JOHANNES KEPLER
UNIVERSITY LINZ

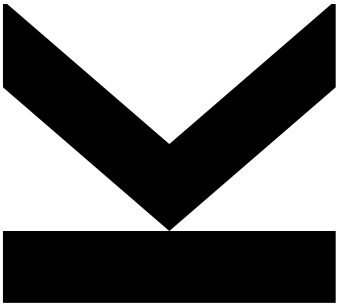
Submitted by
Jürgen Ratzenböck

Submitted at
Institut für Telekooperation

Supervisor
Univ. Prof. Mag. Dr. Gabriele Anderst-Kotsis

January 2018

Using Behavioral Analytics to reason about Customer Satisfaction in data-intensive Software systems



Master Thesis
to obtain the academic degree of
Diplom-Ingenieur
in the Master's Program
Computer Science

JOHANNES KEPLER
UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

Affidavit

I hereby declare that the following dissertation "Using Behavioral Analytics to reason about Customer Satisfaction in data-intensive Software systems" has been written only by the undersigned and without any assistance from third parties.

Furthermore, I confirm that no sources have been used in the preparation of this thesis other than those indicated in the thesis itself. This printed thesis is identical with the electronic version submitted.

Linz, on February 1, 2018

Jürgen Ratzenböck

Acknowledgment

Hereby I would like to thank my supervisor Univ.-Prof. Mag. Dr. Gabriele Anderst-Kotsis for her support throughout the whole work on this thesis. She always provided me useful feedback and used her professional skills and knowledge about this topic to give me hints and advices when I needed it.

Abstract

Customer satisfaction is a key criteria for modern businesses as it is an important ingredient for establishing a long-term relationship with a customer. Unfortunately, conducting a survey is only a temporary solution as the satisfaction level is very volatile.

This thesis tried to tackle the existing issue by researching a methodology with which it should be possible to reason about customer satisfaction automatically using provided data from a large software system. Based on existing literature about customer satisfaction models, the researcher investigated relevant data sources with supposed influence on the satisfaction outcome of customers. The work presents the cornerstones of the implemented data extraction tool to bring data into the desired format for analysis. It then outlines the two different approaches applied on the extracted data to derive desired knowledge about influential factors and predict whether a customer tends to be (dis)satisfied.

The first approach starts based on predefined business critical hypotheses with regard to customer satisfaction in order to find relationships within the data and thus filter satisfaction critical attributes. The thesis exemplarily illustrates the analysis with certain statistical tests for verifying respectively falsifying the hypotheses.

The second approach presents a knowledge discovery process applied on customer feedback data gathered from a satisfaction survey. A framework implementing researched data mining algorithms will be outlined step-wise.

Finally, it could be shown that an optimally configured Random forest classifier is able to classify about 75% of customers correctly as either dissatisfied or satisfied. Hence, it clearly outperformed the support vector machine classifier. The evaluation emphasizes that customer satisfaction is not yet well represented in collected data and thus the research reveals room for improvement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This first chapter sets the context of this thesis by outlining a shift in business strategies over the last decades pointing out the importance of customers as key figure in this development. Based on these findings the actual problem to be solved as well as the research objectives will be defined.

1.1 Problem statement

Until the 1980s companies were only focused on the quality of their products and services. Demand at this time was usually high and, due to little competition on the market, customers discovered the product they wanted to have more easily. Companies focused on optimizing product quality based on internal standards. An important task was to collect and monitor internal data about quality developments in the product life cycle to assess the improvement of the internal processes over time. However, increasing supply and resulting competition on the market caused companies to shift towards a more customer centric approach starting in the early 1990s. As online services and products have become more popular and successful over the next few years, the dramatic growth of oversupply and market intensity required companies to start building their whole business around customer needs and wants to stay profitable in the long-term. [53]. [12] for instance showed that an 5% increase of customer retention has a positive impact of 25 to 125% on the profit.

In such a competitive market, customers rarely stay at the same service provider due to convenience only. Switching to a competitor has never been easier [57] as it is in most part of the software industry. However, for businesses it gets harder to attract and recruit new customers which led companies to treat a purchase of a new customer as the start of a long-term relationship. Keeping customers over longer periods turns out to be a major business issue nowadays [49] [53]. Attracting and recruiting a new customer

instead requires much more costs than selling to an existing faithful customer [6]. There has been a lot of research going on for many years with the goal to reason about the main drivers and their effect on customer retention, whereby Customer Satisfaction has been identified as a central factor as proved by large-scale analysis in the industry [25] [11] [28]. Although there has been an ongoing debate on the strength of this effect, software businesses are subject of a heteroegenous industry and thus more affected by the satisfaction of their customers in contrast to homegeneous industries [25]. As a result of these findings, competitive software businesses have to find a way to treat their customers carefully and ensure their happiness over a longer period.

Fortunately technology in the software industry has evolved dramatically. The Web 2.0 capabilities allow to collect huge amounts of customer data while computational power of hardware enables businesses to store every single touch point of customers in large-scale database systems and analyze them efficiently [17] [53]. Data mining techniques allow to derive patterns based on customer data turned into comprehensive knowledge and moreover provide opportunities to predict future events depending on them. Despite the massive ongoing research in the field of data- anlaysis and mining related to customer knowledge, automatic reasoning about Customer Satisfaction with regard to an Internet business lacks major research success. Although businesses nowadays leverage a huge amount of customer data, invest a lot of resources to treat their customers as personal as possible and usually employ a first-class customer support service to quickly tackle problems and complaints, this thesis sees room for improvement in assessing satisfaction of arbitrary customers pro-actively and automatically [53].

1.2 Research objectives

The research question of this thesis is about finding out how to leverage useful data related to the behavior of customers and their interaction with services to derive patterns which allow to make a statement on how (dis)satisfied a customer is. Furthermore the results should show which data have major influence on customers satisfactory level and thus outline the essential product characteristics a service provider has to pay attention to and optimize to be successful in the future. If the results can be considered as reliable and valid, they can be used for early detection of unsatisfied customers and help to pro-actively solve their problems instead of handling their problem in a reactive manner. The route towards the goal of the research looks the following:

- Define what customer satisfaction means and how to differentiate between satisfactory levels.

- Elaborate which data has potential to influence customer satisfaction regarding the case study example used in this research.
- Find a way to represent Customer Satisfaction as reliable and measurable metric.
- Implement statistical tests to find out which metrics in the data affect Customer Satisfaction.
- Implement a software solution to analyze behavior of customers, learn from this data and do a predictive analysis of how (dis)satisfied an arbitrary customer will be.

1.3 Case Study: Tractive

As illustrative example the research relies on the data of a pet tracking company named Tractive GmbH. The company which was founded as a startup in 2012 in Pasching, Upper Austria produces hardware devices and software applications for different kinds of pets. Their most popular product is a GPS (Global Positioning System) device which can be put on a pet and allows the customer to track its position live on a mobile phone or via a website. This way Tractive helps pet owners worldwide not to lose their pets again since they can always have a eye on them via a smartphone or desktop computer. Since the company is quite successful with about 80k paying customers using a Tractive GPS device, there is also a lot of data available related to each customer. This starts with data related to the usage of the hardware device ranging to subscription data of a customer and his/her interaction with customer support service. Since the company is built up on a subscription model which requires customers to pay on a regular basis for the usage of the device, there is special interest in binding customers for a long time to the company. In essence, this means that satisfied customers are a key asset for the company. Therefore its model should fit well to the research objectives and promising results could support solving customer issues earlier and as a result affect drop out rate positively.

1.4 Structure of the Thesis

The thesis will contribute to a more efficient way of measuring customer satisfaction using large sets of collected data related to usage behavior. Chapter 1 first gives an introduction about the general problem context and outlines the research objectives as

well as the case study this thesis is based on. Chapter 2 will take a closer look at related research work. It will define the term customer satisfaction and its limitations. It gives an overview on existing approaches to measure customer satisfaction and reasons about their efficiency, problems and improvement potential. Furthermore this chapter outlines necessary background research done to gain a deeper understanding on what it means to reason about satisfaction and sheds some light onto different methods suitable for the implementation part. Following, chapter 3 outlines the process of selecting relevant data sources to be used for customer satisfaction determination. Subsequently the chapter illustrates statistical analysis methods applied on this data to reason about relationships of particular data collections and attributes. Chapter 4 illustrates an approach of learning from explicit customer feedback in order to predict satisfaction of arbitrary customers based on collected data about them. Chapter 5 evaluates the outcome from this implementation and compares the approaches regarding their performance results retrieved. Finally, chapter 6 concludes the work by summarizing the important findings and a short look into future work.

Chapter 2

Existing Approaches in Behavior Analysis and Detection of Customer Satisfaction

As the cornerstones of the research objectives have been defined, the goal is to shed more light onto the key point this thesis deals with, namely customer satisfaction. The following chapter defines the term customer satisfaction and outlines which properties are paid attention to in this thesis. Moreover measurement opportunities and existing approaches related to analysis of usage behavior and measurement of customer satisfaction, with focus on their relevance regarding this thesis, will be discussed in detail.

2.1 Definition and Limitations of Customer Satisfaction

Before evaluating approaches to determine customer satisfaction via collected knowledge data, the term should be properly understood. Research papers regarding the definition of customer satisfaction were already published in the 70s. Since then different approaches viewing the topic from slightly various angles have been proposed. The principal statement, namely that customer satisfaction is based on the comparison between individual expectations of the customer and perceived performance of the product after usage was the common outcome of research experiments at these times [52] [4]. If the expectations were lower or equal, the customer can be considered as satisfied whereas vice versa the customer is dissatisfied since his expectations were not fulfilled. In case the expectations do not match with the actual performance of the product this is also named positive- respectively negative disconfirmation while a match is described as confirmation [52] [4].

As a result, when imagining it in a more mathematical way customer satisfaction can be represented as a simple function with two input parameters. One the one hand, the expectations of the customer are based on all information gathered before using the product. It is an subjectively built ideal picture of the product depending on advertisement, word-of-mouth critics and company- respectively brand reputation [36] [53]. On the other hand the perceived performance of the product when using it comprises the whole consumption experience of the customer. As main driver the overall provided quality of the product was identified. This metric not only covers the pure quality, in terms of whether the product reliably performs as advertised, but more exactly specifies the actual quality for the price. Thus, when talking about drivers of customer satisfaction it is often described as perceived value [36] [25]. Along with the shift towards importance of customer retention, customer satisfaction models developed from a transaction tied view to a cumulative view which considers a customers experience with a product over a longer time period. This view more specifically includes the fact that customer expectations change over time and become influential for the perceived quality as well. [37]. It became clear that the customer satisfaction construct is quite involved and especially some of the antecedents and consequences remained unclear.

The Swedish Customer Satisfaction Barometer (SCSB) was a big research project in investigating customer satisfaction across several industries within a whole country. It used a model based on the two identified input parameters from previous research. The positive result of customer satisfaction was already anticipated by [11] [28]. The outcome of dissatisfaction was based on the theory of [34], namely that customers are more likely to complain for problems they might have. Furthermore the model stated that positive resolving of complaints supports the establishment of loyalty against the company. A sketch of this model is shown in figure 2.1.

Although the model experienced some critics in the following years as further national customer satisfaction surveys were deployed they all were based on and inspired by the original ideas of the swedish model. A more enhanced analysis of the existing model approaches was done by [36]. As a result of the derived strengths and weaknesses from different customer satisfaction barometer models, they proposed a new model which was used as the base model by this thesis to build its further empirical work on. The model is visualized in figure 2.2. Each of the different parts will be described briefly to get an overview on the essential components and their relationships and adaptations in contrast to the initial base model, namely the SCSB, as well as the limitations drawn by the author regarding the work in this thesis.

The first difference when comparing the latter proposed model with the SCSB is the role of customer expectations. Since satisfaction has to be observed over a longer time period, expectations get adapted throughout repeated consumption experience as well.

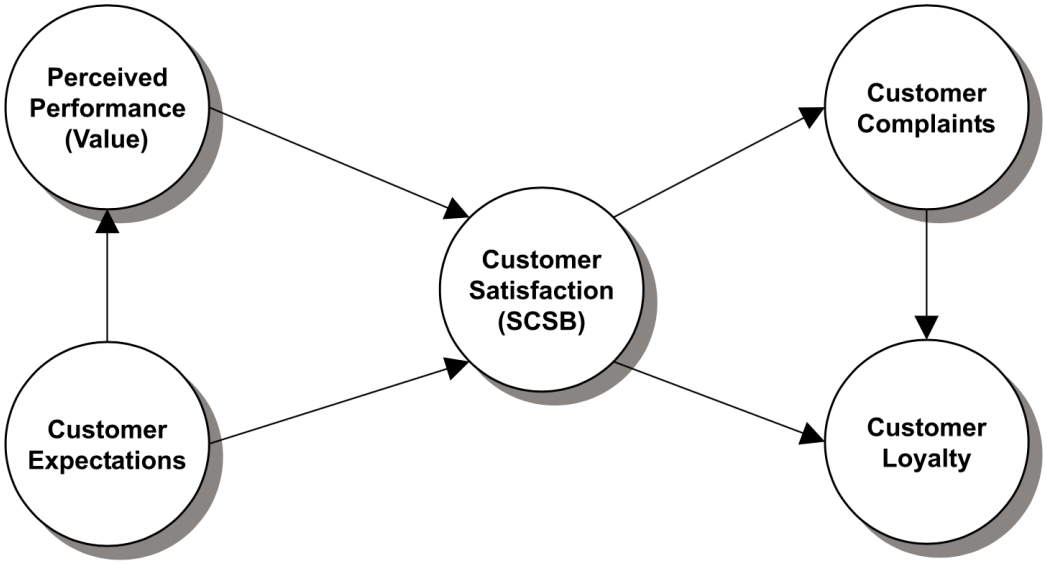


Figure 2.1: Model used for Swedish Customer Satisfaction barometer [25]

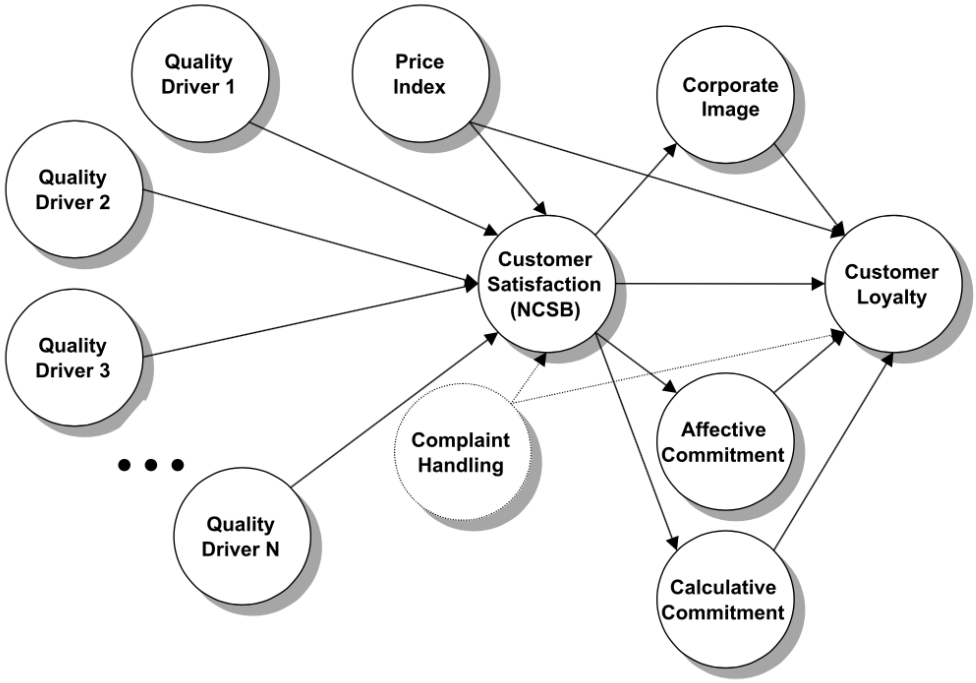


Figure 2.2: Customer satisfaction model proposed by [36]

More emphasis is put on the quality delivered by the product and the expectations are considered to be evaluated regularly due to this quality. These statements can be supported among several industries including the communications sector which the author considers as similar to the field the case study of this thesis operates in [37] [26]. Despite that fact the author does not completely agree with these findings to remove the term "customer expectations" totally. Instead of the expectation parameter, 1-n perceived quality criterion are predictors for customer satisfaction. Next to the quality drivers the perceived price for the provided product quality plays a role as well. The third satisfaction driver illustrated in this model is complaint handling. Due to evolvement of customer support services, customer complaints cannot be stated as simple negative outcome of customer satisfaction. More specifically, support influences emotions either negatively or positively and as a consequence is a direct predictor of satisfaction. It may even has an impact on repurchase behavior of customers which explains the direct relationship to customer loyalty [36].

The three remaining components have not yet been mentioned explicitly regarding their role in the customer satisfaction construct. All of them are directly influenced by the outcome of customer satisfaction while mediating the loyalty parameter. The corporate image, or sometimes called brand reputation, indicates how popular and known a company is in the public. Consumption experiences of a customer affects the corporate image in a certain direction as well. According to a study of [35] a dissatisfied customer tells on average nine other people about negative experiences which can heavily affect loyalty. Although the corporate image is part of the model it will not be discussed or used further in the empirical work of this thesis because it can not be measured by usage behavior and quality data collected by IT systems. Similarly the affective- and calculative commitment mediate parts of the effect of customer satisfaction on loyalty. While the affective component resides on the emotional side and mainly relates to trust with regard to a brand, the calculative component is more rational in terms of costs. (e.g.: costs for a customer to cancel a relationship and switch to a competitor) [36] Both components were mentioned for the purpose of completeness but are not elaborated in more detail since they cannot be represented in the available data of the case study.

2.2 Measurement of Customer Satisfaction

As the reader got an overview on the parts and influential factors of customer satisfaction, the next step is to look at possibilities to transform the defined model properties to quantitative measurements. On the one hand this thesis evaluated survey based measurements and looked in more detail on different types and their performance. On the other hand it was tried to find approaches to measure customer satisfaction without survey data based on implicit satisfaction representing patterns in huge data sets.

2.2.1 Theories about Customer Satisfaction determination

Although there has been a lot of research regarding a formal definition of customer satisfaction, less papers were published on how satisfaction can be quantitatively measured. One approach based on the original expectancy-performance disconfirmation model was to design a survey asking customers how they would rate

- their expectations regarding to a set of chosen dimensions before usage and
- the perceived performance and quality of the product with regard to the same set of dimensions [54].

This approach seems to be promising as it respects the originally identified and well researched customer satisfaction model. However, it brings problems regarding the expectation parameter with it and thus make it inappropriate for this thesis. Firstly, expectations can hardly be measured unbiased since they get evaluated by the customer alongside with the perceived quality in case of using one satisfaction survey [27]. As outlined in 2.1 the proposed model is based on cumulative satisfaction evaluation which yields regular adaptations on what a customer expect from a product. Asking customers about their expectations, with regard to a specific set of product dimensions, explicitly before usage neither makes any sense from a business perspective of Tractive nor is the author able to initiate a process to collect such data. Secondly, measuring customer expectations turns out to be difficult as people tend to rate their expectations, if they have to state them explicitly, very high on average [7]. As a consequence the uniform overrating of customers expectations will not provide any extra information in assessing customer satisfaction from a survey. As last point in the considerations regarding this approach, the validity of stated expectations has to be questioned since customers often do not have specific expectations before using a product. A dominant reason for this phenomenon is especially observed in software products or services due to its variable and intangible nature. Following, it was found that if customer expectations cannot be formed well, the expectancy-performance disconfirmation metric looses its worth [30].

The mentioned arguments and following impracticality required the need for further research on suitable approaches to assess customer satisfaction. In contrast to the research branch which insists on the importance of considering expectations in measurement, there has also been research experience in favoring the perceived performance or, depending on the customer satisfaction model, the perceived value [63]. Based on the mentioned disadvantages in assessing expectations it is claimed that perceived performance reflects the subjective response and emotional feeling as outcome of a consumption experience and thus is seen as the major source in steering the satisfaction

value into a particular direction [30] [19]. Although perceived performance alone is claimed to perform well as measurement indicator for customer satisfaction, a lot of different data dimensions tie together an overall quality or performance. Depending on the dataset the collected parts can correlate in a certain way, be independent, redundant or differ by their importance in contributing to the overall satisfaction. Thus, taking the importance of quality indicators into account can be crucial in determining why a customer might be more satisfied than a similar one [8]. Based on these findings and results in past research this thesis follows, with regard to the case study, the hypothesis that collected usage behavior data is especially of interest if related to quality and performance of the product respectively offered service. With a view to the empirical work, the available data at Tractive provides enough quality and performance indicators. This data is collected transactional-oriented with the customers usage of the Tractive GPS products and can be used when following this measurement approach.

2.2.2 Quantitative metrics for classification of satisfaction outcome

Before being able to elaborate on the taxonomy of statistical and data-mining related methods, the open question, on how a Customers satisfaction feeling can be measured reliably by quantitative metrics, has to be answered.

In order to be able to reveal reliable patterns and influencing factors in quality and performance data when executing analysis methods, some sample satisfaction data from customers is needed. Otherwise no educated comparisons of results are possible and they would only be wild guesses. Due to the complexity of building a customers attitude towards a product, the preferred way is to collect quantitative data representing satisfaction by a survey [63] [31]. Especially the software industry is a sector where it is hard to classify a customers satisfaction based on objective data implicitly gathered by the systems without asking any person [31]. However, conducting a customer satisfaction survey turns to be a difficult task. On the one hand it should support later analysis by enough sample data, while on the other hand it should also collect enough useful satisfaction measurements from customers to derive expressive results [58].

There have been different opinions in research on how a customer satisfaction survey should be structured and how comprehensive the content has to be. [31] proposed a process where all quality dimensions of a product, considered as important by the survey issuer, should be collected since this promises to provide a complete view on the whole product separated by its characteristics. The author of this thesis sees it differently because the research objective is to reason about an overall satisfaction of customers without knowing any subjective opinion from them. Plus, response rates will decrease the more items a survey contains. The essential priorities for the researcher

is first, collecting many survey answers to have a bigger chance in finding significant patterns and learn more reliably from the data and second, keeping understandability of results rather simple [58]. In dependence on the scarce resources of businesses in very competitive markets, as the software industry for instance, approaches and outcomes of research developed into the direction of making surveys much shorter but with wisely chosen questions which are useful to reason about loyalty of customers which in turn is strongly connected to profit [56]. When designing a customer satisfaction survey to collect sample data for the analysis part, these latter ideas were taken under consideration.

2.3 Taxonomy of methods for behavioral analysis and satisfaction prediction

The research work in finding and evaluating different analysis methods and prediction approaches is described in this section. The following paragraphs will outline existing process models and approaches which were either already tried and evaluated in the extended field of customer satisfaction or may be otherwise relevant and suitable for the implementation part of this thesis.

2.3.1 Overview on an analysis approach

Before thinking about suitable approaches and their implementation, it is important to clearly state the goal of the analysis and data planned to use in order to reach the goal. The analysis data contains the information necessary to perform the concrete analysis and should support reaching the desired goal. In this generic picture, it does not matter how this data got collected, its representation and how it can technically be extracted and queried. In first place it is important to ensure that this data is available and relevant. On the opposite end of the process one has to formulate the analysis goal which guides the procedure and always has to be kept in mind as the ultimate question which should be answered. Thus, the analysis goal is limited to yet unknown properties and information which cannot be directly queried from available data. Specifying analysis- data and goal yields an analysis problem to be solved [53] [40]. The process is illustrated in figure 2.3.

When having specified an analysis problem, the hard part of solving it with a suitable analysis approach is still pending. It is not limited to the analysis algorithms itself but rather involves a whole process of selecting, preparing- and finally analyzing the data in a way to satisfy the formulated goal.

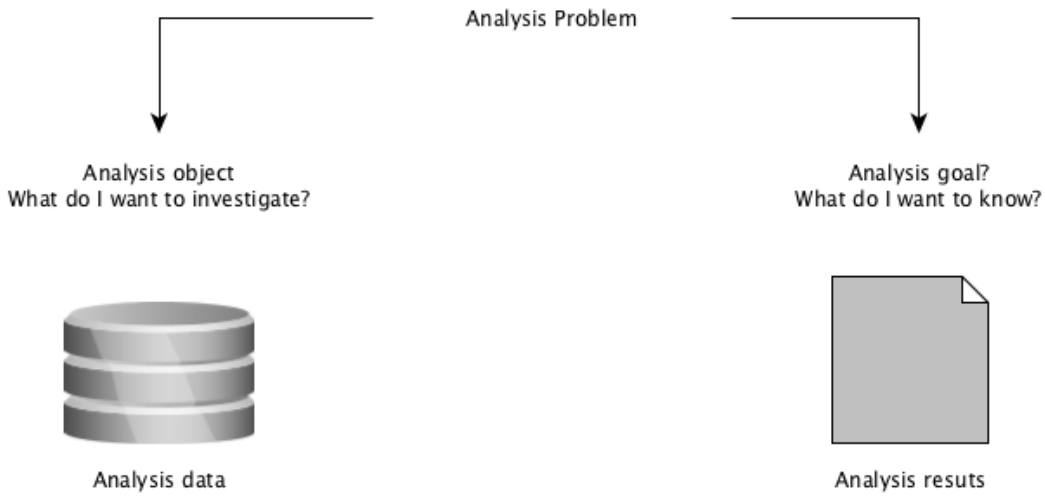


Figure 2.3: Analysis process [53]

Basically, when analyzing customer related data and its effect on a business, it has to be distinguished between two major analysis categories, namely the top-down (hypotheses-driven)- and the bottom-up (data-driven) approach [53]. As this thesis followed these two approaches on the way to answer the stated research question, the upcoming sections will shed more light onto them and their appliance in existing research projects.

2.3.2 Top-down analysis approach

The objective of a top-down analysis is to get an overview on the initially selected data and be able to reason about the influence of particular attributes. In order to perform a top-down analysis it is required to have a better knowledge of the data to be analyzed. The reason is that this type of analysis is based on an explicitly formulated hypothesis which should be proved and as a result verified or falsified. In essence, such an analysis usually starts with well-defined assumptions. In terms of customer satisfaction analysis, a domain expert supposes influential factors and knows which data is representative enough. However, it is unknown whether he or she is right at all, how strong such a particular relationship in reality is and which direction the potential predictor correlates with the target variable. Those are then the resulting questions stated explicitly with hypotheses and to be answered by a top-down analysis approach. Due to the nature, it heavily relies on statistical methods to prove the formulated hypotheses. This involves different kinds of statistical tests depending on the type of data required to execute the analysis [53]. Published research resources about top-down analysis in the field of

customer satisfaction are rare. The main purpose turned out to be inductive statistical analysis.

2.3.2.1 Suitable statistical tests

The main purpose of a statistical analysis in the top-down approach is making a statement about the behavior of a population based on a representative sample. Such an analysis is also called Inductive statistics. Since it cannot be known upfront how well the selected sample actually represents the population, the probability of a particular result with regard to prediction of the actual outcome gets quantified. This is usually done via a confidence interval, indicating an allowed error α within a hypothesis is proven to be valid [51]. Depending on the type of data, an appropriate statistical test should be chosen. Regarding the empirical work in this thesis, two types of sample data will be relevant in the top-down approach, namely categorical- and continuous numerical data.

As the categorical data to be considered in the very first analysis will only be of binary type, the Fisher's exact test turned out to be the best choice to analyze a potential statistical significant difference between the two categorical groups under consideration. Details about this test can be found in [59].

In order to compare data rows consisting of numerical values which do not have an order to allow for directly ranking them, the popular Pearson correlation coefficient was leveraged by this thesis. It is a quite powerful solution to analyze a linear relationship between numerical data rows and also indicates its direction and strength. Although this method has the disadvantage that it cannot reason about any non-linear relationship, the author expected that the correlation behaves in a linear manner in order to identify drivers of customer satisfaction. Since the correlation coefficient is only a point estimation, there is need for an inductive measurement to reason about the population. Under the assumption that both data rows are at least asymptotic normal distributed, there is the possibility to calculate a confidence interval for the correlation coefficient and in turn execute a statistical test to verify a hypothesis. This test is called Pearson's product moment correlation coefficient. If the value confidence interval does not include the value 0.0, a statistical significant result can be followed [42] [5].

2.3.2.2 Selecting data samples

Despite the fact that the scope of data to be analyzed is specified by a given hypothesis, statistics were found to react sensitively with regard to sample sizes and quality of

data. While the representativeness of a selected sample is left to the particular use case, interesting research results were proposed with regard to the sample size to choose. One has to keep in mind that the sample size can heavily influence the power of analysis methods as a statistical significance alone is not expressive enough. One also has to put emphasize on the so called clinical relevance which originates from inductive statistical analysis in the medical field [13]. This relevance value defines the minimum effect which should be detected as for instance in case of categorical data between the control and treatment group [38]. In case of correlation analysis, this would be the minimum coefficient considered as relevant for indicating a sufficient linear relationship [47]. It is important to notice that in order to prove for a smaller effect size, which is equal to a small confidence interval, the sample size has to be larger. In contrast, a large effect size, which equals a bigger confidence interval, reduces the required sample size. In addition to the minimum effect size, the α-error, indicating the maximum allowed false-positive rate and the β-error, indicating the false-negative rate, are necessary input parameters to estimate an optimal sample size [38].

2.3.3 Bottom-up analysis approach

In contrast to the top-down analysis approach, the bottom-up (also known as data-driven approach) does not involve explicit hypotheses to prove. Instead, the questions of a data analyst in this case start very general and initial assumptions are vague. Researchers for instance can use the stated research question as initial analysis goal. As a result, these characteristics make the whole analysis approach much more complex as there are no upfront decisions made about specific datasets to be used in order to answer the question respectively to solve the analysis goal. This type of analysis can also be interpreted as searching in a huge data jungle where several intermediate analysis problems have to be solved before being able to solve the original analysis problem. After solving such an intermediate challenge, one gathers more insights in the data and new patterns are revealed over time [53] [24]. Research in this are has become highly competitive and different buzzwords started to raise attention to a broad audience. To prevent confusion in the upcoming sections, the author decided to briefly explain the essential terms. Following listing names the terms as they will be used in the rest of this thesis. The interpretations are based on the findings from [24].

- Knowledge Discovery in Databases (KDD): In abstract terms KDD can be seen as the comprehensive process to gather some useful knowledge from collected data in a database. With regard to the prediction of customer satisfaction, it can be interpreted as the procedure starting with processing of all customer related data till building a model which allows predicting whether a customer is (dis)satisfied.

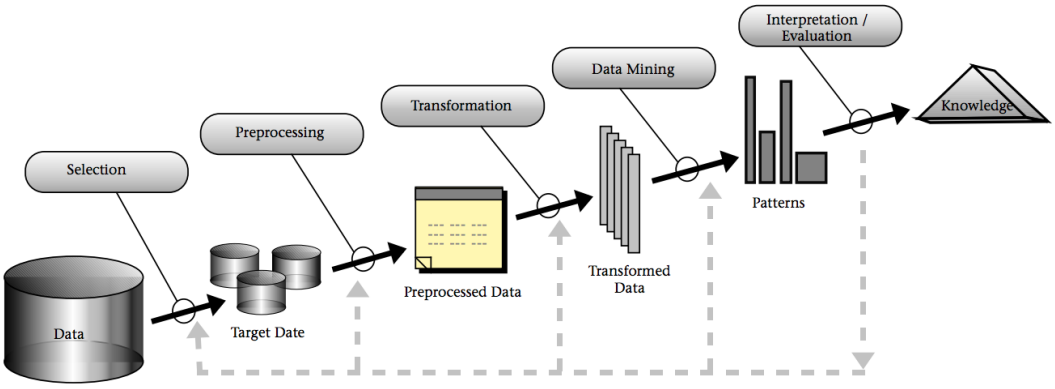


Figure 2.4: Big picture of a KDD process [24]

- Data mining: This involves the actual algorithms to find and extract the desired information from the prepared data. In essence, it is merely one step in the knowledge discovery process, namely the prediction step with regard to this thesis.

- Machine learning: It is a very controversial term as it is used differently in practice. In research however it describes techniques and algorithms to learn from data and as a result follow statements for related but unknown data (i.e. derive new knowledge). Machine learning is therefore a methodology applied in data mining.

A comprehensive KDD process is illustrated in figure 2.4.

Understanding and predicting customers behavior is considered valuable for businesses and therefore a lot of research has been done in mining customer data to gain interesting insights and take the right decision before a competitor does so [50]. Since a major part of the empirical work in this thesis dealt with finding behavioral patterns in customer usage data, different existing methods were studied and evaluated. The following sections provide an overview on the steps of a suitable bottom-up approach and focuses on relevant methods with regard to the implementation.

2.3.4 Pre-analysis work

As extracted from the literature review by [50], dozens of research papers introducing data mining approaches to solve particular business problems in dealing with customers have already been published. However, no matter how powerful and sophisticated those

proposed methods are, none of them will work successfully on unwisely chosen, invalid or bad quality data. This fact emphasizes the importance of preprocessing and preparing data accordingly to increase the chance of retrieving meaningful analysis results [53] [40]. The pre-analysis can be broadly divided into following sub categories:

- Selection of data
- Preprocessing of data
- Transformation of data

The following subsections will provide a more detailed overview on these pre-analysis categories and should give the reader an understanding of the involved tasks.

2.3.4.1 Selection of data

Depending on the type and goal of analysis tasks, the whole database management system of the business should be searched for relevant data collections. In order to optimize this raw data selection process, quality criterion should be determined to prevent choosing irrelevant or invalid data at an early stage [48] [24]. In terms of customer satisfaction analysis, focus has to be put on data collections indicating quality and performance dimension resulting from usage behavior, as this was identified as reliable measurement source in section 2.2.1. Existing approaches belonging to the context of this thesis, like [48], [45] or [64] insist that identifying relevant data sources upfront by domain experts makes the overall process better as only a subset of available customer data has to be analyzed and this saves time. A not negligible part in the selection stage is the orientation of data. Research talks about transactional-oriented data in case it results from operative application. Although this transactional data provides opportunity for intensive analyses, it is not always on the desired level of detail. Then it makes more sense to use data which is stored in an analytical oriented format and thus provides a meaningful aggregated view on interesting parts of a data collection [53].

2.3.4.2 Preprocessing of data

After limiting the number of data sources, a crucial step is to explore the collected data in order to reason about quality and find potential weaknesses- and problems one has to tackle before moving on to the next step. What was found out during litera-

ture research is the fact that despite the rather huge amount of publications regarding technical implementation of knowledge discovery and data mining methods in the customer management area, few put emphasis on preprocessing of data. The effort taken by many authors on particular stages in the preprocessing pipeline, as illustrated in figure 2.4, emphasizes the importance of preparing data before applying specific data mining techniques, like classification, on it. The optimal path through the preprocessing pipeline is hard and cannot be generalized as it turns out to be different for each dataset. Due to the heterogeneous nature of usage behavior data, this step has major influence on the prediction outcome of this thesis research [24].

In large customer generated datasets it is common that some data attributes may be important on the one hand but provide unreliable, erroneous or missing information among several data observations on the other hand. Thus data cleaning was identified as essential part in the preprocessing step [24]. Especially a well defined procedure for handling missing attribute values among data observations has been a desire for a long time. The work of [9] compares methods for handling missing attribute values for data observations with each other and analyzes both strengths weaknesses of them. It was found out that one has to get a clear understanding on the reasons why certain attribute values are missing. Three different categories, namely MCAR (Missing Completely At Random), MAR (Missing At Random) and NMAR (Not Missing At Random) can be differentiated according to [44].

Completely random missing data does not allow to find any predicting variable. Neither known values of the affected instance nor any other missing value among data observations has an impact. In this case any of the available treatment methods leads to unbiased results. In essence this means that it does not matter which method is chosen. The second type, namely NMAR, deals with missing data resulting from unobserved characteristics. This could for instance affect the outcome of interviews with persons who keep some secrets for them and therefore introduce missing data. This type is not considered as relevant in the customer behavior data as customers, frankly said, cannot hide details collected due to their usage behavior. As a consequence data when collected automatically by a computer system mostly falls into the MAR category [22]. It was determined that neither simply discarding particular instances with missing values among attributes nor totally ignoring a subset of attributes with too many missing values are favorable techniques for this type of data. Different methods to replace missing values, as evaluated by [22] and [9], were analyzed in the literature review of this thesis and serve as base for data cleaning work in the implementation part of this thesis. Although not promised as the most reliable method in literature, methods like imputing missing values with mean and mode are, due to its simplicity, popular in the area of machine learning. However, depending on the nature of data it is recommended to take an eye on more sophisticated imputation approaches [9].

The second essential thing to have an eye on when preprocessing the extracted raw data set is the potentially unfavorable distribution of satisfaction levels. Literature research showed that many existing data mining approaches, dealing with classification, had to actively combat this problem in order to increase accuracy of their algorithms in the prediction process later on. Domains affected heavily by this problem are for instance medicine, when it comes to diagnosis of diseases, or fraud detection algorithms in payment systems [16]. These domains can sometimes experience imbalances as extreme as 99% of a majority- against 1% of the minority class [32]. The identified difficulties resulting from such extreme imbalances are on the one hand algorithms, which try to learn from this data, and on the other hand misleading accuracy results presented to the researcher. A typical classification algorithm is well advised to choose the majority class as a prediction result and does not really consider the minority class at all. As a result this leads to an accuracy of correctly classified instances close to 100% but is totally ignoring the minority class. Such a view on the results turns out to be absolutely worthless [15] [41].

As a consequence different solutions have been proposed over the last decade to alleviate the problem and find better performance measures to evaluate classification algorithms. For imbalance problems, sampling mechanisms with the goal of artificially creating balance between the classes were developed. Due to the rather high number of researchers trying to optimize their solutions as well as the amount of different approaches proposed, the sampling mechanisms became a controversial topic. Although working quite well when evaluating them in training, some produced completely different results in practice. This led to an increased number of authors who started to put their hands on the algorithms itself to introduce penalties for wrong classification of data observations and thus steer the algorithm in the correct direction [16]. As mentioned by many authors, simply choosing the classification accuracy as first-class measure is definitely not preferred in cases of imbalance. Over the years especially the ROC (Receiver-Operating-Curve) has established as favorable performance measure [23]. Although it is not expected that there is an extreme imbalance between satisfied and dissatisfied customers, this step in preprocessing attracted attention to the author as it turned out to be a critical point in the area of classification problems. Especially when false negatives are expensive, as it is supposed in case of missing dissatisfied customers, the ROC curve is a good tool to plot the relation between true- and false positives. The importance of ROC will be paid attention in the implementation.

2.3.4.3 Transformation of data

After having revisited the selected data, brought it into the desired format and prepared it by applying and tuning chosen preprocessing approaches, a final step before moving

on to the actual prediction task is data transformation. A general applicable task within this category is feature scaling also known as feature normalization. The purpose hereby is to overcome the problem of different types and units of data as it is typical for datasets which are topic of data mining. Due to the rather large amount and heterogeneity of features in related research projects as [48], [45] or [64], scaling features to a unit scale is considered a duty in order to ensure that classification algorithms take right decisions when learning from provided data. Most popular and widely used scaling mechanisms are the linear scaling to a unit interval range and the standardization to a normal distribution with zero mean and standard deviation of one. Following are the mathematical equations [3].

Linear scaling unit: $\tilde{x} = \frac{x-min}{max-min}$

Whereby x is the value of a particular data observation for a specific feature, min and max is the minimum respectively maximum existing value for this feature among all data observations.

Standardization to normal distribution: $\hat{x} = \frac{x-\mu}{\sigma}$

In the equation above μ is the mean value of the feature currently looked at and σ is its standard deviation.

As it can be seen from the equations, these scaling mechanisms are quite trivial but turn out to be an effective strategy to make working with different distance measures in data mining possible and correct.

2.3.5 Feature Engineering

A whole research topic on its own is crafting relevant features out of the preprocessed- and transformed data. Although much effort is put on the actual machine learning algorithms and new ones get proposed and presented regularly, many data mining projects still fail. It often comes down to bad decisions taken when selecting or constructing features as input for the learning algorithm. While popular machine learning algorithms get applied to thousands of projects, each task is unique with regard to its nature of data and therefore feature engineering has to be tailored to the specific task in order to ensure a good performance in the prediction task later on [21]. A lot of different methods are available to determine importance of features and build a subset used for the learning process. As the data set this thesis has to deal with, contains a number of features easily manageable by a human domain expert, focus will be put at first on manual feature construction as already mentioned in 2.3.4.1. In addition to the selec-

tion of data based on statistical analysis results, automatic feature selection should be tested in the bottom-up approach.

It is important to notice that, as the name implies, feature selection reduces the number of features only by selecting particular ones of the original set. It can therefore be clearly differentiated from feature extraction methods which automatically construct new features out of the raw ones. In research feature selection techniques are differentiated by their evaluation criteria (i.e. how they evaluate which features contribute most) and their generation process. (i.e. how subsets of features are constructed) Firstly, this thesis takes a look on the evaluation criterion considered as part of the implementation. In general an evaluation criteria should always strive for feature sets which allow for optimal discrimination with regard to the ground truth value [20].

Based on the literature review work on feature selection techniques from [14], focus was put on filter- and wrapper method as evaluation criteria. A filter method automatically generates and ranks features based on some calculation criteria judging the contribution of each feature to determination of the predicted value. Based on this ranking, the features contributing the most to the class attribute get selected.

A popular and well researched ranking criteria is the correlation based filter method which is based on the Pearson correlation coefficient heuristic. As major reason for the author to favor this technique is that In contrast to other filter based methods it does not only constrain its view to individual features alone but rather generates and evaluates feature subsets. Other filter-based methods only evaluate the importance of features based on their relationship with the ground truth and therefore cannot detect complementary features within a subset. A correlation based filter adheres to the following principle: The higher the correlation of features with the class attribute and the lower the correlation among features within a subset is, the more likely it gets considered [29]. In mathematical terms the evaluation heuristic for a given feature subset can be formulated like this:

$$r = \frac{\frac{1}{n} * (\sum_{k=0}^{n} \frac{Covariance(Observations(k), Observations(c))}{sd(Observations(k)) * sd(Observations(c))})}{\binom{k}{2} * \sum_{k}^{m-1} \sum_{j=1}^{m} \frac{Covariance(Observations(k), Observations(j))}{sd(Observations(k)) * sd(Observations(j))}}$$

The nominator calculates the correlation between each feature k and the ground truth attribute c averaged by the number of data observations n, while the denominator calculates the correlation between feature k and j among the number of features m present in the subset divided by the number of combinations to select two features out of the subset, which is represented by the binomial coefficient [29].

A second approach evaluating the worth of a feature subset, namely wrapper based selection, uses a learning algorithm itself instead of an measure like a correlation coefficient. According to [20] this has an advantage with regard to the classification accuracy a selected feature set can achieve. However, in contrast to the correlation based method it has two considerable disadvantages. Firstly, it is constrained in its generality as due to the dependence on an explicit learner during feature selection. It requires to use at least the same type of learning algorithm for feature selection as the one intended to be used when applying the actual classification. Secondly, learning and evaluating a classifier is in general a complex task with regard to runtime. Depending on the number of features evaluating each generated subset this way can be very costly. This observations directly leads to the question of how feature subsets get generated which will be discussed in the following paragraph.

Since there are 2^k possibilities to create a subset generating all possible combinations of subsets would result in a bad performance regarding runtime. As this is not practicable for the size of a data set like the one this thesis has to deal with, it is necessary to have a look on suitable generation procedures. The point of stopping to generate subsets can either be based on the generation process itself or the evaluation of a subset under consideration. While the first option limits the maximum number of iterations with a custom defined threshold, the latter one bases its decision whether to continue on the correlation result and the potential for further improvement. The best first search algorithm was found as optimal trade-off between runtime complexity and feature quality. Instead of blindly iterating all 2^k possibilities the algorithm starts with a single feature subset and expands this set in each step with the next feature yielding an improvement regarding the correlation. If the expansion does not yield any improvement the algorithm backtracks and tries another expansion [29] [20].

2.3.6 Predicting satisfaction by leveraging classification algorithms

When the data is finally in the required representation and preprocessed in a way to have eliminated as many flaws in the data as possible, the actual data mining work can be started. Therefore this section deals with the appliance of a machine learning algorithm which fits best to the prepared customer data set. There has already been lot of research in predicting customer churn as this is an interesting management topic in various industry sectors. As its prediction outcome is binary (i.e. churn or not churn) a classification algorithm turned out to fit best [60] [62]. Although customer satisfaction is on a higher abstraction level and more difficult to grasp, features used for churn prediction show similarities to the potential features used for overall satisfaction prediction. It thus served as a good starting point for getting an understanding which classification techniques seem promising. Literature research showed that finding an

outstanding classification algorithm cannot be guaranteed for this domain in general. More specifically to customer satisfaction tied material from [48] and [45] did not lead to clear results regarding algorithm choice. However, it can be derived that decision trees and SVM (Support Vector Machine) were evaluated as well performing algorithms for such a type of problem [60], [62], [48], [45]. Therefore the implementation work of this thesis mainly concentrated on these algorithms which will be briefly introduced in the following sections.

2.3.6.1 Decision trees

Decision tree algorithms are a one of the most mature techniques available in machine learning. They are considered as quite popular and are often used in practice since they are pretty easy to understand, well-performing across different data mining challenges and much faster than other techniques with high runtime complexity. Per definition a decision tree algorithms tries to find an optimal way to hierarchically split the labeled training input data based on expressive features in order to build a model of rules which best represent the target variable. This procedure is done recursively whereby for each split all available features get considered, the one which promises the best separation regarding the class label will be taken for the current split and in turn represents the according split rule. This model can then subsequently be applied on a real dataset to predict the class of data observations following the best path defined by the rules built earlier [43].

Although the basic principle of a decision tree is fixed, different implementations, which mainly differ in the way of determining the optimal split in each iteration, have been proposed over the last years. As a particularly powerful measurement regarding an optimal selection of a feature as well as a split rule, the concept of Information Gain respectively entropy reduction has emerged. The technique is used in the nowadays popular C4.5 decision tree implementation developed by Ross Quinlan. It is the concrete base algorithm used in the empirical work.

A promising modification regarding classification accuracy is the Random Forest algorithm proposed by Leo Breiman in 2001. Instead of finding the best split in each iteration, a defined number of feature subsets is taken. Several trees for the best splits found among those subsets get constructed. The best combination is subsequently obtained by a voting mechanism.

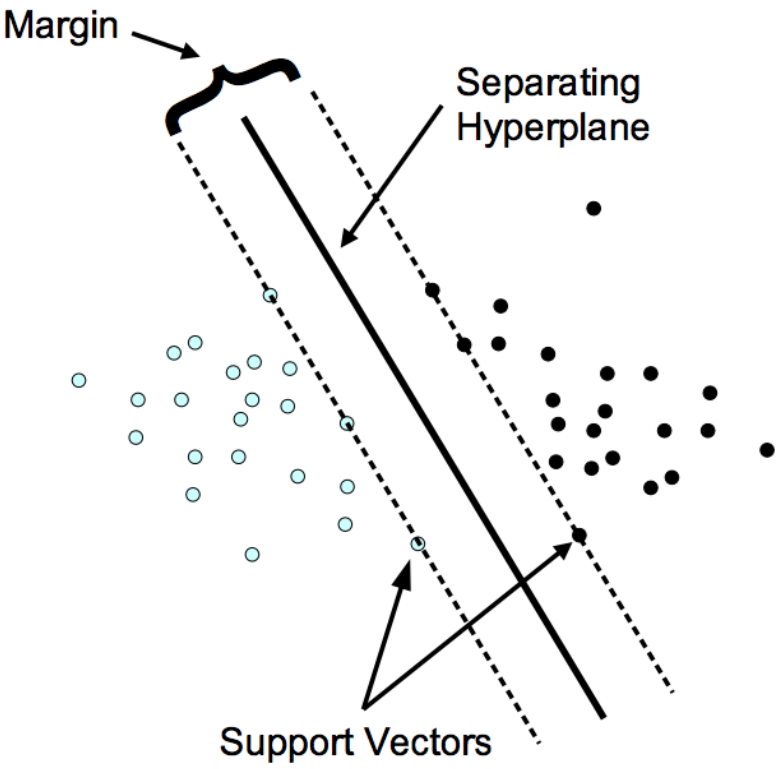


Figure 2.5: SVM - Hyperplane optimization [46]

2.3.6.2 Support Vector Machines (SVM)

The general idea of Support Vector Machines, which were originally proposed back in 1995 by Cortes & Vapnik, is to separate data observations optimally by a hyperplane inserted into the feature space. In order to achieve an optimal separation of the training instances belonging to different classes, the algorithm tries to maximize the margin between hyperplane and the closest vector represented by a training instance on both sides of the intended hyperplane. These closest training instances are also called support vectors. A SVM was only designed for binary classification problems. In an idealized world, the training instances are aligned in a linear separable manner. In such a case there the hyperplane has a vector aligned perpendicular to it. The problem to be solved is the maximization of the distance between vectors of different classes to create enough space for later classification of unknown instances [18], [33]. In order to get a better understanding, figure 2.5 illustrates the hyperplane optimization.

In most of the real world examples, the dataset of this thesis will be no exception, there is no strict linear separation between training instances and as a consequence the basic hyperplane optimization solution will not work out-of-the-box. Instead some method is required to transform the original non-linear feature space into a high-dimensional transformed space which drastically increases the chances of finding the optimal hyper-

plane. As mapping the original feature space into a high-dimensional turns out to be costly, SVMs usually use kernel functions internally to do an implicit transformation [18], [33]. A mathematical summary of this process was done by [33].

Chapter 3

Analyzing data with regard to Customer Satisfaction

This chapter focuses on the design and implementation of approaches to analyze the data made available by the case study and build a solution to make predictions about customer satisfaction based on available usage data. The implementation relies on ideas which were introduced and described in more detail within the previous chapter.

3.1 Overview on available data

Before diving into the details of the proposed approach, it will be briefly pointed out how much data at Tractive could be provided, which data sources were available for the practical part of this thesis and how they are technologically represented.

Although offering some miscellaneous apps and hardware accessories, the main focus of Tractive is selling a GPS device to allow customers to track their pets on a smartphone or in the web. The work in this thesis focused on data produced by customers of the GPS device. Due to the reason that this product is very versatile, data flows into the system from different sources. Data is generated by the hardware device itself, by the customers relationship with the company over time or the usage of provided software products (smartphone- and web app) and their functionalities, to name the important sources.

From a technological perspective, in order to manage this data, a persistent semi-structured database, namely MongoDB, is used. Following listing outlines essential metrics regarding this database to give the reader some understanding about the dimensions.

- Infrastructure: 6 nodes with MongoDB server whereby data is horizontally distributed among 5 clusters (in MongoDB also called shards). This ensures high availability even if a database node goes down or a shard becomes unavailable.

- Databases: The whole DBMS (Database Management System) is split into a main database, containing all raw and transactional data, and a metrics database which contains aggregated data like KPIs (Key Process Indicators) and analytical relevant data.

- Number of collections: 66

- Number of customers actively (i.e. paying for the subscription) using a device: 78038 (as of 26.11.2017 09:58 UTC+1)

Besides this database, a relevant point, especially when it comes to customer satisfaction, is the contact of customers with the company internal support service, as already indicated by the customer satisfaction model in figure 2.2. The support service of Tractive uses a third party software, called Zendesk, to handle customer concerns. This data is managed completely by Zendesk. With regard to the implementation, the author considered to use the official API (Application Programmer Interface) in order to fetch relevant data about customer interactions.

3.2 Identification of relevant data sources

On the one hand, statistical analysis and data-driven approaches are rich tools to gain new insights into the data and find hidden associations and relationships but on the other hand these approaches will not deliver any satisfying results if they work with wrong or invalid datasets as input. Therefore it is essential to invest enough resources on selecting relevant data sources and ensuring high quality within this data before starting with any analytical approach. Considering right and valid data was not only mentioned once in literature as a key factor for successful data analysis [53].

3.2.1 Selecting the right data based on its representativeness regarding Customer Satisfaction

The approach this thesis followed for selecting relevant data sources is based on the definition of customer satisfaction as it was modeled in section 2.1. Furthermore, as already indicated in section 2.2.1, the focus is on the provided service quality and observed

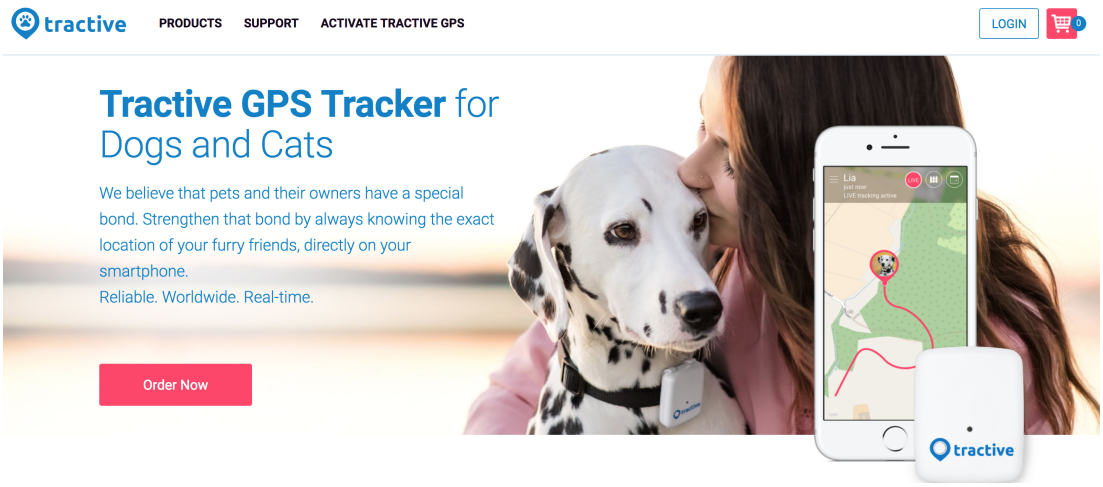


Figure 3.1: Extract of landing page https://tractive.com

performance yielded due to customer behavior, since these metrics can be represented in data and as a result are measurable. In order to create a list of all relevant data collections, the author started with a summarization of properties and features of the Tractive GPS device which are advertised actively to potential customers. As representative resource the landing web page of the company, namely https://tractive.com plus the necessary sub pages with information about the Tractive GPS device, were used to extract this information. The website is the major source of customer conversion and lists all relevant characteristics and features of the product. Figure 3.1 shows an extract of the home page.

Other sources used in marketing like social media channels provide brief summarizations and link to the website. Therefore it can be implied that product descriptions on the landing web page are decisive for customers and drive their expectations with regard to the product. Table 3.1 gives an overview on important properties and features regarding Tractive GPS from a customers perspective. Moreover, it assigns the data which considered as useful for reasoning about customer satisfaction, i.e. it indicates what a customer can expect when he or she purchases the product.

The table excludes properties which are advertised but the nature is static and stays the same among all customers. Such properties are usually product characteristics which do not change due to customer use and as a result are not measurable in collected data. An example is the handy charging unit or the fact that the GPS device is one of the smallest and lightest devices for pet tracking available on the market. From a marketing point of view these are essential characteristics but with regard to this thesis they are negligible due to the reason that they are independent of customer usage behavior.

Feature / Characteristic	**Description**	**Nature of data suitable for predicting customer satisfaction**
Locate pet anytime, anywhere	See location of pet accurately on a map on the Smartphone	Position data (GPS, Mobile Cell)
Live Tracking	Follow the trace of a pet in realtime on a smartphone or on a web page	Live Tracking commands statistics
Virtual fences	Creating a virtual fence and get notified if pet leaves selected safe area	Number of times pet leaves and enters virtual fence, reliability data regarding notifications
Battery indication	If battery of GPS device is full or is nearly empty, it will be indicated in the Smartphone app	Reliability of data regarding battery notifications
Integrated light	Customers can turn on the integrated LED (Light-Emitting Diode) on the GPS device via the Smartphone	LED command statistics
100% waterproof	A product characteristic customers trust in, since many pets often get wet	Hardware defects due to water damage
Premium Customer Service	With a premium service plan, it is promised that customers get feedback within 24 hours on weekdays	Customer service data related to ticket resolving times

Table 3.1: Features / characteristics of product among with their nature of data suitable for predicting customer satisfaction.

3.2.2 Elaboration on data collection regarding identified features

This section will have a detailed look into availability, representation and content of stored data for the features from table 3.1. This task started with an investigation to find out by which of the data collections each feature is represented best and how much value is included in the content. Following paragraphs assign identified data sources to categories and introduce them shortly to make it more understandable for the reader. It is not the aim of the following paragraphs to discuss every data attribute which could contribute to predicting customer satisfaction in depth. Instead the following paragraphs will rather explain briefly which type of data is stored in the collections and how the data is generated as a result of customer behavior. The idea is to get a sense why this data would be important when it comes to customer satisfaction prediction.

3.2.2.1 Device related data

This group contains potentially valuable data resulting from device usage. This type of data is typically generated in two ways. Firstly, by the device itself which sends data to an application running on the companies rented server infrastructure. This backend service is connected to the MongoDB where it stores the incoming data in

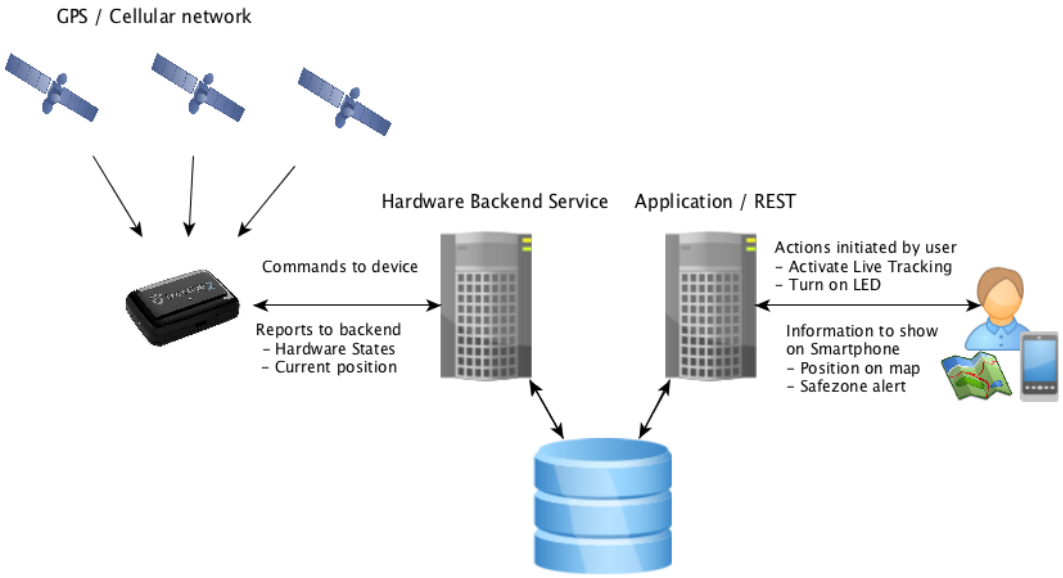


Figure 3.2: Overview on communication and data flow between client and device

the database. Secondly, by sending commands from a client via a REST (Representational State Transfer) API to the backend services which persists data if necessary and communicates with the device by sending encoded information via SMS to the device. Finally the device should initiate some action due to the provided command. Before looking on concrete schemata of the data, figure 3.2 illustrates in a simplified way the communication and thus the data exchange between client and device.

The following list outlines the essential device related data collections leveraged in the analytical part of the thesis.

- **Hardware information**: Devices usually differ when they leave the production process. Therefore some hardware details about a device are stored in its data. Following list is not comprehensive but points out attributes causing potential differences in quality for the end customer and therefore considered as valuable enough to be part of the customer satisfaction analysis process.

  - Batch number: Devices are manufactured in mass production and assigned to a so called batch which contains devices with same configuration. If there are major changes on a devices conceptual design a new batch number gets created for the new series of devices.

  - Firmware version: This is the software embedded in the device. Similar to the batch the firmware gets updated as well if new features are added or problems occur.

  - Model number: Currently three GPS device models exist. The model of a particular device is indicated by this attribute.

  - Hardware edition: Tractive offers three editions which differ in their color and style but have the same hardware specs. Although not supposed as having as much quality influence as the other three attributes, it was decided to include this property in the analysis in order to investigate possible relationships between a user's taste and satisfaction.

  - SIM (Subscriber Identity Module) type: The communication with Tractive GPS devices is done via SMS and therefore each device has an integrated SIM card comparable to a mobile phone. Tractive uses SIM cards of three different providers to manage SMS communication.

- **Device ID reports (Device identification reports)** are sent regularly if the device is switched on and has network coverage. The ID reports contain information about the current firmware- and hardware version of the device. Since there are bug fixes and improvements on the firmware quite often, differences in service quality over time are possible. Furthermore the device identification reports allow to reason about the activity level (eg.: the number of days in use) of a customer's device.

- **Device position reports**: Every time a device receives a position it is sent to the backend service and in turn stored in the database. Along with the latitude and longitude of the position there is data indicating whether GPS or mobile cell network was used for localization, how strong the received signal was, as how accurate the position is classified, the timestamp when the device received the position, the time where it was stored in the database and some further technical GPS data as for instance the number of reachable satellites used for localization. The time interval in which positions are sent is configured in the hardware electronics. An exception is the live tracking feature where positions need to be sent every second to provide the customer his or her pet's location in real time. Due to fact that each position report gets stored, a lot of data is produced which makes this collection the largest data sink in the whole database. Moreover, a lot of service quality information is passed along with each position report and thus makes it a valuable data source for customer satisfaction analysis.

- **Device network reports** contain data about a devices network coverage. Since the device can be used in over 100 countries worldwide each network report provides the MCC (Mobile Country Code) which indicates where the device is currently used. Prior experiences showed that network coverage sometimes varies a

lot among countries which can influence satisfaction of a customer. The mobile network generation which is limited to GSM, for devices of the first generation, and UMTS, the more recent one, can have an impact on the perceived quality as well.

- **Device hardware reports**: These reports are usually sent along with the ID reports described before and contain different kinds of hardware states. Interesting attributes are for instance the current battery level- and voltage, temperature states, collected number of position or error and network logs for the particular device. Moreover hardware reports store specific hardware events for a point in time where they happened. Amongst others this includes valuable information as the time when the device has been switched on/off or battery is charging, full, low or at a critical level.

- **Geofence reports**: Customers have the ability to define a virtual fence for their pet(s) in the mobile- or web app. If a customer's pet with a mounted device leaves or enters such a virtual fence, a geofence report with the according trigger (called in-break in case of entering the virtual fence or out-break otherwise) and source (GPS or mobile cell) is sent to the server. There it gets stored again in the database and the application initiates sending a push or mail notification (depending on the configured settings) to the customer. Both, accuracy based on the trigger source and the number of in- and out-breaks are assumed to influence customer satisfaction as it is one of the most used features among the customer base.

- **Server commands**: These are commands sent due to an action initiated by the user. The most prominent server command according to the statistics is the live tracking function. If this server command is received by the device, it starts sending a new position every few seconds for a maximum defined duration threshold, which allows the user to follow his or her pet in real time on the respective client device (smartphone- or web app). Further commands enable the user to turn on/off the integrated light or trigger a sound on the device. The server commands are essential for customers of the Tractive GPS device and therefore several attributes indicating the quality are collected. For each server command the timestamps when the command was sent by the server to the device, when it was received by the device, the time of confirmation respectively cancellation as well as the duration of the command is recorded. Due to the business relevance of these server commands, it was decided in March 2017 to bring this data in a different structure where detailed analysis can be done more easily than with the transactional raw data. Starting with mid of May, every new server command causes a new entry in a separate metrics collection. The

calculations and persistence is done by an asynchronous job which is started one hour after a server command was created in the database. This metrics collection expands the specific server command data with useful device related data as for instance the hardware edition, batch number, SIM type or the country of usage. The meaning of these properties was already explained in the first bullet point. Furthermore, in addition to the success indication with regard to live tracking commands, the company also wanted to have information regarding time and duration of commands. Therefore, following time information gets calculated:

- Delay to commanded: The time it takes that a command requested by the server is received by the device. This time is only set if the command could be received by the device. The value for this attribute can be missing if the communication between backend and device did not work.

- Delay to confirmed: The time it takes that a command requested by the server is confirmed by the device. A command gets confirmed if the device could initiate the action (e.g. enable the live tracking function). Not all commands get confirmed. (e.g. it can happen that the device is inside a building an cannot get a strong GPS connection required for live tracking)

- Delay to any position: The time it takes from requesting the command by the server until the device could get a position. In order to find the time of retrieving a position, a lookup in the device position reports collection has to be done. For this type of delay there is no restriction regarding the quality of the received position. The maximum delay allowed is 90 seconds after the command was requested by the server. Only under this circumstance the usability of the live tracking function is considered as satisfying.

- Delay to a new position: This delay is very similar to the one before except that it only considers new positions which were received via GPS and judged as accurate enough.

- Delay to ID report: Each server command is supposed to trigger sending of an ID report as well. The maximum delay allowed is again set with 90 seconds after the command was requested by the server.

- Command duration: This indicates the time the command action was enabled. This duration is only measured and stored if the command has been confirmed and terminated successfully. The exact duration hereby measured is the time passed from confirming the command by the device until the point in time where the user disabled the function.

- **Battery lifetime metrics**: Similar to a smartphone the battery lifetime is a critical component in the Tractive GPS device. If the pet is not close by and the customer wants to locate it, the battery should not be discharged. However, especially the network connection and GPS signal exhaust the battery. As Tractive also provides a position history, the device permanently (with regular gaps) tries to fetch a new position. With live tracking enabled battery suffers even more. Depending on the intensity of usage, the battery lifetime varies. This should be expected by customers but despite that fact battery lifetime varies between GPS devices. Since battery lifetime is considered as critical, it was decided to create a new collection suitable for detailed analytics of battery lifetime among devices analogically to the server command metrics. In addition to some hardware information as the model number and hardware edition this collection stores battery cycles, based on the device hardware reports, which were explained in more detail the fourth bullet point. Each time a hardware report with status indicating full battery is received from the device, an asynchronous job is executed to calculate the information for the previous battery cycle, starting with the highest reported battery level and ending with the lowest one before recharging the device again. Due to the complexity of reliably calculating time for a battery cycle threshold were defined to categorize a battery cycle. In order to characterize a cycle as full, the highest battery level reported has to be at minimum 90% and the lowest one at maximum 10%. Moreover the device should not have reported any shutdown since this would distort the result. As of 30.11.2017 17:54, 45.32% of all reported valid battery cycles were characterized as full ones. Despite the fact that durability of a battery in an electronic device is unreliable, this thesis considered only full cycles as representative to reason about average battery lifetime of a tracker as interpolation calculations from incomplete cycles would lead to invalid results. Due to the reason that recording those battery cycles was implemented later during the empirical work, the author had to notice that not as much data as desired is available for the average battery lifetime calculation of devices. This restriction had to be considered accordingly during the analysis part.

3.2.2.2 Notification related data

Notifications play an essential role for Tractive customers, as for instance the information whether a pet has left a defined virtual fence can be crucial for customers. The assumption made in this thesis relies on the fact that reliability of notification delivery either via mail but even more important via push notification on the smartphone has an influence on customer satisfaction. Customers of the Tractive GPS smartphone app can receive a palette of different push notifications. Following listing briefly summarizes the notification types which the author considered as potentially influential.

- **Geofence Notification**: The opportunity to define a virtual fence for his or her pet is one of the mostly used features among Tractive GPS customers. As a result it is important to get a notification if a pet leaves or enters such a virtual fence. This is represented by a Geofence Notification which contains the affected device identifier as well as the information whether the pet left or entered the virtual fence. According to the importance, this type of notification is considered to be of highest priority.

- **Hardware battery notification**: As the device is mounted on the collar of a pet and the customer most of the time is not close by, it is a relief to get updates about the battery state. In addition to the device identifier, a particular battery state of an predefined enumeration of states is stored.

- **Resource notification**: In terms of the use case of Tractive a resource can be of different type whereby an image is the relevant one for this notification type. Next to the Tractive GPS app which is the main product for customers of a GPS device, there are other apps to use. Users can for instance share their photos with other Tractive users or photos can be liked or commented on from the Tractive community. As such notifications are not direct quality indicators, they are ranked low on importance.

- **Hardware alert notification**: These notifications are sent based on sensor measurements of the hardware device. Exposed sensor values are temperature and mount state of the device. A customer should receive a notification if the temperature of the device is too high respectively low as well as when the device gets unmounted from the collar. As these hardware alerts are only present for a specific version of the GPS device, there is little data available for analysis. Thus, these notifications are considered as low priority when it comes to customer satisfaction.

Every notification sent from the system is stored in the database with the information, depending on the specific type, described in the previous listing along with user data. As a consequence it can be followed which and how many notifications of a specific type a customer has received until a point in time.

In addition, any mail- and push notification results are stored in according log collections. Especially for time critical notifications like geofence notifications, the time passed between the event has been triggered by the device and received as notification in the push bar of the smartphone would be of big interest. Since push notifications are sent via a publish/subscribe message bus and later through the respective push service of iOS and Android, there is currently no mechanism implemented to measure

and store this overall time. As a consequence, the available notification data is limited to the bare number of received notifications in a specific interval.

3.2.2.3 Customer service related data

Tractive advertises a first-class customer service which about a third of customers of a GPS device actually make use of. An extracted statistic from the database shows that since February 1st, the date where Tractive started to sync support contacts to the database, 34.42% of new customers who activated a device had at least once contact with the customer support. These numbers clearly emphasize the importance of a good and responsive customer service. As explained in section 2.1 when illustrating the model from [36], customer service is assumed to contribute to the customer satisfaction complex.

In order to bring more structure in the collected information, Tractive employs a customer support service tool called Zendesk® which is the central place for desires, complaints and support requests. Instead of traditional email, a customer can send a support request via a help center web page. As result, a support ticket in Zendesk® gets created which can then be handled by support employee of Tractive. This ticket based approach makes the support team more efficient. In addition, it allows a better data integration and has the advantage that all collected information regarding particular support tickets is stored transparently in Zendesk® and can be retrieved if needed.

From a technological perspective every update on a support ticket is synced with the backend service and stored in the database. Moreover, Zendesk® provides a comprehensive REST API to query details of a specific ticket. As particularly interesting the author considered the opportunity to query ticket metrics provided by Zendesk®. Following listing briefly outlines the ones considered as valuable in terms of quality indication for customer support [1].

- **Ticket count**: The number of tickets a customer opened.
- **Requester waiting time**: The time a customer has to wait in- and outside of business hours until he or she receives the first answer.
- **Reopens**: Indicates how often a ticket was reopened. Reopening a ticket usually happens if the problem of a customer reappeared after the support employee closed the ticket as resolved.

- **Resolution time**: The time it takes to fully resolve a customer's incident in- and outside of business hours. This covers the time from opening a ticket for the first time until it is closed an not reopened again.

- **Replies**: The total number of times a ticket was replied to.

The mentioned metric attributes can be queried for every ticket in the system. Before being able to query these metrics, the tickets of a customer can be found via their email address which is needed while registering a Tractive account.

3.2.2.4 Subscription related data

Every customer who purchased a Tractive GPS device has to activate it in order to be able to use it. This device activation results in a subscription which enables recurring payments for the tracker usage comparable to a mobile phone contract. Tractive offers payment plans differing in their type, which can either be "basic" or "premium", and the payment interval, which can be monthly, annually or biennially. In addition, customers have the opportunity to opt in for a device insurance to get a free substitution device in case the first one gets lost or breaks accidentally. Customers can cancel their contract any time they want and also have to possibility to change their payment plan at any point in time during their relationship with the company. This can be done in form of upgrading or downgrading the subscription type or changing the payment interval. Based on previous work regarding customer satisfaction theories as described in section 2.1, price and costs are considered as influential factors for customer satisfaction.

The whole payment and subscription related data of customers is spread over several collections in the database. From a thesis' perspective the focus is put on the subscription information which is the actual interface between a customer and his or her device. The type and payment interval as well as the information whether a customer added the device insurance to his subscription were assumed as most influential factor. Tractive on the one hand offers a cheaper price if the customer chooses to subscribe for a year or even two years instead of just a month and on the other hand advertises more features to premium customers. As a consequence both customer expectations and perceived quality should vary based on his or her choices.

3.2.2.5 Other relevant data

In addition to the categorized data sources, the search yielded some more data attributes among different collections which are worth to investigate in more detail during the

analysis part. Since this data does not fit together in any of the defined categories, the thesis groups them together in this section.

As interesting metric, the number of times a customer opened the Tractive GPS smartphone app in a specific time interval, was considered. Although this information is stored in a separate mobile events collection, the data is not highly reliable as the sent events vary between iOS and Android. Moreover, there are different events, like app_foreground and app_startup on Android, where it is later not possible to be sure what behavior the user really showed. As a result, these events are only vague numbers and have to be handled with care. Next to the Tractive GPS app, some other mobile apps from Tractive are available in the according app stores. This information is supposed to give more insights into the engagement of a customer in the Tractive ecosystem and his or her appreciation of the brand.

The importance of social media marketing definitely plays a major role at Tractive. Customers love to share photos of their pets to other people and thus Tractive also employs their own photo platform. Users can upload photos as well as comment and like photos of others. Although social activities do not directly emerge as predictor from the customer satisfaction model introduced in section 2.1, this thesis wants to check possible correlations and therefore included available data in the analysis. It was found out that the number of resources liked and commented on associated with a user can be derived from available data. As no published research work focusing on such a relationship could be found during literature research, it was worth to take a look on possible correlations as part of the analysis.

3.2.2.6 Data not considered as relevant

Based on the relevant data sources introduced in this chapter so far, the attentive reader can see that there is already a lot of data which has the potential to make a difference in terms of customer satisfaction. However, to make clear why these choice were made when searching for relevant data sources, this section exemplarily outlines attributes which did not make it in the analysis part.

Since Tractive is operating world wide, customers come from different countries and speak various languages. Geographical- and demographical data, like country, language or gender is considered as static and does not reflect a customer's behavior. Customers can assign a pet to their tracker and enter detailed information about their pet like type, name, weight, height or breed. Despite the fact, that this information can be valuable for the company in order to get more insights about the type of pets the

device is often used for, it is not topic of usage behavior and thus not considered as relevant for customer satisfaction in this thesis.

Lastly, it is important to notice the general limitation of customers whose data is used during analysis, as only those who activated at least one GPS device, are considered. Other users who have registered and probably use some other Tractive app without a GPS device will not be considered at all in this thesis. On the one hand, there is too little data available for them and on the other hand these customers are topic of customer- acquisition instead of retention. Moreover, some customers own more than one device which would produce problems when data is tied to the user ID and not to the particular customer-device relation. An example would be the app usage where no separation based on the device can be made. It could be for instance the case that a user often opens the app to locate one of his or her pets but is not interested in the other activated devices. When comparing this number with device specific information the matching would lead to wrong results and as a consequence wrong input for the analysis part. Since the customer base of Tractive is big enough, the analysis only considers customers with one activated device which fixes the outlined problem and prevents distorted results.

3.3 Implementation of a data extraction tool

After identifying relevant data sources and matching them to specific collections and attributes in the NoSQL database, the first major technical task was the implementation of a program to extract this data and bring it in a format which allows analysis by a statistical program.

In contrast to a relational database, MongoDB uses JavaScript for querying and manipulating data. Principally data can be queried via the MongoDB shell which comes in handy when doing standalone queries. However, for the planned amount of data extraction a lot of querying has to be done among different collections. In order to connect results and manipulate them accordingly to prepare it for the analysis, more logic was required. Thus, a data extraction program was implemented in JavaScript which uses the NodeJS MongoDB driver to connect to the database as illustrated in code listing 3.1.

```
1 var connection = {};
2
3 function createDbConnection(url, port, database, cb) {
4   MongoClient.connect('mongodb://' + url + ':' + port + '/' + database,
        function (err, db) {
5     if (err) {
6       throw err;
```

```
    }
    if (database === 'tractivedb') {
      connection.tractivedb = db;
    } else {
      connection.metrics = db;
    }
    cb(null);
  });
}
```

Listing 3.1: Connecting to the database via MongoDB NodeJS driver

The implementation leverages the powerful aggregation framework of MongoDB for most of the collections where data should be collected. This aggregation framework provides a palette of operators for grouping and aggregating data which makes it easier to perform certain queries which are standard in SQL (Structured Query Language) but hard to do with a NoSQL database. The programmer can specify a pipeline of operations where the intermediate result is passed top-down. Especially the rather huge amount of promising attributes in the device related data collections made it necessary to make querying functions generic and reusable. The code snippet illustrated in listing 4.1 shows how to query the average of a certain metric of an arbitrary server command for a device used within a defined time interval. As a result this method can be used for different server commands not matter whether it is the live tracking command or led switcher. Plus it can be reused to calculate for instance the average of successfully-, canceled -, or terminated commands which makes the function quite flexible to use.

```
var COLLECTION = 'server_command_metrics';

function queryServerCommandMetricsForTracker(commands, cmdStatistic,
    startDate, endDate, trackerId, cb) {
  var matchCriteria = {};
  matchCriteria[cmdStatistic] = {'$exists': true};
  matchCriteria['msg_name'] = {$in: commands};
  matchCriteria['mode_on'] = true;
  matchCriteria['device_id'] = trackerId;
  matchCriteria['requested_at'] = {'$gte': startDate, '$lt': endDate};

  var groupStage = {_id: '$device_id'};
  groupStage[cmdStatistic] = {$avg: '$' + cmdStatistic};

  var pipeline = [
    {'$match': matchCriteria},
    {'$group': groupStage}
  ];

  db.aggregate(db.getMetricsDbConnection(), COLLECTION, pipeline, cb);
}
```

Listing 3.2: Aggregation query on server command metrics collection

The previous code snippet shows that the execution of all those MongoDB queries is asynchronous as indicated by the last parameter of the function called ”cb”. Every asynchronous function in the data extraction program is handled by a callback which returns either an error if the operation failed or the respective result object. Handling asynchronous results in JavaScript quickly gets confusing and increases maintenance effort due to the nesting of callback functions. Since some of the queries require as input the result from a previous query, the implementation makes use of the async library to handle the control flow of asynchronous function executions. How this can look like in JavaScript is shown in listing **??**.

```
1  function getAppUsagesForUsers(userIds, startDate, endDate, cb) {
2    async.series({
3      clientIds: queryClients
4    }, function (err, result) {
5      async.waterfall([
6        async.apply(db.createConnection, 'tractivedb_metrics'),
7        async.apply(queryAppUsagesForUsers, userIds,
8        util.getObjectIdsAsStringArray(result.clientIds), startDate,
             endDate)
9      ], function (err, appUsages) {
10       cb(err, appUsages);
11     });
12 });
13 }
```

Listing 3.3: Example of using async for handling the control flow of asnychronous database queries

The illustrated JavaScript example provides an insight how finding the app usage of users is implemented. On the top level there is an asynchronous function to retrieve all available clients (i.e. the different Tractive apps). The waterfall function is used to pass the resulting connection object from the callback of the createConnection method to the actual function which queries the app usage of customers. With async.apply additional parameters can be passed to the respective function.

To construct pairs of attribute vectors, the query results represented as JSON objects available in memory have to be combined correctly to make them comparable by statistical methods. In order to analyze for instance a relationship between the average success rate of server commands and the number of times the customer opened the Tractive apps, a common property has to be used to merge the results. This reference property in most cases is the unique ID of the registered user or, if not available in one of the two data sets to merge, the unique device ID. As already mentioned in section 3.2.2.6 only customers with one device are present in the samples which allows to safely merge based on both properties. With regard to the upcoming section which will shed light onto the first analysis approach, the resulting feature vectors finally were printed

to separate CSV files which is the most common file format and interpretable by any statistics software.

3.4 Hypotheses-driven approach to gain knowledge about interrelationships in data

Based on the prepared data, analysis work started with a top-down approach as described in section 2.3.2. The aim was to get an understanding which of the extracted data related to customer behavior is expressive enough to be considered as influential factor for customer satisfaction. Hypotheses were explicitly defined and either verified or falsified by statistical measurements. It was planned to use results if they are promising to build a framework which allows predicting the satisfaction level of other customers.

3.4.1 Choosing target data approximating Customer Satisfaction

While section 3.2 explored the available data sources to identify those which are assumed to be predictors for customer satisfaction, the work explained in this section analyzed data with regard to actual correlations. Due to the lack of explicit customer feedback, the author had to overcome this difficulty by choosing implicit data as target variable for customer satisfaction. This data should be expressive enough to distinguish between satisfied and dissatisfied customers. The author proceeded by finding behavioral patterns, customers would follow if they feel pleased or disappointed and came up with following collected data:

- Recurring service active or canceled: The assumption hereby is that satisfied customers will keep their service where they pay monthly, yearly or biennially and are therefore able to use the device anytime. Dissatisfied customers are vulnerable to cancel the service and end the relationship with the company after the current payment interval has ended. These thoughts are based on the theory of relationship between customer satisfaction and loyalty outlined in section 1.1.

- Increased or diminished app usage: The assumed behavioral property resulting from (dis)satisfaction is that customers increase respectively diminish the usage of the smartphone app. According to analytics data, the Tractive GPS app for iOS and Android is most important for customers of a Tractive GPS device. As a result of good user experience it is expected that customers get more satisfied and therefore use the app more often.

- How many days a GPS device is in use: The device ID reports, explained in more detail in section 3.2.2.1, can be seen as an activity indicator when aggregated over a period of time. The thesis considered the number of days the device sent ID reports as a meaningful number related to increased of decreased satisfaction.

3.4.2 Formulation of Hypotheses and solving analysis problem

The upcoming paragraphs provide a more detailed insight into the procedure of defining hypotheses and the appliance of statistical tools to verify respectively falsify them. Following, a sample of executed analysis tasks is exemplarily outlined.

3.4.2.1 Live tracking feature as customer churn indicator

This first analysis done in the implementation part of this thesis is based on the relationship between customer satisfaction and loyalty. Therefore the thesis differentiated between the service status "active", which indicates that a customer is paying on a recurring basis, and the service status "terminated", which indicates a manual service cancellation by a customer. This implies the fact, that this task relies on binary categorical data. The analysis goal was then stated as following: Does Live Tracking success influence service termination behavior?

1. Hypotheses

   H0 There is no significant difference in service termination behavior between customers who belong to the group of bad live-tracking users and customers who belong to the group of good live-tracking users.

   H1 There is a significant difference between these two groups.

2. Analysis objects

   - Live Tracking Server commands related to a specific device
   - Service status of customers

3. Analysis problem

- Select representative sample and split it up into two groups to compare, namely a treatment and control group. It is important to exclude any other influencing factors as best as possible and therefore define common base data shared among both groups.

- Choose a suitable sample size

4. Analysis solution: Statistical test which should check whether the Null-hypothesis can be rejected and thus statistical significant difference can be shown. The analysis was done on 17.04.2017.

   - Group selection: The first group contains random customers suffering from bad live tracking which was defined by an overall success rate below 70%. In contrast, the control group consisted of customers with a success rate greater or equal than 80%. These thresholds were set according to experience in customer support where complaints from users regarding live tracking usually show success rates around this threshold. As common base data only users from Germany who own one Tractive GPS device and have a subscription of type basic with a monthly payment interval, which was created before 01.10.2016 and is at least valid until 01.10.2016, were considered.

   - Choosing a suitable sample size: The sample size for the two groups was estimated based on findings of [13]. Regarding statistical significance a widely used α-error of 5% was used while a β-error of 10% should reduce the probability of false-negative results, meaning non-rejection of H0 although it could have been rejected. Since a statistical significant result does not necessarily say something about expressiveness of the computed result, a minimal relevance level had to be set for choosing an appropriate sample size. A 20% decrease of active subscriptions due to bad Live Tracking was considered as relevant. Picking the right number from the sample size table proposed by [13] yielded a size of 79 items group, or 158 in total.

   - Querying data: At the time where this particular analysis was conducted no analytical view on the server command metrics was available and therefore the procedure was to calculate those metrics for the devices under consideration manually. Therefore the implementation fetched a random sample of subscriptions with the specified size, either of status "active" or "terminated" with the described common data. Based on the device ID reference, server commands of type "Live Tracking" within the given time period were filtered and the success rate averaged. Based on the output data a post-processing step provided a simple 2x2 table which is illustrated in 3.2.

	Service status = ACTIVE	**Service status = TERMINATED**
Good Live Tracking	54	8
Bad Live Tracking	43	19

Table 3.2: 2x2 table showing influence of Live Tracking on service status

- Statistical test to verify or falsify H0: There are different statistical tests for binary data whereby Fisher's exact test is proposed as most suitable for a rather small sample size as it was in this case [55]. The test works based on a 2x2 table as input and calculates a 95% confidence interval indicating where the true odds ratio lies. The odds ratio is an often used value when analyzing binary data and states the relative probability than an event occurs against that it does not occur [10]. The Null-hypothesis in Fisher's exact test can be rejected if the confidence interval does not include the odds ratio 1. Then it is safe to claim that there is a difference between the two sample groups under consideration. The open source statistic program package R was used to execute the Fisher exact test, as illustrated in listing 3.4, for the 2x2 table.

```
1  m <- matrix(c(54, 8, 43, 19), 2, 2)
2  fisher.test(m)
```

Listing 3.4: Execution of Fisher's exact test in R

The result yielded a confidence interval of $[1.105933, 8.604534]$ which indeed allows rejecting H0. Based on this statistical experiment it is thus valid to say that there is a statistical significant and based on the minimum of 20% difference in terminations between treatment and control group also a relevant result.

- Interpreting the result: This data analysis confirms the expected importance of the live tracking function for Tractive customers since terminating an active service is equal to ending the relationship with the company and can therefore be considered as a last resort in case of dissatisfaction. As a result it can be stated that live tracking success rate qualifies as potentially promising feature for customer satisfaction.

3.4.2.2 Abnormally canceled live tracking as indicator for app usage

This example investigated the potential relationship between live tracking commands, which were not canceled on purpose by the customer, and the app usage.

1. Hypotheses

H0 There no significant difference in the app usage between customers with a higher canceled LT rate and the ones with a lower rate.

H1 There is a difference in the app usage between those two groups.

2. Analysis objects

   - Server command metrics collection
   - App Events collection containing app startup and app foreground events

3. Analysis solution

   - Choose a suitable sample size: For an appropriate choice the width of the confidence interval, where the true correlation coefficient of the population lies, was considered. The author decided on 0.1 as its width for the experiments to ensure a high precision. Since the app usage was not considered as a direct measurement of a customer satisfaction value, a sample correlation coefficient of 0.4 was determined as relevant. Based on these parameters and the research of [47], a sample size of 1086 could be derived for the correlation analyses.

   - Select sample users: The selection was done by randomly skipping users. Since a few users did not execute any server command with the selected time period, the sample size was automatically reduced by the program to 1065. The output of this selection stage contained user- and device ids.

   - Query data to compare: The sample users fetched in the previous stage were used as input parameter for the function which fetches the number of app foreground events within a selected time period for the current user under consideration.

   - Match data based on user ID: The data collected was stored in JSON (Javascript Object Notation) arrays in-memory. A matching function finally generates the input vectors for the subsequent statistical calculations, by matching the nested JSON objects via the user id and writing data to one CSV file each.

   - Use correlation analysis to reason about linear relationship: After finishing the querying part, the data was ready for correlation analysis. Therefore the statistical open source program package R was used to calculate a Bravai-

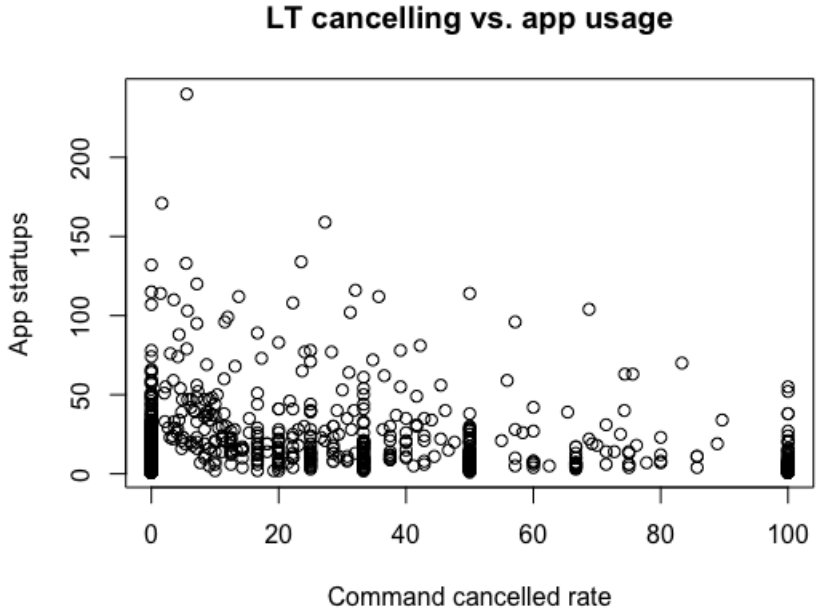


Figure 3.3: Scatterplot - Canceled rate vs. app usage

Pearson correlation coefficient. The calculation part is illustrated in listing 3.6.

```
1  cmdCancelleDrate = read.table("hypo-tests/data/
       cmdCancelledRate.csv", sep = ',')
2  appUsages = read.table("hypo-tests/data/appUsageCancelled.
       csv", sep = ',')
3  cor(cmdCancelleDrate, appUsages)
```

Listing 3.5: Calculation of pearson correlation and statistical test in R

The scatter plot shown in figure 3.3 does not look promising with regard to any kind of linear relationship between the abnormally canceled live tracking rate and the app usage.

The calculated Pearson correlation coefficient is $r = -0.0420426$. This result indicates a negative linear correlation as it was expected by the null hypothesis, which in essence claims that a higher rate of canceled server commands leads to a diminished app usage. However, the result confirms the visualization in the scatter plot from before as it is close to zero. In order to formally prove the hypotheses under test, the Pearson's product moment correlation test was applied on the two data rows. This test calculated a 95% confidence interval to verify whether the Pearson correlation coefficient is statistically significantly different than 0.0. (indicating no linear relationship at all) Listing 3.6 shows how this was done in R.

```
1  cor.test(cmdCancelledRate, appUsages)
```

Listing 3.6: Pearson correlation test in R

Following, the result of this correlation test is shown. $alternativehypothesis$: $truecorrelationisnotequalto0 95percentconfidenceinterval$: $-0.10185605 0.01807374$ As a result it can be stated that here is no significant relationship between these two data attributes.

3.4.2.3 Signal Strength vs days in use

The third example concluding the exemplary listing of hypotheses regarding customer satisfaction, had the goal to investigate whether the quality of the received signal from the telecommunication network influences the activity level of the tracker determined by a customer's usage behavior. For the signal strength an available indicator are the RSSI (Received Signal Strength Indicator) values from the GSM network (Global System for Mobile Communications). These RSSI are stored inside a position report. Due to the reason that customers also sometimes try to locate the device when their pet is indoor, it can happen that the RSSI value of a position report is good although there is no GPS signal available and as a result the location is inaccurate. Therefore it was important to only consider RSSI values where the used sensor was GPS. The activity level of a device can be queried via the ID reports. The most common metric used at Tractive's business is the number of ID report aggregated per day. (i.e. the number of days the device was in use) Following is a summary of the analysis executed.

1. Hypotheses

   H0 There is no significant difference regarding the days a device was in use between customers with a higher average RSSI and the ones with a lower one.

   H1 There is a difference in the days a device was in use between those two groups.

2. Analysis objects

   - Device position reports collection
   - Device ID reports collection

3. Analysis solution

- Choose a suitable sample size and sample users: Selecting the sample of users for this analysis task was chosen based on the thoughts from the previous analysis task outlined in 3.4.2.2. As a result, 1086 users could be randomly selected.

- Query data: Since the data extraction tool was implemented in a generic way it provides a function to query any average value for the different hardware attributes which come along in a position report. The average GSM RSSI value for GPS based position reports was queried and stored in a JSON array. The second part to prepare the data was querying the device ID reports grouped by the ID of the device with an $dayOfMonth$ aggregation over the report time. Finally, the sum of documents retrieved are summed up for each device which yield the number of days in use.

- Match data based on device id: The base matching function to merge the RSSI data with the days in use was reused here with the exception of taking the device ID as matching parameter because the user ID was not available in the considered collections.

- Similar to the previous analysis task, this example also deals with two continuous numeric data vectors and therefore the Bravai-Pearson correlation coefficient could be used as statistical relationship measurement.

Results The scatter plot illustrated in figure 3.4 shows a random distribution of points indicating no relationship between the GSM RSSI value and the number of days the GPS device was in use within the given time period.

Calculated Pearson correlation coefficient: $r = 0.01835016$. Analogically to the previous analysis tasks, the Pearson's product moment correlation test was calculated and yielded as a result that the Null hypothesis cannot be rejected since the calculated 95% confidence interval includes the value 0.0. The essential result from R is shown below. $alternativehypothesis$: $truecorrelationisnotequalto095percentconfidenceinterval : -0.041181620.07775211$

3.4.3 Evaluation of approach and derived decisions

The first analysis task for a randomly selected sample of subscriptions yielded a statistical significant and clinical relevant difference between users tending to behave loyal in terms of staying active subscribes and users tending to cancel their subscription based on the live tracking success rate. The asymptotic Chi-Square test as well as

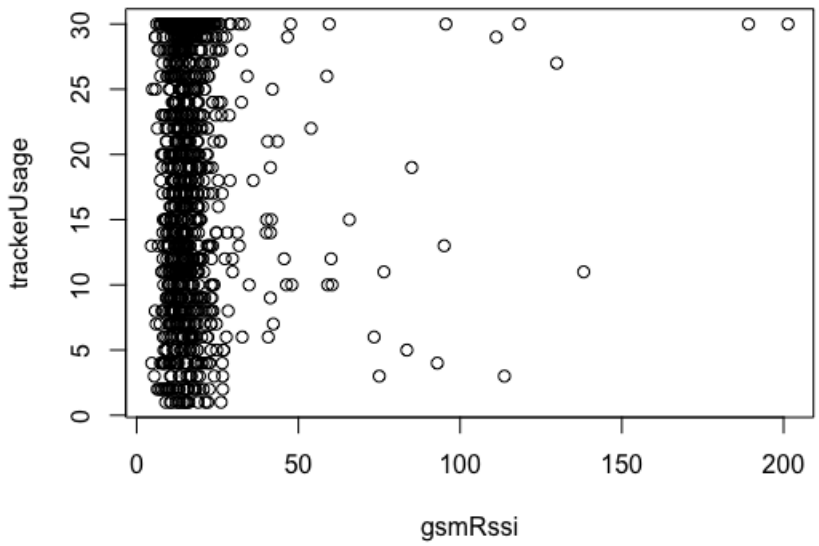


Figure 3.4: Scatterplot - GSM RSSI vs. Days in use

Fisher's exact test supposed to reject the Null hypothesis which confirmed the initial assumption that the live tracking success rate is a critical business factor for Tractive. However, due to complexity regarding the aggregation queries and hypothesis test possibility, this statistical test only considered two groups of subscription states, namely "active" and "terminated". Therefore payment failures and as a result expired- as well as paused services were not considered. In the following hypotheses driven tests several correlation coefficients between a given behavior metric and an assumed customer satisfaction driver, like the app foreground events or the number of a GPS device's usage days, were calculated. For none of them the author of this thesis could derive any further influential factor for customer satisfaction. One of the major issues identified is the single implicit metric used to explain customer satisfaction. The lack of explicit feedback from a sample of customers introduced big difficulties to see customer satisfaction through the eyes of the proposed base model from 2.2 introduced in section 2.1. customer satisfaction is a much more complex construct and it turned out to that the app usage or usage days of the GPS device are not sufficient to make a promising statement. Although [39] came to the conclusion, after their survey results analysis, that mobile usage of customers in nowadays smartphone dominated world has a major influence on satisfaction, their proven hypotheses contained mobile engagement motiviation as primary factor. This engagement can be characterized as a three dimensional model consisting of the functionality of a mobile app driving a users efficiency to accomplish tasks, the ease of use and entertainment factor and a social component indicating whether it is possible to connect with friends via the app [61]. The results of [39] furthermore show that mobile engagement and perceived value can create a basic

satisfaction level for a user which in turn leads to more engagement motivation and as a result increases satisfaction. This mobile engagement- and satisfaction model is not well represented by the single dimension of app opening events. As a result the data analysis tasks explained in the previous part of this thesis underperformed.

The difficulty in finding a representative metric, which provides a better chance to get more accurate results, from the collected data at Tractive led to the following decision. In conjunction with the company the author decided to design a customer survey which will be sent to users who purchase and successfully activate a GPS device. The survey asks them a few questions to find out how satisfied they are with the product. An essential part was the type, formulation and number of questions to ask the customer in order to get representative knowledge. The goal was to incorporate the gathered data back into the software system, extract potentially satisfaction driving features as mentioned in section 3.2 for those customers and learn from this data. Identified relevant data sources should be reused and further extended in this part of the thesis but the approach should be a different one. The upcoming chapter 4 will shed more light onto the approach and describes the implementation in greater detail.

Chapter 4

Data-driven approach leveraging explicit feedback as target variable on Customer Satisfaction

Instead of using identified relevant features from the first approach to make predictions about a satisfaction level of customers, this approach should be fed with more features and find its way to the promising ones automatically. The goal was to work on a software framework which analyses data and selects the influential factors for Customer Satisfaction automatically while throwing away garbage data and redundant features. Furthermore, the lack of metrics in the data representing Customer Satisfaction identified as major issue during the first part of the thesis should be tackled by getting explicit feedback from customers on how satisfied they are. The expectation from this customer survey is that it provides a much more reliable quantitative measurement of the satisfaction level of a customer than any other kind of data Tractive collects in its database system. As described in more detail in the satisfaction theory part of chapter 2 understanding why a customer behaves in a certain way can be quite subjective, involve psychological factors and vary among different types of customers. Therefore it has to be admitted that this can hardly be extracted from some objectively collected data and the actual source of truth is to ask people themselves. With a sufficient number of survey answers the framework should first analyse all observations retrieved by leveraging descriptive statistics and evaluate chances to predict satisfaction for arbitrary customers, who have not filled in the survey. With specific features selected machine learning algorithms can then be trained and evaluated on the data to find a model which is suited to do automated predictions.

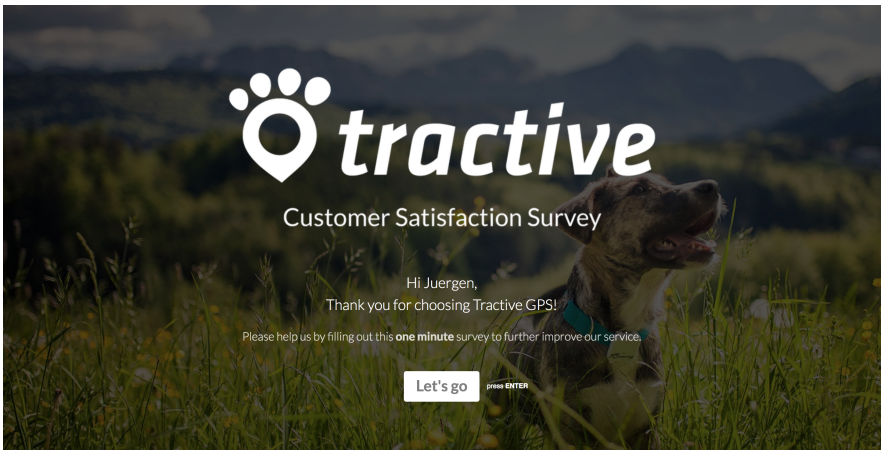


Figure 4.1: Customer Survey - Landing page

4.1 Implementation of a survey to gather explicit feedback from customers

Before any further data analysis task was started the initial task was to create a customer survey to collect usable information which gives a reliable insight into how a customer rates his experience with the product so far. A requirement was to keep the effort for the customer to fill in the survey low, which in first place means to require little time to fill in the duty part of the survey. Little time consumption correlates positively with response rate. According to [56] customer surveys are often too complex and as a result overwhelming for the user even though the company cannot derive better decision from it. With regard to customer satisfaction and loyalty [56] claim that the most essential number is the willingness of a customer to recommend the product to a friend or colleague. This metric should be on a scale of 0-10 to allow a computation of the NPS (Net-Promoter-Score), a widely used score to constitute growth of a company. In collaboration with responsible people at Tractive it was decided to limit the number of questions to the following two which were used later on to establish the target attribute in the prediction framework.

1. How satisfied are you with Tractive GPS? (Scale 1-5)

2. How likely are you to recommend Tractive to a friend or colleague? (Scale 0-10)

Next to the specifications of questions, the survey should be appealing, personalized and easy to use. The final landing page of the customer survey is shown in figure 4.1.

The goal of the company was to get customer feedback early on to be able to respond to unsatisfied customers quickly and in best case before they complain about a problem

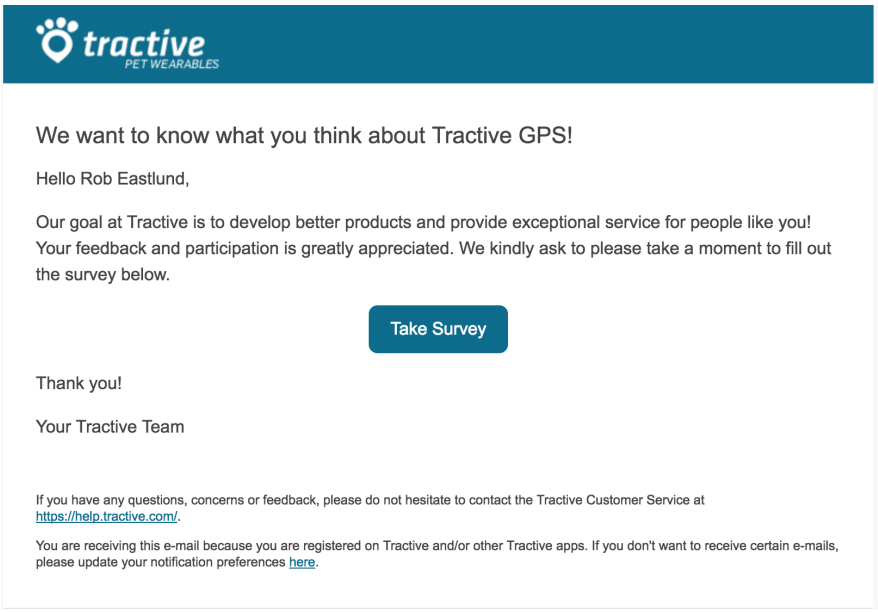


Figure 4.2: Customer Survey - Email

they have at customer support. As a result, the author of the thesis could arrange with the responsible people of Tractive to following procedure. When a new user converts successfully to a paying customer, he or she has to activate the GPS device before first usage. This enables the device to be actively used, sending positions and showing up on the map in the apps. Directly after activating the device, the backend service schedules an email in two weeks asking the customer to fill in the survey. To reach as many users as possible the survey was translated into the five major languages supported at Tractive, namely German, English, French, Italian and Spanish. Depending on the users stored demographical settings the correct language version is triggered. An example of how the email looks like is shown in figure 4.2.

When the user clicks on the ”Take Survey” button in the email, a browser window will open with the customer survey landing page. Behind the scenes In the URL (Uniform Resource Locator), the language version of the survey as well as the name of the customer is passed. For better personalization of the voluntary additional questions the name of the pet associated with the GPS device is sent as URL parameter. A necessary information is the unique id of the device since it allows to query all of the identified hardware- and position related data identified in section 3.2. For designing the customer survey the external tool Typeform was used. The following section describes briefly how the data gets extracted and integrated back into the internal database to be processable. Furthermore a few statistical details will be pointed out to get an impression with regard to the pending satisfaction analysis and prediction task.

Attribute	Datatype	Description
_id	ObjectId	Unique identifier of document
survey_id	String	Typeform identifier for the particular language version
submit_date	DateTime	Date and time when customer submitted survey
user_id	ObjectId	Reference to the user
tracker_id	ObjectId	Reference to the device
rating	Integer	Overall satisfaction (scale: 1-5)
recommendation_score	Integer	Recommendation potential (scale: 0-10)

Table 4.1: Structure of a survey response represented in the company database

Survey metric	Min.	1. Quartile	Median	Mean	3. Quartile	Max.	Variance	Standard dev.
Satisfaction	1	3	4	3.688	5	5	1.327	1.152
Recommendation	0	6	8	7.137	9	10	7.574	2.752

Table 4.2: Statistical summary - Overall satisfaction and recommendation score

4.2 Results and interpretation of survey results

To fetch the survey results from Typeform, a nightly job was implemented which uses the provided API of Typeform and gets the results from the previous day. The essential metrics from the two duty questions are extracted and along with the user- , pet- and device data stored in a new collection in the main database of Tractive. A typical structure of a document in this collection is illustrated in table 4.1.

The rating and recommendation score are the two interesting numbers when it comes to predicting satisfaction for an arbitrary user afterwards. The customer survey was deployed to the productive environment on 03.07.2017 which yielded the first survey result submitted on 17.07.2017. As of 28.10.2017 17:42 following statistics regarding the survey could be extracted.

- Number of customer survey emails sent: 15695
- Number of customers who filled in the survey: 2182
- Percentage of users who filled in the survey: 13.9%

As these numbers show, the response rate of the customer survey fortunately is quite good. The collected amount of customer survey data so far should be sufficient for finding potential patterns in the data. In order to get a better understanding some descriptive statistic values for the satisfaction rating and the value indicating willigness of a customer to recommend Tractive were calculated. The results are shown in table 4.2.

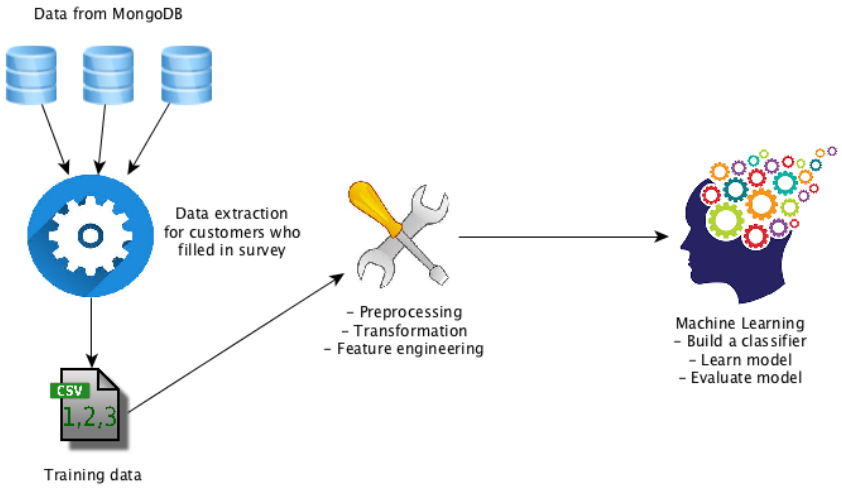


Figure 4.3: Overview on the prediction framework

Based on the results it can be followed that the average customer rates his or her satisfaction with the Tractive GPS product as mostly satisfied represented by a value close to 4. Similar is the recommendation likeliness value where the average lies between 7 and 8. Since the first quartile with a value of 6 is rather high, it can be followed that 75% of the customers are more likely to recommend Tractive to a friend or colleague. To close this statistical analysis of survey responses it can be stated that the majority of customers tend to be satisfied which had to be considered accordingly in the prediction framework outlined in more detail in the following section 4.3.

4.3 Software architecture and implementation of prediction framework

This chapter provides detailed insights into the architecture and implementation of a Customer Satisfaction prediction framework leveraging introduced data mining techniques from section 2.3.3. Based on the data extracted by the implemented tool outlined in the previous chapter, it was the goal to train a suitable machine learning model on this data of users who answered the customer survey and increase the certainty of satisfaction outcome. As common in nearly any machine learning task, intensive experimenting in order to achieve good results is inevitable. Therefore the author tried to automate those experiments. Before elaborating on the individual components of the prediction framework, figure 4.3 provides a broad overview.

4.3.1 Preparing training data

As the relevant data to analyze has already been queried with the implemented data extraction tool introduced in the last chapter, big parts of the implementation could be reused to prepare the training data for the prediction framework. However in contrast to the selection of random samples, it was necessary to limit the selection of the data to customers who had filled in the survey before. Therefore the first step was to query user and subscription data for those, including the user- and device ID in order to match the usage data queried later. In contrast to the hypotheses-driven approach where extracted data rows were confronted with each other to analyze them with statistical methods, the data was combined to large CSV file, which is the default input format for machine learning algorithms. Therefore, a data expansion function was implemented which passes gathered data to the next MongoDB aggregation task and combines queried data as new column in the CSV file. A short code extract of the combiner function is shown in listing **??**.

```
async.waterfall([
async.apply(user.insertUserData, customerSurveyData),
subscription.insertSubscriptionData,
customerSupport.insertCustomerSupportData,
device.insertDeviceData,
shareData.insertReceivedSharesForUsers,
shareData.insertSentSharesForTrackers,
resourceData.insertResourceData,
resourceSocialData.insertResourceCommentsData,
resourceSocialData.insertResourceLikesData,
notificationData.insertNotificationsData,
geofenceData.insertGeofenceData,
geofenceReportData.insertGeofenceReportData,
posReport.insertPosReportData,
idReport.insertNumberOfDaysInUse,
serverCmd.insertServerCmdMetricsData,
appUsageData.insertAppUsageData,
hwMetricsData.insertAverageBatteryLifeTime
], cb);
```

Listing 4.1: Expanding CSV data with results from MongoDB aggregation queries

As introduced in the code listings from section 3.3, the async library comes in handy here again to pass an array of JSON objects along each callback in a function. This way it can be used as ”hidden” input parameter in each function. Finally, the expanded JSON array gets written to a CSV file.

4.3.2 Process training data and learn a classifier

After having prepared the raw data, the empirical work continued with the implementation of the actual prediction framework. It was decided to develop a Java application and use the open source data mining software Weka implemented and maintained by the University of Waikato in New Zealand. Based on the theory of a comprehensive KDD process as outlined by [24] and [53], the software architecture of the framework was designed similar to a pipeline where the data flows through and gets modified at each station. The final step in this pipeline was to learn a model based on different classifier algorithms and evaluate them. The fact that survey results were collected only two weeks after a customer had activated a device, some of the attributes have invalid values among several data observations. Moreover the ratio between satisfied and dissatisfied customers is about 1.875 : 1. This led to a more difficult situation and required sophisticated methods in pre-processing. Figure 4.4 illustrates a sketch of the software architecture. Afterwards the individual parts shown in the figure will be explained. The thesis then dives into the implementation details with Java and Weka.

Before moving on, the essential Weka terminology should be briefly introduced to prevent any confusion afterwards. An in-memory data set in Weka is represented by the Instances object. A single data observation is called Instance and can be retrieved by calling dataSet.enumerateInstances(). Similarly, features can be accessed via dataSet.enumerateAttributes() whereby a feature in Weka terms is named Attribute. Next to a large set of different classification algorithms, Weka provides different useful filters for data pre-processing- and transformation as well as algorithms for feature selection.

4.3.2.1 FileProcessingEngine

The task of this first component is to read the CSV outcome from the data extraction tool and convert it into the required format, namely ARFF (Attribute Relation File Format), which Weka is able to work with. The ARFF format is very similar to CSV except that it contains metadata describing the type and value range of each feature (called attribute in terms of Weka). To read in the CSV file and return so called Weka Instances, representing the data set in-memory, the ConverterUtils from Weka were used. The resulting raw data set is then passed on to the pre-processing station. Furthermore, the FileProcessingEngine implements a method to write a current data set to an ARFF file which is used to store intermediate results. Listing 4.2 shows the implementation of those methods.

```
1 public static Instances readDataSetFromFile(String filename) {
```

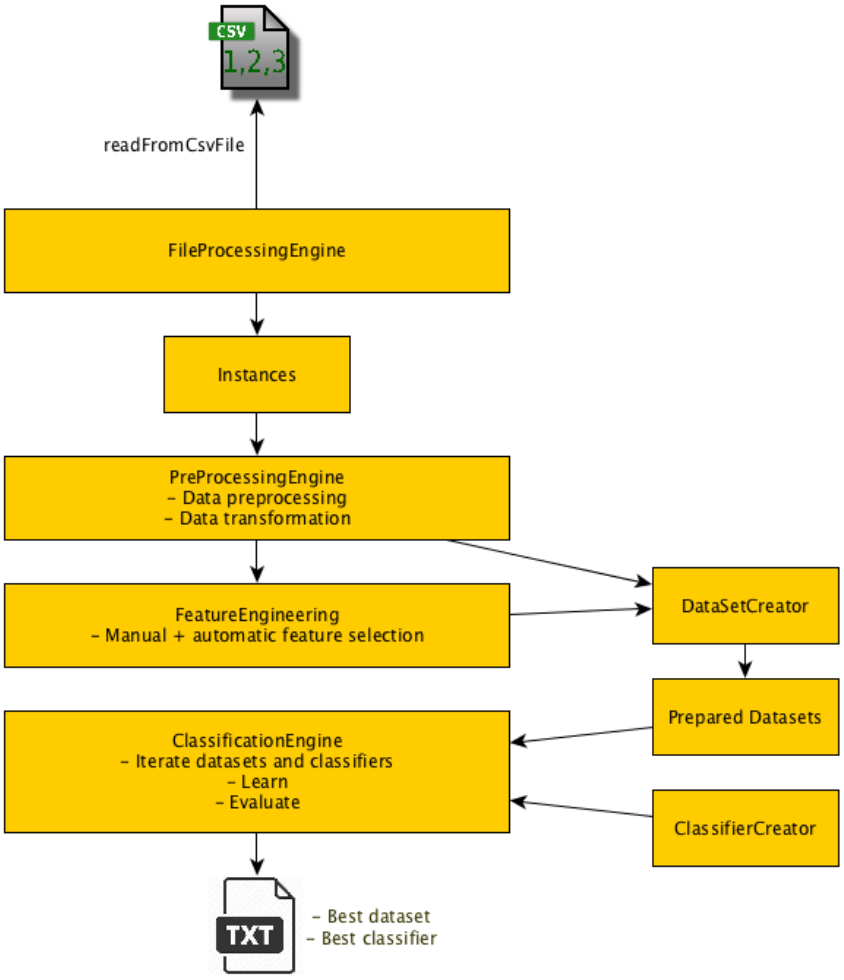


Figure 4.4: Software architecture of the prediction framework

Feature	Missing values
Support ticket metrics	68%
Battery lifetime	83%
Command duration	10%
Command delays	6%

Table 4.3: Missing values for some features

```
2   try {
3     DataSource source = new DataSource(filename);
4     return source.getDataSet();
5   } catch (Exception e) {
6     throw new RuntimeException("Data set could not be loaded from passed
          filename", e);
7   }
8 }
9
10 public static void writeDataSetToFile(Instances dataSet, String filename
      ) {
11   try {
12     ArffSaver arffSaver = new ArffSaver();
13     arffSaver.setInstances(dataSet);
14     arffSaver.setFile(new File(filename));
15     arffSaver.writeBatch();
16   } catch (Exception e) {
17     throw new RuntimeException("Data set could not be written to file",
          e);
18   }
19 }
```

Listing 4.2: Implementation of FileProessingEngine

4.3.2.2 PreProcessingEngine

The goal of this component is to prepare the data set in a way to build the best possible model when training a classifier at a later stage. Putting the raw data set from the FileProcessingEngine into a classification algorithm would lead to poor results as the quality of data for some features turned out to be bad. A view onto the statistics reveals that some attributes have a remarkable percentage of missing values. The features were grouped into smaller categories and are illustrated in table 4.3.

The reason why there are missing values for these features can be explained as following. As expected upfront there are not that many customers, who have contact with the company support team. Thus, 68% of customers have a ticket count of 0 and as a consequence missing support ticket metrics. Despite that, there is data available for

about 1/3 of customers who filled in the survey which allows some separate analysis later on. Since the calculation of hardware metrics started later, there is no battery lifetime data available for most of the devices from customers which gave satisfaction feedback. Although not that extreme, also the server command duration values are missing for some data observations most likely due to the short time span between activating the tracker and filling in the survey. This indicates that some users have not yet enabled live tracking at least once a time or did not have any successful live tracking so far. From these findings it can be concluded that missing data falls into the category of NMAR (Not-Missing-At-Random) as it was explained in the background research in section 2.3.4.2. Techniques proposed in the theory were implemented to handle those missing values. Following paragraphs take a look on the implementation details.

4.3.2.2.1 Filter unneeded features manually

Although the author could not exclude any particular features based on the statistical analysis from the previous chapter, the ticket metrics and the battery lifetime cannot contribute anything positive to learn a reliable model due to the large amount of missing values. They rather distort the learned model and thus have negative impact. Therefore it was decided to remove all features which exceed a defined threshold of missing values completely from the data set. With the RemoveFilter from Weka the features can be removed from the data set contained in the passed Instances object. As applying a Weka filter is a repetitive action, it was implemented as utility method in the Utils class. Code listing 4.4 shows an example of how to remove unneeded features which have more than 25% missing values.

```
1  public static Instances filterAttributesWithTooManyMissingValues(
       Instances dataSet) {
2    return filterAttributesNotFulfillingCriteria(dataSet, attribute -> {
3      int numberOfMissingValues = (int) Collections.list(dataSet.
           enumerateInstances())
4        .stream()
5        .filter(instance -> instance.isMissing(attribute))
6        .count();
7      float missingValuesPctg = ((float) numberOfMissingValues / dataSet.
           numInstances());
8      return missingValuesPctg > 0.25;
9    });
10 }
11 private static Instances filterAttributesNotFulfillingCriteria
12 (Instances dataSet, Predicate<Attribute> criteria) {
13   String[] attributeNamesToRemove = Collections.list(dataSet.
         enumerateAttributes())
14     .stream()
15     .filter(criteria)
```

```
16      .map(Attribute::name).toArray(String[]::new);
17
18    return filterUnneededAttributes(dataSet, attributeNamesToRemove);
19  }
20  private satic Instances filterUnneededAttributes(Instances dataSet,
        String[] attributeNames) {
21    Remove removeFilter = new Remove();
22    removeFilter.setAttributeIndicesArray(getAttributeIndices(dataSet,
          attributeNames));
23    return Utils.applyFilter(dataSet, removeFilter);
24  }
```

Listing 4.3: Remove unneeded features

4.3.2.2.2 Filter features with no variance

The variance of numeric features was investigated to filter extreme cases which show no variance at all. To overcome this problem those features were removed as well from the data set. The filterAttributesNotFulfillingCriteria method shown in the previous code listing was reused here with a different predicate function passed. Algorithm **??** shows how this was implemented.

```
1  public static Instances filterAttributesWithNoVariance(Instances dataSet
       ) {
2    return filterAttributesNotFulfillingCriteria(dataSet, attribute ->
3      attribute.isNumeric() && dataSet.variance(attribute) == 0.0);
4  }
```

Listing 4.4: Remove features

4.3.2.2.3 Imputation of missing values

As indicated earlier in this section, 5-10 % of the server command data collected showed missing feature values. In contrast to the cases considered before, these features could not be simply removed from the dataset as they deliver important information regarding Customer Satisfaction. The categorical analysis in section 3.4 showed for a small random sample a statistical significant evidence that the live tracking success rate influences customer churn. It was decided that this feature has to be kept for the data mining stage. As a result, the best possible technique was imputing the missing values accordingly. First of all, the imputation was done with the mean value of the feature under consideration. This method is available only for numeric attributes which was fine in this situation as the server command related data only consists of numeric features. Extracts of the implementation are shown in listing 4.6.

```
1  private static Instances replaceMissingAttributeValues(Instances dataSet
       , Set<Attribute> replaceableAttributes) {
2    if (!replaceableAttributes.stream().allMatch(Attribute::isNumeric)) {
3      throw new RuntimeException("Not all attributes to replace with the
           replacement value are numeric");
4    }
5    Enumeration<Instance> instanceEnumeration = dataSet.enumerateInstances
         ();
6
7    while (instanceEnumeration.hasMoreElements()) {
8      Instance instance = instanceEnumeration.nextElement();
9
10     for (int i = 0; i < instance.numAttributes(); i++) {
11       Attribute currentAttribute = instance.attribute(i);
12       if (instance.isMissing(i) && replaceableAttributes.contains(
             currentAttribute)) {
13         instance.setValue(currentAttribute, getReplacementValue(dataSet,
               currentAttribute));
14       }
15     }
16   }
17   return dataSet;
18 }
19
20 private static double getReplacementValue(Instances dataSet, Attribute
       attribute) {
21   return Collections.list(dataSet.enumerateInstances())
22     .stream()
23     .filter(instance -> !instance.isMissing(attribute))
24     .collect(Collectors.averagingDouble(x -> x.value(attribute)));
25 }
```

Listing 4.5: Mean and mode imputation of missing values

Next to the input data set the method takes as second parameter a set of features to replace with the mean value. First of all the method asserts whether all features to replace are numeric, otherwise an exception will be thrown. Within the subsequent while loop the features to replace are searched and the value gets substituted by a replacement value, calculated in getReplacementValue(dataSet, currentAttribute). This method iterates over all instances and calculates the arithmetic mean among the present values for the current feature.

Although this mean and mode imputation technique is widely used in data preprocessing for knowledge discovery processes, it has a downside of not considering feature values missing due to dependence on certain other features (i.e data is missing not completely randomly, see 2.3.4.2) and as a consequence could cause learning a biased model which in turn causes overfitting during the classification stage later on. There-

fore the author of this thesis decided to invest time in constructing a more sophisticated imputation method based on feature values from similar data observations instead of a global average. With the k-nearest neighbor imputation a promising technique was found in an analysis of methods for treating missing values correctly by [9]. The main part of the implementation is shown in code listing **??** and will be explained briefly.

```
1  public static Instances kNearestNeighborImputation(Instances dataSet,
       int k) {
2    Collections.list(dataSet.enumerateInstances())
3      .stream()
4      .filter(Instance::hasMissingValue)
5      .forEach(instance -> {
6        LinearNNSearch knn = new LinearNNSearch(dataSet);
7        Instances neighbors;
8        try {
9          neighbors = knn.kNearestNeighbours(instance, k);
10       } catch (Exception e) {
11         throw new RuntimeException("An error occurred while searching
               for nearest neighbors", e);
12       }
13       for (int i = 0; i < instance.numAttributes(); i++) {
14         if (instance.isMissing(i)) {
15           Attribute attr = instance.attribute(i);
16           if (attr.isNumeric()) {
17             double meanAttrValue = Utils.calculateMeanAttrValue(
                   neighbors, attr);
18             instance.setValue(attr, meanAttrValue);
19           } else {
20             String modeAttrValue = Utils.calculateModeAttrValue(
                   neighbors, attr);
21             instance.setValue(attr, modeAttrValue);
22           }
23         }
24       }
25     });
26     return dataSet;
27 }
```

Listing 4.6: Mean and mode imputation of missing values

After filtering all instances with missing values, the k nearest neighbors based on their similarity regarding feature values are determined. Therefore, Weka's kNearestNeighbours(instance, k) method was used. Subsequently, for each of the instances every attribute with a missing value gets iterated and replaced. Depending on the type of feature the replacement is done either with the mean (in case of a numeric feature) or the mode (in case of a nominal feature) among the k identified neighbors provided that their specific value is present. Otherwise, the particular neighbor instance is not considered.

4.3.2.2.4 Defining the Weka class attribute

After reading in the raw data from the CSV file, Weka does not know which of the features is used as class attribute needed to perform a classification task. The class attribute is the ground truth value for each data observation in the training set a classification algorithm uses for learning and evaluating a model. The important metrics provided by the survey results are the overall satisfaction level on a scale from 1-5 and the recommendation potential on a scale from 0-10. As the gathered experience from theory and hypotheses-driven testing at this point in time showed the difficulty of deriving relationships between certain features and Customer Satisfaction, it was decided to make the class value binary, namely satisfied or dissatisfied. With regard to business requirements it made sense to claim that below a rating of 4 customers tend to be dissatisfied while a value of 4 and 5 means that the customer tends to be satisfied. A similar machine learning approach for Customer Satisfaction, documented by [45], which was mentioned in the related work section did the same satisfaction level classification and thus confirms the thoughts. Therefore, in advance of setting the feature as class attribute it was normalized according to this satisfaction classification.

As alternative method it was tried to include the recommendation score into the satisfaction classification. This consideration is supported by literature work outlined in section 2.2.2. The satisfaction score was defined as the mean value of the overall satisfaction- and the recommendation score. 6.5 was set as threshold value to differentiate between satisfied and dissatisfied customers. After the satisfaction score had been brought into the desired format, the class attribute could be set for the current Weka Instances.

4.3.2.2.5 Normalization of numeric values

As the numeric features are a mix of different units, like percentages for the server command metrics, absolute numbers or timestamps, a successive classification algorithm would consider some features more important as others. In order to overcome the problem of different units, a linear scaling normalization of numeric values was implemented. As a result, all values are between 0 and 1. Since Weka provides a helpful filter, the normalization can be done conveniently. Code snippet 4.7 shows how to apply this filter.

```
1 public static Instances normalizeNumericAttributeValues(Instances
     dataSet, double minNormalizeValue, double maxNormalizeValue) {
2   Normalize normalizeFilter = new Normalize();
3   normalizeFilter.setScale(maxNormalizeValue);
4   normalizeFilter.setTranslation(minNormalizeValue);
5   return Utils.applyFilter(dataSet, normalizeFilter);
```

```
6 }
```

Listing 4.7: Normalization of numeric feature values

4.3.2.2.6 Transform nominal to binary values

Next to the numeric features, there are also attributes which are enumerations and can take on only a few values. In Weka these features are called nominal. They are not optimal since various classification algorithms can not deal with such enumerations well. The implemented solution transforms the nominal features into binary ones by splitting the given feature into as many separate features as values are possible. For each data observation either 0 or 1 gets associated for the particular transformed feature. Weka again provides a filter which automatically creates new features based on existing ones. Code listing 4.8 illustrates the implemented transformation.

```
1 public static Instances transformNominalToBinaryAttributes(Instances
      dataSet) {
2   NominalToBinary nominalToBinaryFilter = new NominalToBinary();
3   nominalToBinaryFilter.setTransformAllValues(false);
4   nominalToBinaryFilter.setBinaryAttributesNominal(true);
5   return Utils.applyFilter(dataSet, nominalToBinaryFilter);
6 }
```

Listing 4.8: Transformation of nominal- to binary features

4.3.2.2.7 Dealing with class imbalance

Depending on the dataset under consideration an imbalance between the ground truth values is a quite common scenario one has to deal with in classification tasks. A typical example related to class imbalance is the prediction of medical diagnosis cases where the clear majority in the sample is tested negative while a small minority is infected. Before having a look into the data details, the author did not pay any attention to the class distribution in particular but after spot checking a few promising machine learning algorithms on the pre-processed but yet imbalanced dataset, it turned out that the results are very bad and not useful at all. The class distribution of the dataset is shown in a simple bar chart illustrated in figure 4.5.

However, as indicated in the background research, different techniques to successfully deal with such imbalances have been researched. In the implementation part focus was paid in particular to the sampling methods. The initial attempt was to perform a simple undersampling on available training data by randomly taking only as many data observations from the satisfied customers to achieve a desired equality between

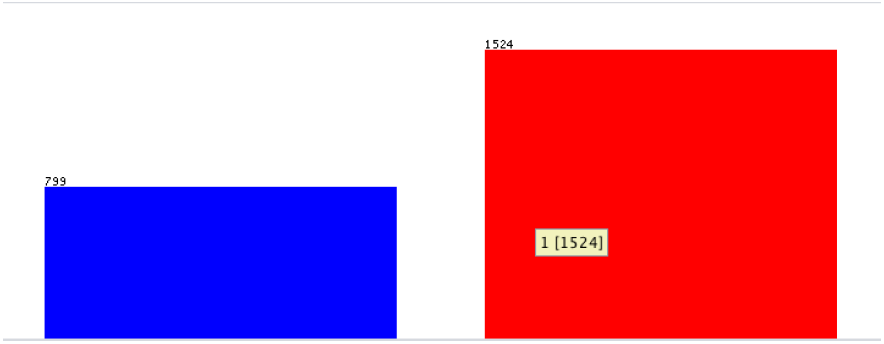


Figure 4.5: Bar chart illustration of class imbalance problem in customer dataset

the number of those observations and the dissatisfied ones. In order to give the method a better applicability in the framework, a factor indicating the percentage of minority- relative to majority class observations can be passed. The implementation is shown in listing 4.9.

```
1  public static Instances simpleUnderSampling(Instances dataSet, double
       minorityClassFactor) {
2    List<Instance> instancesMajorityClass = filterInstancesByClassValue(
       dataSet,1).collect(Collectors.toList());
3    int numRequiredMajorityInstances = (int) (instancesMajorityClass.size
       () * minorityClassFactor);
4
5    Collections.shuffle(instancesMajorityClass);
6    instancesMajorityClass = instancesMajorityClass.subList(0,
       numRequiredMajorityInstances);
7    dataSet.removeIf(instance -> instance.classValue() == 1);
8    dataSet.addAll(instancesMajorityClass);
9
10   return dataSet;
11 }
12 private static Stream<Instance> filterInstancesByClassValue(Instances
       dataSet, double classValue) {
13   return Collections
14     .list(dataSet.enumerateInstances())
15     .stream()
16     .filter(instance -> instance.classValue() == classValue);
17 }
```

Listing 4.9: Implementation of a random undersampling method

Although the class imbalance is not as extreme as in other use cases, like data dealing with the prediction of a disease, the random undersampling approach can be considered as an rather trivial algorithm to resolve class imbalance. Due to the reason that the top-down analysis showed that the quality of the data with regard to Customer Satisfaction prediction is not very good and the available dataset is not that huge, throwing away a few hundred observations could remove valuable information for the classification

task. Therefore this thesis searched for an oversampling method as alternative. As outlined in respective literature, the difficulty in oversampling is the generation of new observations. Although a combination of simple undersampling and duplication of observations belonging to the minority usually leads to improved classification accuracy on a validation set, it is often affected by overfitting to the learned model. Thus, the accuracy on real test set turns out to be completely different in contrast to the training phase. As a result of literature research, the focus was put on the implementation of the SMOTE algorithm to generate new observations fitting to a realistic distribution of provided data. The implementation of SMOTE is illustrated in pseudo code in listing 4.10.

```
1  Instances smote(dataSet, oversamplingPctg, k) {
2    if (oversamplingPctg < 100) {
3      chooseRandomSamples(dataSet, oversamplingPctg)
4    }
5    for i <- 0..dataSet.length {
6      nearestNeighbors = chooseNearestNeighbors(dataSet[i], dataSet, k)
7      syntheticObservations.add(generateSyntheticObservations(dataSet[i],
           nearestNeighbors))
8    }
9    return merge(dataSet, syntheticObservations)
10 }
11
12 Instances chooseNearestNeighbors(dataSet[i], dataSet, k) {
13   nearestNeighbors = []
14   for i <- 0..k {
15     nearestNeighbors[i] = euclideanDistance(dataSet[i], dataSet)
16   }
17   return nearestNeighborIndices
18 }
19
20 Instances generateSyntheticObservations(dataSet, observation,
       nearestNeighborIndices, k) {
21   syntheticSamples = []
22   for i <- 0..k {
23     for j <- 0..dataSet[i].attributesNr {
24       diff = observation[i] - dataSet[nearestNeighborIndices][i]
25       syntheticSamples[i][j] = observation[i] + random(0, 1) * diff
26     }
27   }
28   return syntheticSamples
29 }
```

Listing 4.10: SMOTE algorithm in pseudo code

4.3.2.3 Feature Engineering

The first step in limiting the feature space was done manually based on the fit of data sources to the proposed Customer Satisfaction model. Since there are still a lot of features in the preprocessed data set, the author decided to implement an automatic feature selection algorithm to find and skip redundant ones which potentially even diminish classification accuracy. The first algorithm is based on the intra-inter-correlation feature selection technique introduced in section 2.3.5. Therefore an abstract class called AttributeSelectionStrategy was implemented which generates a list of possible feature subsets and returns the best one depending on the concrete selection algorithm. An extract of this class is shown in code listing 4.12.

```
1  @Override
2  public Set<Attribute> getBestAttributeSet() {
3    List<Set<Attribute>> attributeSubSets = generateAttributeSubsets();
4    return getBestAttributeSubSet(attributeSubSets);
5  }
6
7  List<Set<Attribute>> generateAttributeSubsets() {
8    List<Set<Attribute>> attributeSubSets = new ArrayList<>();
9    ThreadLocalRandom random = ThreadLocalRandom.current();
10
11   for (int i = 0; i < maxIterations; i++) {
12     Set<Attribute> currentAttributeSet = new HashSet<>();
13
14     for (int j = 0; j <= random.nextInt(minSetSize, maxSetSize); j++) {
15       int randomAttributeIndex = random.nextInt(0, getDataSet().
             numAttributes());
16       currentAttributeSet.add(getDataSet().attribute(
             randomAttributeIndex));
17     }
18     attributeSubSets.add(currentAttributeSet);
19   }
20   return attributeSubSets;
21 }
22
23 abstract Set<Attribute> getBestAttributeSubSet(List<Set<Attribute>>
       attributeSubSets);
```

Listing 4.11: AttributeSelectionStrategy

The method getBestAttributeSubSet is implemented in a concrete class inheriting the AttributeSelectionStrategy and provides an implementation for the selection of the best possible attribute subset. In terms of correlation based feature selection, first each all subsets are iterated and for each one the overall correlation, consisting of inter- and intra coefficient, as outlined in section 2.3.5 gets calculated. The according implementation of this computation is show in code listing 4.12.

```
Correlation calculateCorrelation(Set<Attribute> attributeSubSet) {
  double interCorrelation = calculateInterCorrelation(attributeSubSet,
      getClassAttributeValues());
  double intraCorrelation = calculateIntraCorrelation(attributeSubSet);
  return new Correlation(attributeSubSet, interCorrelation /
      intraCorrelation);
}

double calculateInterCorrelation(Set<Attribute> attributeSubSet, double
    [] classAttributeValues) {
  return attributeSubSet
    .stream()
    .map(attribute -> Collections.list(getDataSet().enumerateInstances()
        )
      .stream()
      .mapToDouble(instance -> instance.value(attribute))
      .toArray())
    .collect(Collectors.averagingDouble(attributeValues ->
      new PearsonsCorrelation().correlation(attributeValues,
          classAttributeValues)));
}

double calculateIntraCorrelation(Set<Attribute> attributeSubSet) {
  List<Attribute> attributes = new ArrayList<>(attributeSubSet);
  double sumCorrelationCoefficients = 0;

  for (int i = 0; i < attributes.size() - 1; i++) {
    double[] valuesOfFirstAttribute = getDataSet().
        attributeToDoubleArray(attributes.get(i).index());
    for (int j = i + 1; j < attributes.size(); j++) {
      double[] valuesOfSecondAttribute = getDataSet().
          attributeToDoubleArray(attributes.get(j).index());
      sumCorrelationCoefficients += new PearsonsCorrelation().
          correlation(valuesOfFirstAttribute, valuesOfSecondAttribute);
    }
  }
  return sumCorrelationCoefficients / CombinatoricsUtils.
      binomialCoefficient(attributeSubSet.size(), 2);
}
```

Listing 4.12: AttributeSelectionStrategy

After calculating the overall correlation for each feature subsets, the results are then sorted in reverse order and the first one is finally picked as the best one. Subsequently the feature engineering super class filters the identified features within the best subset by applying the Remove filter from Weka on the non-selected ones.

As indicated in the theory of feature engineering, the learner based feature selection can be a powerful technique. Although it has the disadvantage of higher runtime complexity, especially with an increased number of features and also is bound to a specific classification algorithm type, the author decided to give it a try and check whether results can be improved. For implementation the WrapperSubsetEval class provided by Weka was used. Due to time constraints, for the learner evaluation typically a fast classifier had to be used. As mentioned in the background research on classification algorithms in section 2.3.6, decision trees are rather trivial and therefore fast to build, the C4.5 Java implementation called J48 was selected as base learner. How this learner based feature selection was configured and applied with Weka is illustrated in code extract 4.13.

```
1 public static Instances learnerBasedFeatureSelection(Instances dataSet)
     {
2   AttributeSelection attributeSelectionFilter = new AttributeSelection()
       ;
3   WrapperSubsetEval learner = new WrapperSubsetEval();
4   learner.setClassifier(new J48());
5   attributeSelectionFilter.setEvaluator(learner);
6   attributeSelectionFilter.setSearch(new BestFirst());
7
8   return Utils.applyFilter(dataSet, attributeSelectionFilter);
9 }
```

Listing 4.13: AttributeSelectionStrategy

The last attribute to be set in the attributeSelectionFilter is the method indicating how the algorithm searches along the feature space and generates subsets. The best first technique, introduced in the background research in section 2.3.5 was specified here. It was expected that it ensures a high quality of selected features and adheres time constraints.

4.3.2.4 ClassificationEngine

The final part in the Customer Satisfaction prediction framework was the training of classification algorithms on the preprocessed, transformed and feature engineered data set. As revealed in literature research pointed out in section 2.3.3, there is no ultimate classifier for such a type of data set. Although Weka provides a GUI called Weka Explorer which allows fast application of classifiers on a data set, it is only intended for manual experiments. Thus, the Explorer was only used a few times in the beginning to get a feeling on the development of prediction accuracy and getting an overview on the provided evaluation measures. Instead, the goal was to implement a program, which gets passed different versions of preprocessed data sets and a set of classification

algorithms as input, and finds the best combination of data set and classifier based on a representative measure.

In order to realize this intention, a new class called DataSetCreator was introduced. This class first of all is responsible for configuring necessary parameters regarding preprocessing, data transformation and feature engineering. The main functionality of the DataSetCreator is to use a passed configuration and generate the desired data sets as output. A code extract of this class is illustrated in listing 4.15.

```
1  private Instances createPreprocessedDataSet(Instances dataSet, boolean
       filterTooManyMissing,
2  boolean replaceMissingAttrValues, boolean markAllMissingNumericValues,
3  boolean replaceAllMissingAttrValues,
4  AbstractMap.SimpleEntry<Boolean, Integer> discretizeConfig,
5  SamplingConfig simpleUndersampling, SamplingConfig simpleOversampling,
6  SmoteSamplingConfig smoteOversampling) {
7    System.out.println("Starting to create a preprocessed data set");
8
9    if (filterTooManyMissing) {
10     dataSet = PreProcessingEngine.
           filterAttributesWithTooManyMissingValues(dataSet);
11   }
12   if (replaceAllMissingAttrValues) {
13     dataSet = PreProcessingEngine.replaceAllMissingAttributeValues(
           dataSet);
14   }
15   if (simpleUndersampling.isEnabled()) {
16     dataSet = PreProcessingEngine.simpleUnderSampling(dataSet,
           simpleUndersampling.getPercentageMinorityClass());
17   }
18   if (smoteOversampling.isEnabled()) {
19     dataSet = PreProcessingEngine.smoteOversampling(dataSet,
           smoteOversampling.getPercentageMinorityClass(), smoteOversampling
           .getNumNeighbors());
20   }
21   // more code ...
22   return dataSet;
23 }
```

Listing 4.14: DataSetCreator

Different preprocessing combinations were defined by the author. For all generated datasets the unneeded attributes which were mentioned in the PreprocessingEngine get removed upfront and nominal values transformed to binary ones as classification algorithms cannot deal well with pure nominal ones. A selection of data sets was done by using combinations of following configurations:

- Replace missing numeric values with the mean value of a particular feature

- Replace binary features with the mode value of a particular feature
- Discretize numeric values by putting them into 10 bins. (Due to the amount of configuration combinations and resulting time constraints, no further experimenting regarding the number of discretization bins was conducted)
- Undersampling the training set of satisfied customers by applying the random undersampling introduced earlier in this chapter.
- Oversampling the training set of dissatisfied customers by duplicating randomly chosen observations.
- SMOTE oversampling the training set of dissatisfied customers.

Regarding the sampling methods, the minority respectively majority class was sampled on a range starting with 70% and increased in 5% steps. As a result of generating data sets by these different configurations, a total number of xx data sets was created.

The next challenge now deals with the processing of these data sets by classification algorithms. Therefore a class named ClassifiersCreator was implemented. The main task of this class is to instantiate different classifiers, which were promised to perform well on similar Customer Satisfaction related problems introduced in literature. Moreover as baseline the author decided to use an algorithm called ZeroR in Weka which only relies on the class values and only predicts the majority class correctly. This classifier is not considered as relevant for the thesis but should only be used as benchmark for comparison. The actual classification algorithms discussed in more detail in section 2.3.6 were included in the framework.

Due to the reason that dissatisfied customers are more important to be reliably detected in order to tackle their problems early enough, the author decided to experiment with penalties for misclassification. Therefore Weka provides an abstract class called CostSensitiveClassifier. This class allows to pass a penalty matrix where the author specified a value ranging between 1.1 and 2.0 incremented by 0.1. The remaining step after initializing the penalty matrix is to wrap the base classifier. Code snippet **??** outlines the concrete implementation shortly.

```
1 private static Set<CostSensitiveClassifier>
     createCostSensitiveClassifiers(Set<Classifier> baseClassifiers) {
2   Set<CostSensitiveClassifier> classifiers = new HashSet<>();
3
4   baseClassifiers.forEach(classifier -> {
5     for (double penalty = 1.1; penalty <= 2.0; penalty += 0.1) {
6       CostSensitiveClassifier costSensitiveClassifier = new
           CostSensitiveClassifier();
```

```
7         costSensitiveClassifier.setClassifier(classifier);
8         costSensitiveClassifier.setCostMatrix(buildCostMatrix(penalty));
9         classifiers.add(costSensitiveClassifier);
10      }
11    });
12    return classifiers;
13  }
14
15  private static CostMatrix buildCostMatrix(double penaltyFalseNegatives)
        {
16    CostMatrix costMatrix = new CostMatrix(2);
17    costMatrix.setElement(0, 0, 0.0);
18    costMatrix.setElement(0, 1, penaltyFalseNegatives);
19    costMatrix.setElement(1, 0, 1.0);
20    costMatrix.setElement(1, 1, 0.0);
21    return costMatrix;
22  }
```

Listing 4.15: DataSetCreator

Having instantiated a set of Weka Instances, representing the data sets to be classified, and a set of Weka classifiers, the models can be built and evaluated.

Hence, the main method in the ClassificationEngine iterates over the data sets and trains every selected classifier on each of them. Based on literature research the preferred evaluation method is a 10-fold cross validation, which was used in the implementation of this prediction framework as well. Therefore the Weka Evaluation class with the method crossValidateModel was used. Evaluated classifiers get compared first by the AUC value, representing the area under the ROC calculated by integrating the respective function of the curve. As second measurement it was decided to consider the percentage of correctly classified instances as the preprocessing actively tackled the class imbalance issue. These values are provided by the Evaluation class after cross validating the learned model. If both values are high enough, the combination of configured data set and classifier gets stored in memory. After completing evaluation iterations, both the best- dataset and classifier gets stored with all details in separate files on disk for later analysis of the results (see chapter 5). The quintessence of the ClassificationEngine is illustrated in code extract 4.16.

```
1  public void classifyDataSets() {
2    double bestPctCorrect = 0.0;
3    double bestAuc = 0.0;
4    AbstractMap.SimpleEntry<Instances, Classifier> bestCombination = null;
5    Evaluation bestEvaluation = null;
6    long startTime = new Date().getTime();
7    int datasetNr = 1;
8
9    for (Instances dataSet : dataSets) {
```

```
10      for (Classifier classifier : classifiers) {
11        try {
12          classifier.buildClassifier(dataSet);
13          Evaluation evaluation = new Evaluation(dataSet);
14          evaluation.crossValidateModel(classifier, dataSet, NUM_FOLDS,
                new Random(1));
15
16          double auc = evaluation.weightedAreaUnderROC();
17          double pctCorrect = evaluation.pctCorrect();
18
19          if (bestCombination == null || (pctCorrect > bestPctCorrect &&
                auc > bestAuc)) {
20            bestCombination = new AbstractMap.SimpleEntry<>(dataSet,
                  classifier);
21            bestPctCorrect = evaluation.pctCorrect();
22            bestAuc = auc;
23            bestEvaluation = evaluation;
24          }
25        } catch (Exception e) {
26          System.out.println("Applying classifier " + classifier.toString
                () + " to dataset with relation name" +
27          dataSet.relationName() + " was not successful.");
28        }
29      }
30      System.out.println("Classifier evaluation for data set " + datasetNr
            + " took " +
31      Utils.timeDifferenceToString(new Date().getTime() - startTime));
32      datasetNr++;
33    }
34    System.out.println("Wohoo! All data sets evaluated!");
35    saveBestCombination(bestCombination, bestEvaluation);
36  }
```

Listing 4.16: Applying configured classifiers on preprocessed data sets

This chapter provided a thorough view onto the functionality and implementation of the automatic Customer Satisfaction prediction framework based on a data-driven approach. The outcome will be illustrated and discussed in the following chapter.

Chapter 5

Evaluation of classification results

This chapter provides insights into the results after applying the implemented prediction framework described in chapter 4.

The results shown were retrieved by a run finished on the 29.11.2017 at 09:50. Based on the different configurations to generate datasets, the prediction framework finally computed a data set containing 35 features (excluding the class attribute representing the ground truth) whereby 19 features are binary (transformed from nominal ones) and the other 16 are numeric. The data set contained 3082 instances reached by oversampling the minority class with the SMOTE algorithm. It generated 95% synthetic observations based on the previous number of instances. As a result the new training set contained 1558 instances representing dissatisfied customers and the original 1524 satisfied ones. Regarding missing values, the framework decided to impute all nominal features by the mode of the according feature while missing numeric values were imputed by the mean.

Following paragraphs compare the results from the evaluated classifiers with each other. Thereby, a clear differentiation between the two considered types namely, function-based- and rule-based classifiers was done. All following results were retrieved after applying a 10-fold cross validation on the learned model. The first evaluation was done on a support vector machine algorithm based on a polynomial kernel function. Similar to the weak results from the top-down approach, the SVM algorithm turned out to be not able to find an appropriate hyperplane to distinguish between the two classes of satisfied and dissatisfied customers. The results provided showed a weak classification accuracy of only 64.5036%. The important metrics are summarized in table 5.1 and will be briefly explained in the following paragraph.

As very remarkable for the author was the high discrepancy between the true-positive rates although the data set was balanced via SMOTE oversampling before. This prediction weakness can be illustrated quite well with the corresponding visualization of the

Classification	True positives	False positives	Precision	ROC Area
Dissatisfied	0.481	0.188	0.724	0.647
Satisfied	0.812	0.519	0.605	0.647

Table 5.1: Classification results for support vector machine

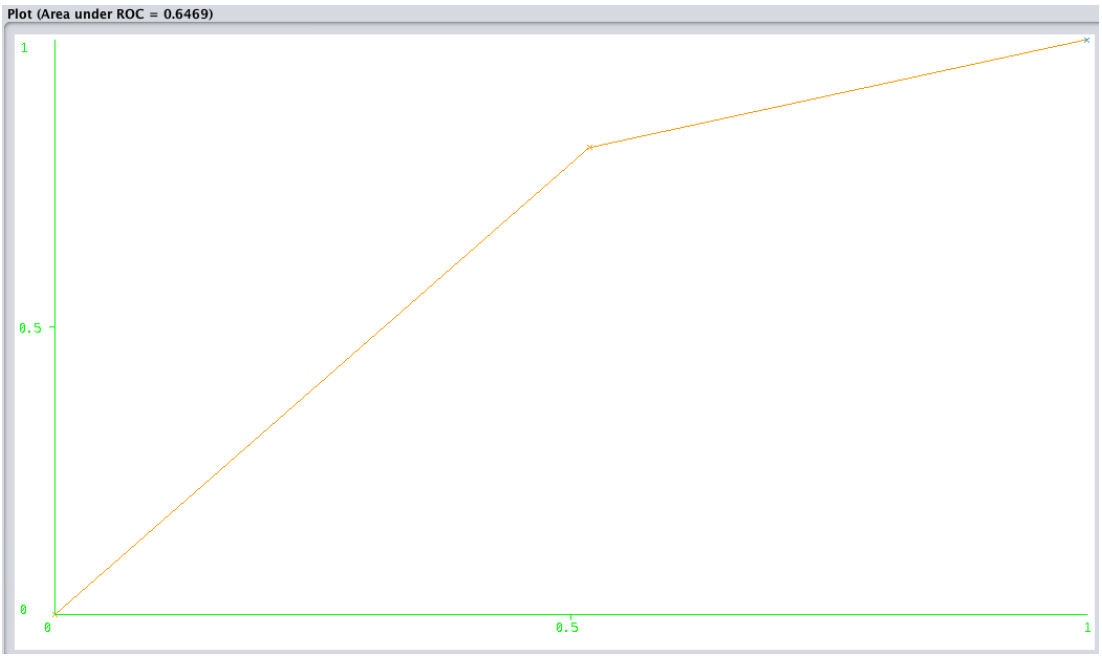


Figure 5.1: ROC curve for SVM

ROC curve in figure 5.1, showing the relation between true-positives and false-positives regarding classification of satisfied customers.

From the perspective of the author it can be followed that the SVM was very vulnerable with regard to the weak correlations of features with the class value and thus did not find any fitting hyperplane function in order to reduce the extremely high false-positive rate from a viewpoint of the satisfied class.

As outlined in the background research in section 2.3.6.1, a completely different approach is taken by decision trees where instead of mathematical functions, a rule set of how to split the training data based on the decisive power of features is built. After applying the standard decision tree implementation in Java, namely J48, the overall classification accuracy could increased to 65.7041%. Moreover the false-positive rates could be stabilized in contrast to the SVM which is a good sign. The detailed metrics are illustrated in table 5.2.

Although a decision tree can be considered as rather simple classification technique, it seems to fit the data set used in the case study of this thesis much better than the SVM. Moreover, learning the model is much faster as well. However, from an objective

Classification	True positives	False positives	Precision	ROC Area
Dissatisfied	0.696	0.383	0.650	0.677
Satisfied	0.617	0.304	0.665	0.677

Table 5.2: Classification results for J48 decision tree

Classification	True positives	False positives	Precision	ROC Area
Dissatisfied	0.847	0.375	0.698	0.847
Satisfied	0.625	0.153	0.8	0.625

Table 5.3: Classification results for Cost-sensitive Random forest

perspective the results are still not expressive as they are far too error prone and do not allow for a reliable prediction. As a result the framework decided also evaluated the decision tree variant called Random forest which was introduced in section 2.3.6.1.

The evaluation results of the Random forest algorithm showed major improvements in contrast to the standard decision tree and SVM algorithm. The basic configuration used for the random forest constructed 100 decision trees. For the preprocessed data set adhering the described configuration, a classification accuracy of 75.146 % could be achieved. Although this was the highest accuracy achieved, the AUC value indicating the area under ROC curve is worse than the evaluation with a cost sensitive meta classifier. As mentioned in the literature research, depending on the task the classification accuracy is not expressive enough. Since from a business perspective correctly detecting dissatisfied customers is essential, the number of false negatives should be as low as possible even though it sacrifices the false positives. As a consequence the appropriate choice taken by the author was to penalize false negative classifications with a cost of 1.5. This way the true positive rate with regard to dissatisfied customers could be established on an acceptable level. Due to this change the according false positive rate increased accordingly. Table 5.3 shows a summary of the important measurement results.

The corresponding ROC curve is shown in figure 5.2.

In order to verify how powerful the built model which produced the results from table 5.3 is, it was validated against a test set not included in the training phase. This should check whether and to which extent the model is vulnerable to the overfitting phenomenon, i.e. how good the classification accuracy can be expected on a real production data set. Therefore the full training set, which was used for cross-validation during the training phase, was split randomly into two separate files. The new training set contained 80% while the test set consisted of the remaining 20%. This approach was done manually via the Weka Explorer using the Resample filter. After loading the new training set and the trained model from the previous step into the Weka Explorer, the

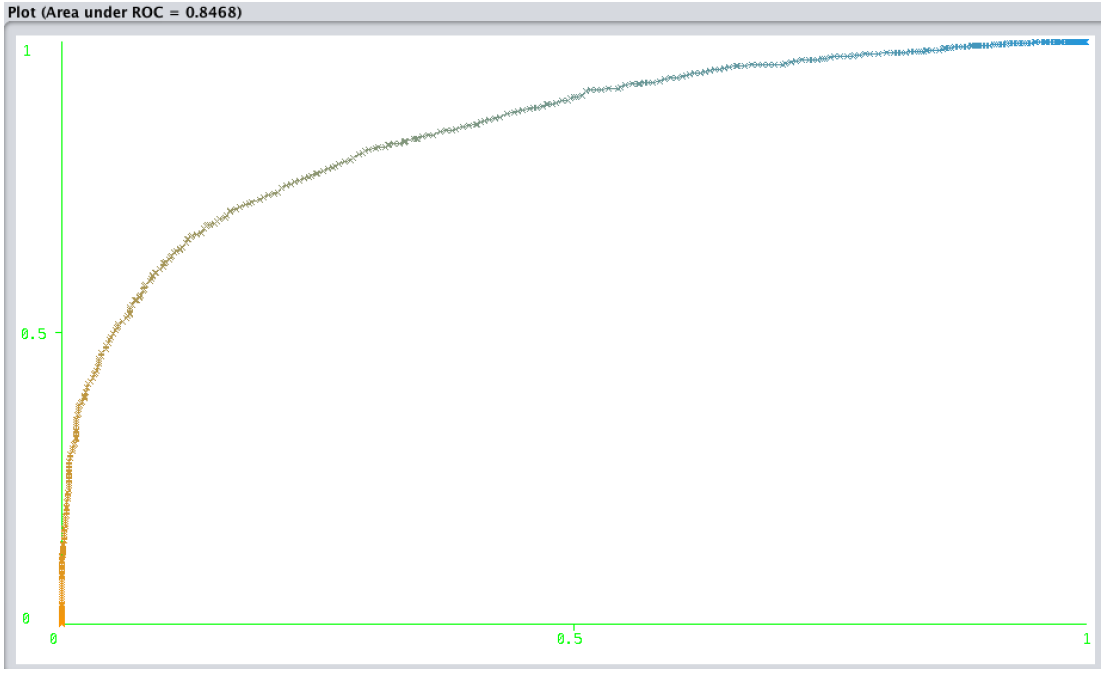


Figure 5.2: ROC curve for Cost-Sensitive Random Forest

Classification	**True positives**	**False positives**	**Precision**	**ROC Area**
Dissatisfied	0.779	0.354	0.667	0.804
Satisfied	0.646	0.221	0.763	0.804

Table 5.4: Classification results for Cost-sensitive Random forest evaluated on supplied test set

evaluation could be done against the provided test set. The results showed a slightly diminished overall accuracy of 70.928%. Table 5.4 shows again a more detailed overview on the important metrics.

Analyzing the results of the prediction framework led to following technical findings. Firstly, decision trees as base classifier and especially the ensemble algorithm Random Forest outperformed the Support Vector machines. Thus it can be followed that a rule-based classification approach suits the prediction of Customer Satisfaction based on usage features better than a functional approach like SVM. Regarding preprocessing and transformation of the data set, oversampling drastically improved the performance of the classifier. Generated data sets in the preprocessing phase which did not deal with the class imbalance explicitly, produced very bad results due to the reason that the given classification algorithm favored the class of satisfied customers which caused far too many false-positives. This shifted attention of the author to an intensive research and experimenting phase to tackle the class imbalance problem whereby the SMOTE algorithm finally turned out to be well suited since it could keep the performance in training high but did not overfit the trained model too much. Quite some effort was put on finding a preferred method to handle missing data. Although in the literature, mean

and mode imputation were sometimes criticized to introduce bias in the data and thus diminish power of classification, the prediction framework selected the rather simple but broadly used method when achieving the best results. This was a bit surprising to the author as the k-nearest neighbor implementation was considered as more sophisticated and also yielded better results in dedicated research experiments as outlined for instance in [9].

The following final chapter will conclude the thesis by wrapping up the work done in this research. Furthermore it draws consequences based on the findings and provides a look into future work.

Chapter 6

Conclusion and Future Work

The work in this thesis conducted methods of analyzing customer behavior, interaction data and product quality characteristics to detect a customers satisfaction respectively dissatisfaction automatically and on-demand. The contribution of this work should help software businesses, processing large amounts of data, to detect potentially dissatisfied customers at an early stage to tackle their issues or concerns pro-actively and as a result prevent them from churning or switching to a competitor. From a research perspective, the work shed light onto the complexity of the customer satisfaction construct and showed how collected data can be leveraged with scientific approaches to turn gathered knowledge about customers into a judgment of their satisfaction. Particular data insights and results of the work, revealed different questions and problems to be solved in future research work whereby some them will be mentioned at the end of this chapter. The following paragraphs briefly summarize the cornerstones of this thesis and draw essential conclusions.

In the first part, the objective was to gather enough knowledge on the topic of customer satisfaction itself. Based on the original definition of comparing a customers individual expectations with the perceived value of a product, different satisfaction models were analyzed and matched regarding its applicability with the concept of this thesis' case study, a pet tracking company with >100k customers. Based on a selected satisfaction model, an intensive analysis of data collections within the available database system of the case study was conducted to filter data sources considered as influencing factor for customer satisfaction. The implementation of an extraction tool brought the data in an appropriate form for analyzing it with regard to customer satisfaction. Two approaches with different philosophies were considered to have a detailed look on the data and find desired patterns. Firstly, the author started with a top-down analysis where the goal was to identify essential features driving customer satisfaction metrics based on business relevant hypotheses defined prior to the individual analysis task. Implicit data as the mobile app usage or the frequency of device usage were considered as hand-crafted customer satisfaction predictor. Using statistical tests those hypotheses were verified

respectively falsified. The thesis showed exemplarily how those statistical tests were applied. Due to the weak correlations in the data, it was necessary to gather explicit satisfaction feedback from customers to start with a data-driven knowledge discovery process. The extracted feedback from the conducted satisfaction survey was merged with the relevant data sources collected. With the knowledge of mature data mining research tools a framework was implemented which automatically preprocessed input data accordingly to overcome weaknesses in data quality, conducted feature selection techniques in addition to the manual feature construction done prior and learned a model based on machine learning techniques, like decision trees and support vector machines, which was then used for classifying customers by their satisfaction rating.

The final results retrieved from the data-driven approach, revealed at first glance a remarkable difference in classification power between the support vector machine- and the Random forest approach. Due to the limitations of this master thesis topic, no detailed insights into the reasons for the bad performance of SVM respectively the better suitability of the data set for a rule-based approach, were taken. The effort put into various preprocessing approaches but especially into the correct handling of the class imbalance problem were shown to be of major importance in a knowledge discovery process. Both, the statistical and data-driven analysis manifested the difficulty of finding evidence for customer satisfaction in collected behavioral data of a customer, at least in the given data by the case study. The achieved classification presented in the previous chapter shows that the implemented framework can provide a tendency about a customers attitude towards his or her satisfaction level but based on the current state there is a lot of room for improvement. The almost completely missing statistically significant correlations in the data emphasize the difficulty of mapping what makes up customer satisfaction to raw data.

Regarding the future, binding customers over a long term will continue to be essential for profitability of a business. Therefore, constantly monitoring customer satisfaction would be of great interest and help for businesses. Future research work has to put special emphasis on the fact of matching correct raw data onto satisfaction metrics. In addition to the focus on quality features, including emotional data of customers could lead to additional expressive features. Possible data could be for instance fetched via sentiment analysis from customer reviews or communication with the help center of a business. With regard to technical development, other feature engineering techniques like PCA (Principal Component Analysis) or automatic feature learning methods can be applied. In any way, the research area of data mining is still very busy and could bring further interesting analysis tools in the near future.

Bibliography

[1] Zendesk® api documentation. `https://developer.zendesk.com/rest_api/docs/core/ticket_metrics`. Accessed: 2017-11-30.

[2] Megha Aggarwal et al. Performance analysis of different feature selection methods in intrusion detection. *International Journal of Scientific & Technology Research*, 2(6):225–231, 2013.

[3] Selim Aksoy and Robert M Haralick. Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern recognition letters*, 22(5):563–582, 2001.

[4] Rolph E Anderson. Consumer dissatisfaction: The effect of disconfirmed expectancy on perceived product performance. *Journal of marketing research*, pages 38–44, 1973.

[5] R Artusi, P Verderio, and E Marubini. Bravais-pearson and spearman correlation coefficients: meaning, test of hypothesis and confidence interval. *Int J Biol Markers*, 17(148-151. PMID):12113584, 2002.

[6] Serkan Aydin and Gökhan Özer. How switching costs affect subscriber loyalty in the turkish mobile phone market: An exploratory study. *Journal of Targeting, Measurement and Analysis for Marketing*, 14(2):141–155, 2006.

[7] Emin Babakus and Gregory W Boller. An empirical assessment of the servqual scale. *Journal of Business research*, 24(3):253–268, 1992.

[8] Jonathan D Barsky. Customer satisfaction in the hotel industry: meaning and measurement. *Hospitality Research Journal*, 16(1):51–73, 1992.

[9] Gustavo EAPA Batista and Maria Carolina Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17(5-6):519–533, 2003.

[10] J Martin Bland and Douglas G Altman. The odds ratio. *Bmj*, 320(7247):1468, 2000.

[11] Ruth N Bolton. A dynamic model of the duration of the customer's relationship with a continuous service provider: The role of satisfaction. *Marketing science*, 17(1):45–65, 1998.

[12] John T Bowen and Shiang-Lih Chen. The relationship between customer loyalty and customer satisfaction. *International journal of contemporary hospitality management*, 13(5):213–217, 2001.

[13] Michael J Campbell, Steven A Julious, and Douglas G Altman. Estimating sample sizes for binary, ordered categorical, and continuous outcomes in two group comparisons. *BMJ: British Medical Journal*, 311(7013):1145, 1995.

[14] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[15] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[16] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*, 6(1):1–6, 2004.

[17] Injazz J Chen and Karen Popovich. Understanding customer relationship management (crm) people, process and technology. *Business process management journal*, 9(5):672–688, 2003.

[18] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[19] J Joseph Cronin Jr and Steven A Taylor. Measuring service quality: a reexamination and extension. *The journal of marketing*, pages 55–68, 1992.

[20] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent data analysis*, 1(3):131–156, 1997.

[21] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[22] A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. A gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

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

[24] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[25] Claes Fornell. A national customer satisfaction barometer: The swedish experience. *the Journal of Marketing*, pages 6–21, 1992.

[26] Claes Fornell, Michael D Johnson, Eugene W Anderson, Jaesung Cha, and Barbara Everitt Bryant. The american customer satisfaction index: nature, purpose, and findings. *the Journal of Marketing*, pages 7–18, 1996.

[27] Juliet M Getty and Kenneth N Thompson. The relationship between quality, satisfaction, and recommending behavior in lodging decisions. *Journal of Hospitality & Leisure Marketing*, 2(3):3–22, 1995.

[28] Anders Gustafsson, Michael D Johnson, and Inger Roos. The effects of customer satisfaction, relationship commitment dimensions, and triggers on customer retention. *Journal of marketing*, 69(4):210–218, 2005.

[29] Mark A Hall. Correlation-based feature selection of discrete and numeric class machine learning. 2000.

[30] Diane Halstead, David Hartman, and Sandra L Schmidt. Multisource effects on the satisfaction formation process. *Journal of the Academy of marketing science*, 22(2):114–129, 1994.

[31] Bob E Hayes. *Measuring customer satisfaction: Survey design, use, and statistical analysis methods*. ASQ Quality Press, 1998.

[32] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[33] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[34] Joe R Hulett. Exit, voice, and loyalty: Responses to decline in firms, organizations, and states, 1971.

[35] Rahim Hussain, Amjad Al Nasser, and Yomna K Hussain. Service quality and customer satisfaction of a uae-based airline: An empirical investigation. *Journal of Air Transport Management*, 42:167–175, 2015.

[36] Michael D Johnson, Anders Gustafsson, Tor Wallin Andreassen, Line Lervik, and Jaesung Cha. The evolution and future of national customer satisfaction index models. *Journal of economic Psychology*, 22(2):217–245, 2001.

[37] Michael D Johnson, Georg Nader, and Claes Fornell. Expectations, perceived performance, and customer satisfaction for a complex service: The case of bank loans. *Journal of Economic Psychology*, 17(2):163–182, 1996.

[38] Prashant Kadam and Supriya Bhalerao. Sample size calculation. *International journal of Ayurveda research*, 1(1):55, 2010.

[39] Young Hoon Kim, Dan J Kim, and Kathy Wachter. A study of mobile user engagement (moen): Engagement motivations, perceived value, satisfaction, and continued engagement intention. *Decision Support Systems*, 56:361–370, 2013.

[40] Bernd Knobloch. *Der Data-Mining-Ansatz zur Analyse betriebswirtschaftlicher Daten*. Otto-Friedrich-Univ., 2000.

[41] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186. Nashville, USA, 1997.

[42] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[43] Gordon S Linoff and Michael JA Berry. *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons, 2011.

[44] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[45] Stefan Meinzer, Ulf Jensen, Alexander Thamm, Joachim Hornegger, and Björn M Eskofier. Can machine learning techniques predict customer dissatisfaction? a fea-

sibility study for the automotive industry. *Artificial Intelligence Research*, 6(1):80, 2016.

[46] David Meyer and FH Technikum Wien. Support vector machines. *R News*, 1(3):23–26, 2001.

[47] Murray Moinester and Ruth Gottfried. Sample size estimation for correlations with pre-specified confidence interval. *The Quantitative Methods for Psychology*, 10(2):124–130, 2014.

[48] Michael C Mozer, Richard Wolniewicz, David B Grimes, Eric Johnson, and Howard Kaushansky. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on neural networks*, 11(3):690–696, 2000.

[49] Friedemann W Nerdinger, Christina Neumann, and Susanne Curth. Kundenzufriedenheit und kundenbindung. In *Wirtschaftspsychologie*, pages 119–137. Springer, 2015.

[50] Eric WT Ngai, Li Xiu, and Dorothy CK Chau. Application of data mining techniques in customer relationship management: A literature review and classification. *Expert systems with applications*, 36(2):2592–2602, 2009.

[51] Markus Oestreich and Oliver Romberg. *Keine Panik vor Statistik!: Erfolg und Spaß im Horrorfach nichttechnischer Studiengänge*. Springer-Verlag, 2009.

[52] Richard L Oliver. Effect of expectation and disconfirmation on postexposure product evaluations: An alternative interpretation. *Journal of applied psychology*, 62(4):480, 1977.

[53] Bernd Knobloch Peter Neckel. *Customer Relationship Analytics*, volume 2. dpunkt.verlag, 2015.

[54] Ved Prakash and John W Lounsbury. A reliability problem in the measurement of disconfirmation of expectations. *ACR North American Advances*, 1983.

[55] Michel Raymond and Francois Rousset. An exact test for population differentiation. *Evolution*, 49(6):1280–1283, 1995.

[56] Frederick F Reichheld. The one number you need to grow. *Harvard business review*, 81(12):46–55, 2003.

[57] Chris Rygielski, Jyun-Cheng Wang, and David C Yen. Data mining techniques for customer relationship management. *Technology in society*, 24(4):483–502, 2002.

[58] Henry Sauermann and Michael Roach. Increasing web survey response rates in innovation research: An experimental study of static and dynamic contact design features. *Research Policy*, 42(1):273–286, 2013.

[59] Graham J. G. Upton. Fisher's exact test. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, 155(3):395–402, 1992.

[60] Thanasis Vafeiadis, Konstantinos I Diamantaras, George Sarigiannidis, and K Ch Chatzisavvas. A comparison of machine learning techniques for customer churn prediction. *Simulation Modelling Practice and Theory*, 55:1–9, 2015.

[61] Kaan Varnali and Ayşegül Toker. Mobile marketing research: The-state-of-the-art. *International Journal of Information Management*, 30(2):144 – 151, 2010.

[62] Yaya Xie, Xiu Li, EWT Ngai, and Weiyun Ying. Customer churn prediction using improved balanced random forests. *Expert Systems with Applications*, 36(3):5445–5449, 2009.

[63] Atila Yüksel and Mike Rimmington. Customer-satisfaction measurement: Performance counts. *Cornell Hotel and Restaurant Administration Quarterly*, 39(6):60–70, 1998.

[64] Yu Zhao, Bing Li, Xiu Li, Wenhuang Liu, and Shouju Ren. Customer churn prediction using improved one-class support vector machine. *Advanced data mining and applications*, pages 731–731, 2005.