class probability was not considerably much larger than the second, which indicates that the classifier was not very sure. There are most likely a number of different reasons behind this, but the main one is that classification of personas is not easy. User behaviours, which is what the personas encapsulate, are not natively divisible but they are much more fittingly described as spectrums and therefore it is not easy to distinctly classify users based on their behaviours. Furthermore, user behaviours might change with time, meaning that a user might show behaviours matching with different personas at different times. For example a specific user might be a *browser* until he or she finds a hairdressing salon they are completely satisfied with, then they might become a *regular* at that salon.

Something interesting that was found in both of the models' results was that *browsers* were more difficult to classify correctly than the other classes. There could be various reasons why this is, but one is likely that in the labelling process, the criterias for the browser class captures behaviours that have no direct representation in the booking data. It might also be that many *browsers* do not make bookings. For example, how many unique salons that have been viewed by a user are not represented in any way in the booking data. However there seems to be some correlation between the number of unique salons viewed and the number of unique salons a user has booked, as that feature: *nu_salons* had the highest weight, which in other words means it was the most important for classifying the browser class. This was both interesting and positive as it meant that the model managed to capture this presumed implicit relationship.

6.3 Model vs experts

When evaluating the model's results in relation to the experts', it is clear that the model is more accurate in pretty much every way possible. If other factors such as scale, time and monetary cost are also taken into consideration, the model's position as the best alternative is further strengthened. First of all, it would be practically impossible for experts to perform classification of several hundreds of thousand users manually. Should it have been possible, it would have taken an incredible amount of time compared to the model which only needs a couple of days. Lastly, the monetary cost of having experts performing manual classification on this scale would be immense, in relation to the model which has a moderate monetary cost connected to the development and thereafter only negligible maintenance cost. The model is in other words a far superior option for this type of classification problem. This has been found by other studies as well (Goh et al., 2020).

One thing to note regarding the comparison between the experts' manual classification versus the model is that the sample size was only 30 users. It is a quite small sample size and therefore it might not be wise to draw too many conclusions from it. Something that indicates that the result is trustworthy however, is that the model's performance was very similar, in fact even a little better, on the much larger test set. The takeaway from this is that the model's performance is fairly certain, at least that it is in the neighborhood of 70%. This also means that it can be established that the model's classification is considerably more accurate than the experts' manual classification.

When looking at the experts manual classification, it is clear that they had a bias towards *regulars*, there could be multiple reasons behind this; it was potentially easier to grasp what the regular persona entails, thus making the experts more prone to choose that class. It could also be that the experts simply had an easier time identifying the behaviours associated with *regulars* in the provided data. The regular class is also the majority class which holds the most samples, which can cause some bias.

Many other studies compare machine learning classification to human experts' classification, in a variety of different fields. Although there is definitely a difference between classifying images of skin lesions (Tschandl et al., 2019) (Brinker et al., 2019) or audio recordings (Zieliński et al., 2020) compared to classifying user personas, it is still interesting to note that other studies also found that machine learning models often perform better than experts in the area.

6.4 Methodology

During this project there have been some methods used that open up for criticism, for example the labelling process was not perfect. If for example a user's bookings history consists of 10 bookings at the same salon that were all booked one day ahead, one might want to say that this user is an *impulsive regular*, as it matches the general conception of what the impulsive and regular personas entail. In the labelling process however, the criterias set up for the regular class does specify that a user should have made bookings at least 7 days ahead a number of times in the last year, which this user has not. Hence it would not have been captured in the filter for the regular class. This is a general issue with the problem itself, to classify users based on their behaviours, as it is very hard to set up criterias that cover all possible cases. These are of course corner-cases which do not occur very often, but when they do certain criterias have a hard time capturing them.

Another drawback, that has to do with the feature extraction method used, is that all bookings made by a user were merged into one feature vector consisting of aggregated values for the different attributes. Transaction data, which the bookings are, has a timestamp associated with it, where the records have an explicit relationship. When combining multiple objects into one, important information is lost and all records are valued as equally important. For example, a user's most recent bookings, and the behaviours connected to them, are probably more important than old records. Hence recent bookings should potentially be weighted higher than older bookings, to get a more accurate classification of which persona a user currently fits into. However, it is possible for the company to choose which records to use when creating the feature vector, thus they can exclude older bookings.

6.5 Implications

The practical implications of the project's end result is immense for the company. First of all having a large portion of their customer base classified as one of three personas is very useful. It allows for further analysis to be made, for example how the different groups contribute to bookings and thereby in generating revenue. It also provides the company with insights about the customer base that is very useful in decision making for example. Other than that it allows them to customize and personalize the website according to the different groups of users, which hopefully improves the customer satisfaction rate. Many of the features created in the process of building and implementing the model are also very useful "on their own" for other purposes such as marketing for example. The marketing team is now able to filter out users that, based on their booking history, are likely to book within 1 week for example. This allows direct marketing to be tailored for each user, which is obviously great for the company, but also for the customers as the marketing will be more relevant.

Another advantage with the final product is that it continuously updates the classification of a user whenever that user makes a new booking. This means that if a user, previously classified as a browser, starts showing behaviours more in line with the regular persona, i.e. finds a salon and a hairdresser they really like and starts going to it regularly, then the model will updates its predictions each time and after a couple of bookings the new prediction will likely be the regular persona. As active users generally make bookings every month, they will be updated very often and hence have an accurate prediction. The more bookings that are made, the more data the model will have to base it's classification on and the more accurate it will likely be. So as time goes by, the model will give the company even more valuable information about the customer base.

7 Conclusions

The purpose of this thesis was to explore the possibilities of using machine learning as a tool for identifying and understanding user behaviour on a booking platform for beauty and health services. In order to achieve the purpose of the thesis, two classification models have been trained to categorize users into different user personas. The performance of the classification models have been analyzed using a number of evaluation metrics and in terms of computational speed. Their performance was also put into perspective, by comparing them with classifications made by human experts at the company.

The results of this study show that it is possible to identify and categorize users into groups represented by a persona, using historical booking data. The logistic regression model had the best overall result and achieved an accuracy of 71% when trained with four features. Its performance was then compared to two human classifiers on a smaller dataset containing the booking data of 30 users. The two human classifiers achieved an accuracy of 57 and 40 percent respectively, whereas the logistic regression model had an accuracy of 70 percent. These findings confirm that the machine learning algorithm outperforms human experts at classifying users into persona groups. Upon inspection of the probability estimates for each class made by the model, it showed that the largest class probability was not considerably much larger than the second in many cases. This indicates that the model was not certain about many of its predictions. We identified the reason for this to be the personas and the labelling process. User behaviour which the personas encapsulate are not black or white, but more fittingly described as spectrums; the samples were not labelled according to an existing "ground truth" but rather from the company's perception of their user groups and their behaviours. We also believe that having classes that are not naturally divisible made it a lot more difficult for the humans to classify samples. As this is an issue with the problem itself, namely to classify users based on their behaviours, we could not find a better solution for this. Classification of something as inexplicit as behaviours will never be perfect, but it is still worth doing as it offers plenty of valuable insights about users.

The new features for each customer together with the classification model can work as a tool for understanding user behaviour and to identify user groups. The knowledge and insight gained about users can be used as support for making decisions and help the company become more data-driven. For example by customizing the website and platform based on the user's persona, or by tailoring the marketing material to specific users.

It is worth noting that the result of the project, like most other classification problems, is tightly connected to the data available. Should a couple of different decisions have been made during the project, the result would likely have been different. If, for example, other criterias would have been used in the process of getting labelled data, the model would have produced other classifications than it currently does. But in the end, the result of the project satisfied the company's needs and the purpose has been fulfilled. The company now has insight about their customers that they did not have before, and a lot of data that can be used to personalize and enhance their platform.

8 Future research

The result of any machine learning project is directly linked to the model and the data at hand. As shown in the result; having more data and features does not always improve a model, but better data and better features certainly do. Since the sample size was limited in this project, more data to train the model could in fact improve the performance; especially if more features are added. Adding more features, or at least further improving the current features could increase the performance of the model. Therefore it is of interest to investigate if more suitable features, describing user behaviour can be aggregated from the datasource. Furthermore, the method we chose: to aggregate the user's booking records to a feature vector has the drawback of losing information (Tan et al., 2014, pp. 53). Therefore, it could be of interest to explore other methods where the timestamp and relationship between records is not lost.

Both the model and the human classifiers had the most difficulty in classifying the browser class. As mentioned in the discussion, the reason for this could be that the booking data does not directly capture the behaviour on the website which very much defines the browser persona. It is therefore interesting to investigate whether an additional datasource can complement the booking data, with information about website behaviour prior to the booking for example.

Another topic open to improvement is the user personas; how they were defined and labelled. It could be useful to have more information about these user groups, with detailed descriptions and user stories for example (Miaskiewicz and Kozar, 2011). More comprehensive knowledge about the users and better defined personas could simplify and improve the labelling process.

It could also be of interest to explore if unsupervised machine learning methods could be useful for this case, as several other studies on customer segmentation have gotten great results with clustering methods (Zakrzewska and Murlewski, 2005) (Wu and Chou, 2011) (Selini Hadjidimitriou et al., 2017).

References

1. Alemany Oliver, M. & Vayre, J. (2015) “Big data and the future of knowledge production in marketing research: Ethics, digital traces, and abductive reasoning”. Journal of Marketing Analytics, vol. 3, pp. 5–13.
2. Amazon Personalize, Available at: https://aws.amazon.com/personalize/ (viewed 11-05-2021)
3. Amplitude, “The Complete User Analysis Guide: Personas to Design to Analytics”. Available at: https://amplitude.com/user-analysis (viewed 11-05-2021).
4. Band, A. (2020), Multi-class Classification - One-vs-All & One-vs-One. Available at: https://towardsdatascience.com/multi-class-classification-one-vs-all-one-vs-one-94daed32a87b (viewed 2021-03-23)
5. Bokadirekt (2020), Available at: https://blogg.bokadirekt.se/om-bokadirekt/ (viewed 2021-04-14)
6. Brinker, T.J. et al. (2019), "Deep learning outperformed 136 of 157 dermatologists in a head-to-head dermoscopic melanoma image classification task", European journal of cancer, vol. 113, pp. 47-54.
7. Brownlow, J., Zaki, M., Neely, A. & Urmetzer, F. (2015), “Data-Driven Business Models: A Blueprint for Innovation”, Cambridge Service Alliance.
8. Chu, X. and Ilyas, I. (2019), Data Cleaning. 1st Ed. ACM books: New York
9. Data reportal (2021), Digital 2021 Global Overview report. Available at: https://datareportal.com/reports/digital-2021-global-overview-report (viewed 11-05-2021).
10. Dressel, J. & Farid, H. (2018), "The accuracy, fairness, and limits of predicting recidivism", Science advances, vol. 4, no. 1.
11. Fawcett, T. (2006), "An Introduction to ROC Analysis" (PDF). Pattern Recognition Letters. vol. 27 no. 8, pp. 861–874.
12. García, S., Luengo, J., Herrera, F. & SpringerLink (Online service) (2015), Data Preprocessing in Data Mining. Springer International Publishing, Cham.
13. Geron, A. (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 2nd Ed, O'Reilly Media: Cambridge

14. Goh, Y.C., Cai, X.Q., Theseira, W., Ko, G. & Khor, K.A. (2020), "Evaluating human versus machine learning performance in classifying research abstracts", Scientometrics, vol. 125, no. 2, pp. 1197-16.
15. IBM (2020), Machine learning. Available at: https://www.ibm.com/cloud/learn/machine-learning (viewed 2021-03-18)
16. James, G., Witten, D., Hastie, T. & Tibshirani, R. (2017), An introduction to statistical learning: with applications in R. Springer: New York.
17. Kotras, B. (2020), "Mass personalization: Predictive marketing algorithms and the reshaping of consumer knowledge", Big data & society, vol. 7, no. 2.
18. Larose, D.T. (2005), Discovering knowledge in data: an introduction to data mining. Wiley-Interscience: Hoboken.
19. Lindholm, A., Wahlström, N., Lindsten, F., Schön, T., (2021), Machine Learning - A First Course for Engineers and Scientists.
20. López, V., Fernández, A., García, S., Palade, V. & Herrera, F. (2013), "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics", Information sciences, vol. 250, pp. 113-141.
21. Mayer-Schönberger, V. & Cukier, K. (2013), Big data: a revolution that will transform how we live, work, and think. John Murray Publishers: London.
22. Miaskiewicz, T. & Kozar, K.A. (2011), "Personas and user-centered design: How can personas benefit product design processes?", Design studies*,* vol. 32, no. 5, pp. 417-430.
23. Mishra, S., Mallick, P.K., Jena, L. & Chae, G. (2020), "Optimization of Skewed Data Using Sampling-Based Preprocessing Approach", Frontiers in public health, vol. 8, pp. 274-274.
24. Mobasher, B. & Anand, S.S. (2005), Intelligent techniques for Web personalization. Springer: New York.
25. Mohri, M., Rostamizadeh, A. & Talwalkar, A. (2012), Foundations of machine learning. MIT Press: Cambridge.
26. Mutuvi, S. (2019), Introduction to Machine Learning Model Evaluation. Available at: https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f (viewed 2021-03-27)
27. Nationalencyklopedin, Maskininlärning. Available at: http://www.ne.se/uppslagsverk/encyklopedi/lång/maskininlärning (viewed 2021-03-18)

28. Nelli, F. (2018), Python Data Analytics: With Pandas, NumPy, and Matplotlib. 2nd Ed, Apress: Berkeley.
29. Netflix Research, Available at: https://research.netflix.com/business-area/personalization-and-search (viewed 11-05-2021)
30. Ng, S.S.Y., Tse, P.W. & Tsui, K.L. (2014), "A one-versus-all class binarization strategy for bearing diagnostics of concurrent defects", Sensors (Basel, Switzerland), vol. 14, no. 1, pp. 1295-1321.
31. Osborne, J.W. (2012), "Logits and tigers and bears, Oh my! A brief look at the simple math of logistic regression and how it can improve dissemination of results", Practical assessment, research & evaluation, vol. 17, no. 11, pp. 1-10.
32. Pedregosa, F. et al. (2011), Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830.
33. Roman, V. (2019), Unsupervised machine learning clustering analysis. Available at: https://towardsdatascience.com/unsupervised-machine-learning-clustering-analysis-d40f2b34ae7e
34. Selini Hadjidimitriou, N., Mamei, M., Dell'Amico, M. & Kaparias, I. (2017), “Classification of Livebus arrivals user behavior”, Journal of Intelligent Transportation Systems, vol. 21, no. 5, pp. 375-389,
35. Tan, P., Steinbach, M., Karpatne, A. & Kumar, V. (2014), Introduction to data mining. 1st Ed, Pearson: Harlow.
36. Theodoridis, S., Koutroumbas, K., (2009), Pattern Recognition. 4th Ed, Academic Press: Cambridge.
37. Tolles, J. & Meurer, W.J. (2016), "Logistic Regression: Relating Patient Characteristics to Outcomes", JAMA : the journal of the American Medical Association, vol. 316, no. 5, pp. 533-534.
38. Tschandl, P. et al. (2019),“Comparison of the accuracy of human readers versus machine-learning algorithms for pigmented skin lesion classification: an open, web-based, international, diagnostic study”. The Lancet Oncology, vol. 20, no. 7, pp. 938-947.
39. Wu, R. & Chou, P. (2011), "Customer segmentation of multiple category data in e-commerce using a soft-clustering approach", Electronic commerce research and applications, vol. 10, no. 3, pp. 331-341.

40. Zakrzewska, D. and Murlewski,J.(2005), "Clustering algorithms for bank customer segmentation," 5th International Conference on Intelligent Systems Design and Applications (ISDA'05), pp. 197-202.

41. Zieliński, S.K., Lee, H., Antoniuk, P. & Dadan, O. (2020), "A Comparison of Human against Machine-Classification of Spatial Audio Scenes in Binaural Recordings of Music", Applied sciences, vol. 10, no. 17, pp. 5956.