Electronic Markets
https://doi.org/10.1007/s12525-019-00351-0

RESEARCH PAPER

Supporting customer-oriented marketing with artificial intelligence: automatically quantifying customer needs from social media

Niklas Kühl[1] · Marius Mühlthaler[1] · Marc Goutier[1]

Received: 28 September 2018 / Accepted: 16 May 2019

Abstract

The elicitation and monitoring of customer needs is an important task for businesses, allowing them to design customer-centric products and services and control marketing activities. While there are different approaches available, most lack in automation, scalability and monitoring capabilities. In this work, we demonstrate the feasibility towards an automated prioritization and quantification of customer needs from social media data. To do so, we apply a supervised machine learning approach on the example of previously labeled Twitter data from the domain of e-mobility. We descriptively code over 1000 German tweets and build eight distinct classification models, so that a resulting artifact can independently determine the probabilities of a tweet containing each of the eight previously defined needs. To increase the scope of application, we deploy the machine learning models as part of a web service for public use. The resulting artifact can provide valuable insights for need elicitation and monitoring when analyzing user-generated content on a large scale.

Keywords Customer needs · Supervised machine learning · Twitter · Web services · E-mobility · Social information Systems · Marketing

JEL classification C38 · C81 · L94 · O32

Introduction

The identification and prioritization of customer needs is crucial for businesses in order to succeed in the market (Limehouse 1999). By knowing what the needs, wants and demands of (potential) customers are, commercially successful products and services can be designed and marketing activities can be coordinated (Srivastava et al. 1999). Commonly used methods to identify customer needs are interviews, surveys or observations (Edvardsson et al. 2012; Hauser and Griffin 1993). While these methods have proven to be successful, they lack automation and, thus, scalability and continuous monitoring capabilities. Depending on the scope, they can be very time- and cost-intensive.

On the customer side, it is nowadays common to share personal information in social information systems (SIS). SIS are information systems which rely on social technologies and open collaboration (Schlagwein et al. 2011), e.g. social media platforms like Twitter, Facebook or Instagram. As Perrin (2015) shows, 65% of all Americans and 76% of all American Internet users draw on such social media platforms—with a remarkable growth within the last ten years. In the group of young American adults (aged 18 to 29), already 90% use social media. A share of these social media instances contains valuable insights about the needs of customers (Misopoulos et al. 2014). With a high volume of social media, e.g. 500 million tweets (Twitter 2016) and 55 million Facebook status updates (Cuthbertson et al. 2015) per day, social media platforms present a promising data source to gain knowledge about customer needs in order to design new products and services.

This article is part of the Topical Collection on Social Information Systems Minitrack - HICCS 51

Responsible Editors: Rainer Schmidt, Rainer Alt and Selmin Nurcan

✉ Niklas Kühl
niklas.kuehl@kit.edu

Marius Mühlthaler
marius.muehlthaler@student.kit.edu

Marc Goutier
marc.goutier@student.kit.edu

[1] Karlsruhe Institute of Technology (KIT), Karlsruhe Service Research Institute (KSRI), Kaiserstr. 89, 76133 Karlsruhe, Germany

Published online: 15 June 2019

Researchers (Blindheim et al. 2016) and practitioners (Kühl et al. 2018) were typically relying on manual need elicitation. Only in the past years, first published examples consider modern social information systems like social media platforms (e.g. Twitter) as data sources (Misopoulos et al. 2014; Timoshenko and Hauser 2018). In previous work, an artifact has been presented capable of classifying tweets as to whether they contain customer needs (Kuehl et al. 2016). Such an artifact could be used in a more comprehensive approach, which automatically identifies and quantifies customer needs from micro blog data in general and Twitter data in particular. While the successful design of a machine learning based classifier artifact shows that the automatic identification of "need tweets" is generally feasible, one major limitation remains. The artifact so far is only able to identify tweets containing needs, but not the needs themselves.

In order to address this limitation, this work depicts a Design Science Research (DSR) cycle (Gregor and Hevner 2013), in which we apply a supervised machine learning approach to allow automatic need quantification, which means ascertaining the amount of instances for every given need category. Our research is guided by two research questions. First, we aim to investigate the design of a possible artifact by asking the question "How can we design a deployable artifact which is able to automatically quantify customer need tweets for a given domain?" (RQ1). Furthermore, we aim to show the feasibility of the resulting artifact by asking "Does the artifact achieve superior prediction performance for the domain of e-mobility, compared to a random guess classifier?" (RQ2). By answering these questions, the contribution of this work is fourfold. First, it provides an artifact which is able to quantify needs in a much more automated way than all current techniques (surveys, interviews, etc.)—and shows the feasibility of supervised machine learning for automated need quantification from Twitter. Second, it gives an overview of statistically well performing preprocessing steps and used parameters for a specific dataset in the domain of e-mobility. Third, it gives an insight of the different needs expressed on Twitter for our evaluation domain of e-mobility. Fourth, the artifact is deployed as a publicly available web service for further usage and integration into other analytical applications.

The remainder of this paper is structured as follows: After regarding related work, we present our DSR-based methodology, which determines the remaining sections. Guided by the phases of the framework by Kuechler and Vaishnavi (2012), we elaborate the awareness of the problem, suggest a three-phase iteration based on a machine learning process model, implement it and finally deploy the corresponding web service. We then evaluate the results to gain insights about the artifact's performance, discuss generalizability and finish with a conclusion.

Related work

Gaining insight from data which is voluntarily shared by people on the internet has been of soaring interest in the past years. Social media is a group of Web 2.0-based applications that allow the creation and exchange of user generated content (Kaplan and Haenlein 2010). As social media arrived in most people's everyday life, the amount of available information correspondingly increased (Perrin 2015). Not only has it become critical for many businesses to be aware of what their costumers expose on social media (Kietzmann et al. 2011), but also has the analysis of information been able to create value in many other areas. For instance, information systems have been developed to predict the outcome of political elections (Bermingham and Smeaton 2011), to be capable of predicting disease outbreaks (St Louis and Zorlu 2012), to forecast movements in the stock market (R. Chen and Lazer 2013), or to support crime fighting by analyzing a large amount of tweets together with their location data (Gerber 2014). A distinct aspect research has drawn significant attention to is the elicitation of opinions of customers out of customer reviews or social media data. Although a common use case of this so-called "opinion mining" (Lee et al. 2008) is the examination of costumer critiques of products they have purchased on e-commerce platforms, opinion mining has also recently investigated the potential of accumulating the opinions of users of social media platforms (Chen and Zimbra 2010). However, opinion mining does not necessarily include machine learning methods. One established method is to directly search for keywords which, for instance, determine the opinion of the writer of a product review (Srivastava and Sahami 2009). On the contrary, as a more sophisticated approach, machine learning systems for automated text classification have been developed immensely in the last years (Jordan and Mitchell 2015). Hence, a broad variety of different techniques and algorithms exist.

Machine learning are computational methods which focus on improving solving problems with computational resources by learning from experience (Mohri et al. 2012). A major distinction can be made between supervised and unsupervised machine learning. Whereas supervised learning can allocate new instances to previously defined target values, unsupervised learning refers to models built to discover new patterns and relationships (James et al. 2013). The two approaches have different application areas, which have been extensively investigated in previous research (Gollapudi 2016). For instance, Pang et al. 2002successfully implemented a supervised machine learning artifact to determine the sentiment of movie reviews. Turney (2002) conducted an unsupervised approach to predict whether reviews of different domains (e.g. movies, cars) recommend the products they are about. In contrast, the work at hand focuses on a supervised approach to allocate costumer needs from tweets, which—to the best of our knowledge—does not exist so far.

Research design

As an overall research design, we choose Design science research (DSR), as it allows to consider the practical tasks necessary when designing IT artifacts (March and Smith 1995) and has proven to be an important and legitimate paradigm in information systems (IS) research (Gregor and Hevner 2013). We follow the DSR process methodology and its individual phases according to Kuechler and Vaishnavi (2012). Since our approach relies heavily on implementation, we favor a clear differentiation between an abstract "suggestion" and a concrete, more programming-specific "development"—which the method of Kuechler & Vaishnavi provides and is, therefore, well-suited for the task at hand. In terms of knowledge contribution, the presented work is an "exaptation" according to Gregor and Hevner (2013), since we apply mature justificatory knowledge, i.e., supervised machine learning (Kotsiantis 2007), to the new problem of automated need quantification of social media data. The very nature of needs is complicated, as they include human emotions (Khalid and Helander 2006), different possibilities of (textual) expressions (Subramaniam et al. 2009) as well as different abstraction levels (Kotler and Armstrong 2001). The illustration of the feasibility of an automated quantification of needs from textual sources (like Twitter) poses a novel contribution and the resulting insights support the work of Müller and Bostrom (2016), who predict a rapid increase in the (cognitive) capabilities of machine-learning based artificial intelligence for the next decade. At the same time, the feasibility of the presented artifact is a proof point for the potential of intelligent social media analytics (Stieglitz et al. 2014). Additionally, by tackling the problem of automated need quantification with machine learning, several unique challenges arise, which we address in this work—allowing other researchers to benefit from in the future. This includes an adaptation of the supervised machine learning process to the specific problem of need quantification with Twitter data. Automated need quantification from tweets requires a reasoned choice of preprocessing steps and algorithm search spaces. We provide insights on how tweets should be preprocessed, and which algorithm search spaces are promising to support the automated need quantification.

In order to evaluate the artifact, we use a technical experiment as proposed by Peffers et al. (2012). We evaluate the statistical quantification performances of the identified models. Figure 1 presents the activities of this DSR cycle, separated into the steps of problem awareness, suggestion, development, evaluation and conclusion. This structure also determines the remainder of this paper.

Awareness of problem

Before going into detail about our chosen approach and its scope, we define the terms *customer need* as well as our evaluation domain of *e-mobility* to lay the foundations of the remaining work.

Foundations

There are three main scientific fields which do research on needs of customers: Psychology, marketing and information systems. In psychology, the focus of need research is centered around fundamental human needs—but does not take economic aspects into account (Sheldon et al. 2001). However, in the field of marketing, one of the most important challenges is to understand customer needs and ways to satisfy them. Kotler and Armstrong (2001) separate customer needs into three distinct categories (*needs*, *wants* and *demands*). Needs are often intangible—for example, the needs for mobility or financial security can be interpreted and satisfied in many ways. Therefore, individuals concertize them implicitly by transforming them into wants and demands. Additionally, Harding et al. (2001) outline that a customer need can also be expressed as a *requirement* of a product or service. In the field of information systems, research on "requirements engineering" states that a need is considered as a high-level requirement which has to be transformed into low-level requirements to find ways of fulfilling the need (Hull et al. 2010). For the purpose of this work, there is little to be gained differentiating between marketing-oriented customer need definitions (needs, wants, demands) or need definitions in the context of requirement engineering (high-level and low-level requirements): In a first step, any information about needs, regardless of the level of granularity, is valuable information. For simplicity, we, therefore, stick with the term *customer need*—taking all mentioned types into account.

For testing our approach in an application domain, we require candidate domains to be both dependent on fast and ongoing monitoring of arising needs and rich in Twitter data traffic. The domain of electric mobility (*e-mobility*) as defined in Scheurenbrand et al. (2015) fulfills both our requirements. We further have to narrow down the domain to a geographical area with a coherent set of laws and regulations, markets as well as socio-economic conditions. In addition, we require the Twitter data in a unique language, as we need consistent semantics to analyze. As a result of these requirements on the domain, languages like English and Spanish—which cannot be related to one region—are not suitable. Because of our familiarity with German, we focus on the German-speaking

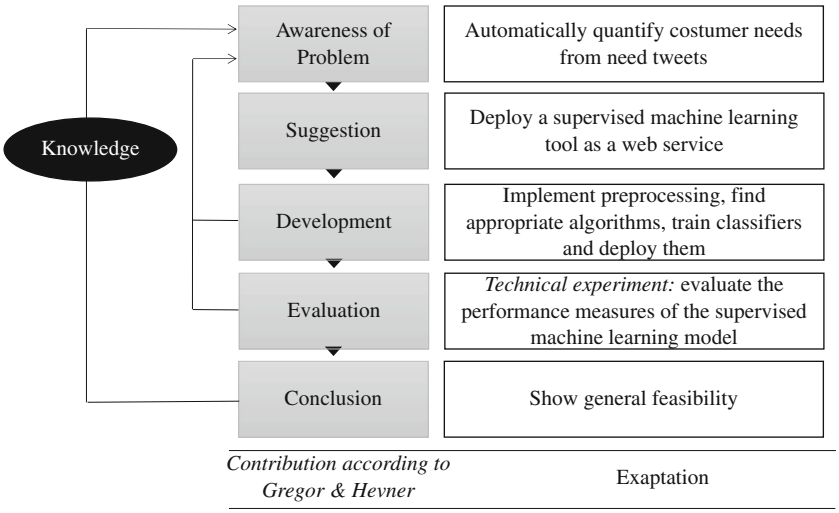

Fig. 1 Design cycle activities

region. While there is also plenty of research on the analysis of micro blog data for English, there is only few research on German instances—on social media in general (Maynard et al. 2012) and Twitter in particular (Cieliebak et al. 2017; Scheffler et al. 2015; Tumasjan et al. 2010).

Scope

While previous work shows the feasibility of identifying whether a tweet contains a need, the need itself remains undetected. Aiming for an information system in social context, which is able to automatically identify and quantify customer needs from Twitter data, it is crucial to not only identify the pure existence of a customer need—but more precisely be able to display the expressed needs in an aggregated version as depicted in Fig. 2, e.g. for a marketing manager.

Two possibilities on how to tackle this problem arise: supervised and unsupervised approaches. While supervised approaches rely on knowledge about the needs ex ante, unsupervised approaches do not require any knowledge about the needs beforehand. Such approaches, if fully working in practice, would allow to identify new needs without labeling, which a supervised approach relying on previously known needs would not be able to achieve. For the task at hand—quantifying customer needs from Twitter—an unsupervised approach would be favorable for innovation managers. In the context of innovation processes, the task of need elicitation would be typically overseen by an innovation manager (Oke 2007) and their focus would be on uncovering novel needs as a basis for innovative products and services. As a prerequisite for this study, we analyze different unsupervised clustering possibilities based on the findings of Hu and Liu (2004a, b,

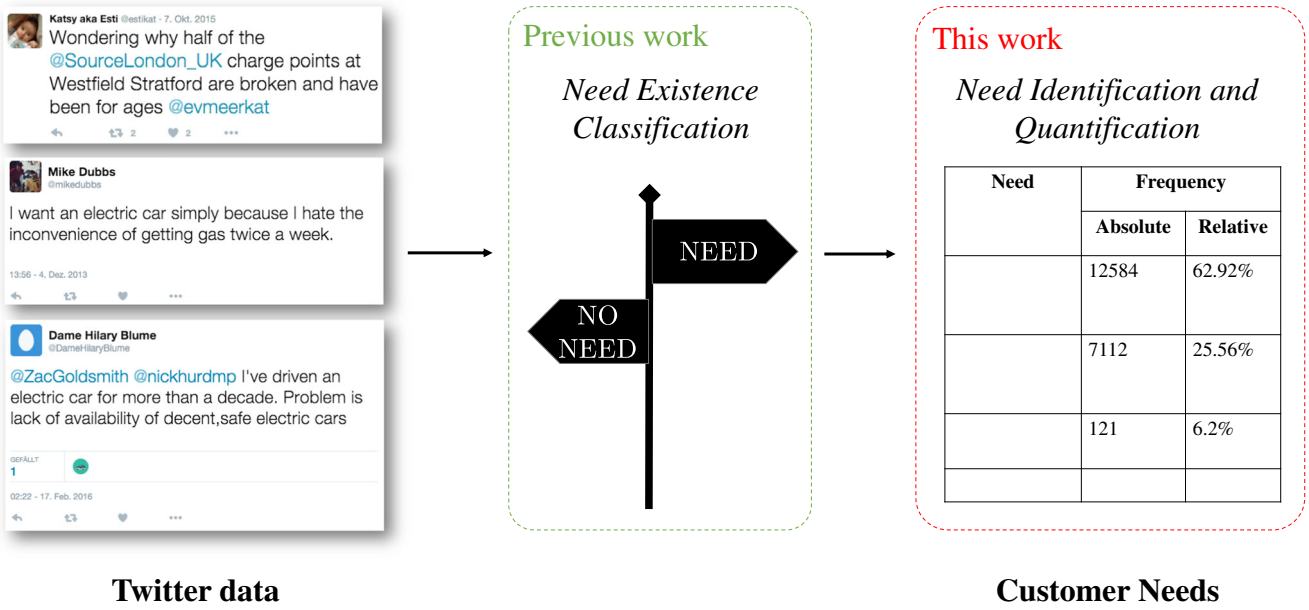

Fig. 2 Positioning of this work

2006), who were confronted with a similar problem in the area of opinion mining and suggest word vectors (Mikolov et al. 2013a, b). We combine their approach with a DBSCAN algorithm (Ester et al. 1996) with cosine similarity (Singhal 2001) for clustering on a data set of ~300,000 unlabeled German tweets in the e-mobility domain. However, over all tested parameter permutations, it is obvious that the results must be incorrect in a semantic sense. The resulting clusters do not actually represent customer needs or need categories. Furthermore, the silhouette coefficient, which can be used to estimate the quality of clustering results, is −0.2451, which indicates poor quality of the clustering results. Obviously, the amount of actual customer needs is too sparse for the proposed approach, since it is designed to work with way larger data sets as are both available and envisaged for the problem at hand.

In summary, unsupervised approaches have the advantage to not require any information beforehand, however, their feasibility for the desired problem remains unclear, and, furthermore, they rely on large amounts of data—which are not always available. We, therefore, explore the option to utilize supervised approaches for a specific evaluation domain as part of this work. In that case, a resulting solution would be of interest for marketing managers, as it allows a monitoring of (previously known) needs (Christopher et al. 1991). A resulting artifact, once trained on manually labeled data, would then be able to automatically classify new incoming tweets regarding their need—and allow to both monitor and quantify specific needs automatically—over any period of time.

Suggestion: A supervised machine learning approach

First, we have to select an overall process model for supervised machine learning. As we regard the special case of multiple classification and the need of deploying a fully-working prediction model, we choose the process model of Hirt et al. (2017), as it specifically addresses classification model initiation, its error estimation and deployment as illustrated in Fig. 3.

Model initiation

The model initiation starts with the acquisition of appropriate data and its labeling, followed by the selection of performance measurements to evaluate the machine learning process. Additionally, we decide how we preprocess our acquired data and choose a machine learning algorithm.

Data Acquisition & Labeling

The first step in the supervised machine learning quantification process is to gather relevant data which we later use for our model training and testing. These manual steps are necessary in supervised machine learning before the automatic need allocation can be implemented. The data should fulfill our requirements for source (Twitter), domain (e-mobility) and target value (need). Historic data of Twitter is not fully receivable (Twitter 2018). However, it is possible to receive the unfiltered stream of real-time tweets via the Twitter Streaming API (Bifet and Frank 2010). The only feasible way to acquire all instances (tweets) from Twitter is therefore to collect the data from the live stream and continuously store it. Since we are only interested in tweets in the field of e-mobility, we limit the collection of tweets from the data stream based on keywords representing our domain of e-mobility and, in our case, additionally restricted to a geological region and language. Next, we remove non-user generated tweets, e.g. from bots or news media. After filtering, the remaining tweets need to be classified according to whether the message itself contains a customer need. To obtain this piece of information in an objective way, independent participants take part in multiple lab labeling sessions. In these sessions, we instruct the participants to classify a set of tweets. They are incentivized and paid as described by Kvaløy et al. (2015), receiving a fixed payment. All participants are given the identical definition of *customer need*, as introduced in the foundations section. Different participants classify each tweet three times. The outcomes are aggregated for the generation of the final data set, regarding only tweets where at least two participants agreed it contains a need.

The remaining instances are fulfilling our requirements for source and domain—and we know that the instances contain needs. However, our target value is not the binary decision if one instance contains a need or not, as this is addressed in previous work (Anonymized 2016). We are rather interested which precise need or need category is mentioned in a tweet. Therefore, we have to label the remaining data again. We

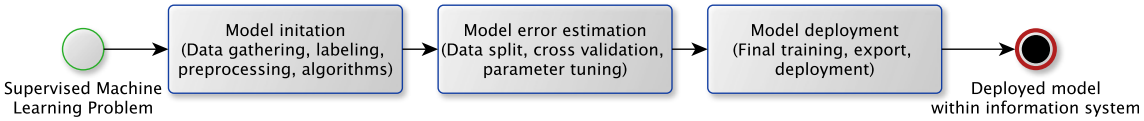

Fig. 3 Process model for supervised machine learning based on Hirt et al. (2017)

choose the method of descriptive coding as described by Saldaña (2015). Descriptive coding consists of multiple iterations in which researchers with domain knowledge analyze the tweet data, instance for instance, and assign codes to every tweet independently. These codes represent customer needs or customer need categories. The first iteration is focused on gaining an understanding about the data. We do not use any predefined list of customer needs. The labeling starts with assigning codes to an instance every time the instance contains a need. The number of codes equals the number of mentioned needs in the tweets. In case a need in one tweet is similar to a need in another tweet, we use the exact same code. An example of an instance which contains three different needs is represented in Table 1. If there is uncertainty about the meaning of a need or which code we should use, we tag the tweet with the *Other* code.

After finishing the first iteration, we compare our need categories, discuss and merge them for a common categorization. Based on the initial iteration we focus on three key features during later iterations: First, if we find any consistency based on disjoint of categories or needs, we will rework our codes from the previous iteration. Second, we aim to find broader, still mutually exclusive major need categories. However, we also aim at finding any sub categories of our last codes to describe the dataset in greater detail. In the end, we receive labels of needs, which represent the most specific coding level as well as major need categories as the broadest level of coding. Every need label belongs only to one major need category, whereas a major need category contains multiple need codes. Third, the category *Other* contains tweets after the initial iteration, so we try to find tweets with similar needs in the category *Other* to build new categories. Usually it is feasible to find new categories for most of the instances; however, it may occur that it is not possible to empty the category *Other* because of ambiguity of the mentioned needs or less similarities to other needs.

We continue the process until all researchers agree on the last coding as well as all key features are fully executed and fulfilled. In the end every need in every instance is labeled by a code which represents the need. Moreover, every code belongs to one major need category. We can now train a machine learning algorithm either on the need labels or on the major need category labels of the tweets. The only requirement for both options is to have a sufficient (Webb et al. 2001) amount of instances for every label in our dataset.

Performance measure

Before starting with the machine learning process, we have to choose an adequate performance measure. We weigh different possibilities. If we look from an overall statistical point of view, the area under the receiver operating characteristic curve (Area Under Curve, short: AUC) (Bradley 1997) is generally accepted as a meaningful classification indicator—and highly preferred over accuracy (Ling et al. 2003). Accuracy does not take class imbalances of the target class into account—but since we want to predict minority classes (5–20% share), accuracy is not specific enough, as it might reveal good results without actually learning the minority class well enough. Generally speaking, AUC represents the probability that a randomly chosen positive subject is correctly ranked with greater suspicion than another, randomly chosen negative subject (Hanley and McNeil 1982). AUC ranges between 0.5, which would be as good as a random guess, and 1.0, which would denote a perfect classifier. While AUC is statistically meaningful, it is still worth regarding precision and recall—as both can vary significantly with similar AUC results (Kuhn and Johnson 2013).

If aiming for the highest possible precision (and thus lowest fall-out rate), or aiming for the highest possible recall, both measures on their own are not helpful, but need to be regarded in combination. The balanced compromise between precision and recall is measured by the F_1-score (Powers 2011). As we expect marketing managers to neither miss out on relevant instances, nor get presented with wrong predictions, we choose the F_1-score as our core performance measure for this work.

Choosing Preprocessing & Algorithm

The previous data acquisition and labeling results in a dataset of tweets labeled with their needs, and every need belongs to one major category. Therefore, one general question is whether to build the models upon the discrete needs or only upon the major need categories. This mainly depends on the actual size of the dataset and the amount of needs or need categories found during the development part. We, therefore, discuss this question in the development. For readability reasons, we continue referring to *needs*, yet every step in the suggestion part can generally be applied to both single needs as well as broader major need categories.

Table 1 Exemplary real-world tweet with the assigned needs as a binary label (translated from German)

Tweet	Car price	Car design	Charging infrastructure	…
I will buy an electric car once the design is "normal", it is not more expensive than a common car and when the charging problem is solved	1	1	1	…
…	1	0	0	…

While every tweet is assigned to at least one, but up to multiple needs, we do not want our machine learning artifact to be bound to these combinations of needs, but rather to be able to identify every need independently. Therefore, we assign binary labels to the tweets, determining whether a tweet contains a particular need or not. Since we need a binary labeling basis in order to train a binary classifier for every particular need, we transform the data to be represented as depicted in Table 1. For the resulting web service, each tweet passes through each deployed classifier and the binary label of each classifier determines whether this tweet contains this particular need or not. Every step of the following process is hence conducted for each need separately. This consistently results in different independent classifiers.

Prior to training a classifier, we first need to find the well-suited preprocessing steps, sampling techniques and classification algorithms for this problem instance. Although the amount of feasible kinds of preprocessing methods is illimitable, we aim to systematically choose and evaluate a broad range of preprocessing steps. We consider the removal of words which do not contain any useful information (stop words) (Silva and Ribeiro 2003), the removal of words that appear very rarely or too frequently to be significant (frequency removal), combining *n* words into one feature (n-gram) (Cavnar and Trenkle 1994), downcasing or linguistic transformations such as stemming (Andrews and Fox 2007) or lemmatizing (Balakrishnan and Lloyd-Yemoh 2014). We use a bag-of-words concept to build a feature vector (Srivastava and Sahami 2009), where one tweet is represented as one feature vector. This vector contains, for all words in the dataset, the number of occurrences of words in this tweet. In order to extract the words from a tweet, different kinds of tokenization are taken into account. The tokenizers are distinguished in the manner of how they treat punctuation (e.g. emoticons) and in the amount of characters a token must at least have to not get removed. In addition, we include over–, under– and no sampling (Chawla 2005) into our collection of possible pre-treatments. We also consider whether to use a term frequency and inverse document frequency (tf-idf) transformer (Srivastava and Sahami 2009), which weights the words in the feature vector according to their relative frequency distribution. Table 2 shows an overview of the preprocessing. Another important aspect clearly is the choice of the classification algorithm. We examine both a Support Vector Machine (SVM) (Steinwart and Christmann 2008) and a Random Forest (RF) classifier (Breiman 2001).

Following the objective to find the most suitable combination of all of these elements, it would be best to try out every single permutation with regard to the final model performance. We, therefore, combine all these elements into a *factorial design* (Montgomery 2013). However, as the amount of different combinations and various values for each element is too high and would therefore be too computationally expensive, we split this step into two runs. In the first run, we gain an overview of the influence of the preprocessing steps on the F_1-score. We are then able to determine the ones that either have very little influence or the ones that always lead to the best scores across all needs. We keep them fixed for the second run, so that we can again try some new steps, together with the remaining variables in the first run that were not consistent in their influence on the F_1-score. We finish with comparing the results of each need. For consistency reasons, we ultimately determine one set of general preprocessing steps that are applied to all need classifiers.

Model error estimation

As an intermediate step between *model initiation* and *model deployment*, we conduct a *model error estimation* for each need separately, which later allows us to use the whole dataset for tuning the hyperparameters and training the classifiers we deploy, while still having a statistically safe model evaluation of every classifier. Tuning the hyperparameters means to find the optimal values for the parameters of a classification algorithm (such as the C and the γ for an SVM) that are not directly learned from the data during the model training. The traditional approach we use to accomplish this is a grid search, meaning to exhaustively search through all permutations of manually specified sets of possible values (Bergstra et al. 2011; Hsu et al. 2003).

The concept we choose for the model error estimation is a nested cross-validation (CV) (Cawley and Talbot 2010; Varma and Simon 2006), which we initiate by defining a parameter search space for the hyperparameters of the algorithm selected in the previous step. For the CV itself, we split the dataset into equal sized folds while maintaining the overall class distribution across all folds. In an inner CV, we tune these hyperparameters with a grid search, and use the hold-out set of the outer CV to gain information about the performance measures on unseen data of a classifier trained with the optimal parameters from the inner CV. This later enables us to perform a grid search with the same parameter space, while being positive that the model performance will be between the minimum and maximum value of the scores on every hold-out set of the outer CV. In the model deployment step, we, therefore, can omit a global test set and exploit the whole data set to perform a grid search (with the same parameter space as in the nested CV), and simply use the best performing parameters while still having an unbiased model error estimation.

Model deployment

As the previous step already results in an overall model error estimation, we can now use the whole dataset to perform a final grid search with the same parameter space as for the nested CV. We identify the best

Table 2 Overview of preprocessing options

Preprocessing step	Short description
stop word removal	remove irrelevant words, such as "are", "the", "get"
frequency removal	remove words that have a lower or higher frequency of occurrences than the specified threshold
n-gram	combine n words into one sequence
stemming	reduce words to their stem (root form), e.g. "apples" becomes "apple"
lemmatizing	replace words by their lemma, e.g. "gone" becomes "go"
tokenization	divide text into units, e.g. single words
oversampling	replicate samples of the minority class towards an equal distribution of both classes
undersampling	ignore samples of the majority class towards an equal distribution of both classes
tf-idf transformation	weight the words in the feature vector according to their relative frequency distribution

combination out of the parameter space using a grid search with a CV, and train the classifier on all the data with this exact combination of parameters. This classifier is now able to elicit needs out of a new tweet. Note that we perform these steps separately for each need, resulting in distinct classifiers for each need. For further usage, we aim our classifiers to be available for other users and for integration into other analytical tools. We choose to implement a web service which can use the persistently stored and trained classifiers. Different classifiers of different domains can be plugged into the web service, too. The web service provides one endpoint with two parameters, one for a new tweet and another one for the domain (e.g. e-mobility). Other services can then access this (remote) endpoint and the web service returns all the previously defined needs in the specified domain and their corresponding probability.

Being a lightweight web service, it is also practicable to integrate it into other more sophisticated tools. Figure 4 shows a proposal of such a holistic tool. Imaginable solutions could include other web services, for example one that can connect to the Twitter Streaming API (Bifet and Frank 2010), and pass the tweet onto a service capable of determining whether a tweet contains a need or not, and again pass all the need tweets onto the web service developed in this work. A database could store this information over a period of time. A user interface would then allow users (e.g. marketing managers) to analyze different aspects and, having stored information over a period of time, analyze the development of needs, or the impact of a marketing campaign.

Development: Model implementation

After we suggest an approach to gain labeled data, initiate a model, estimate its error and deploy it in a web service, we implement each of the individual steps in the following subsections.

Model initiation

We start with collecting tweets from Twitter and label the needs in the tweets by descriptive coding. Furthermore, we evaluate different preprocessing techniques and algorithms by a grid search to select well-suited preprocessing steps for our dataset.

Data Acquisition & Labeling

The technical retrieval of relevant tweets is not part of this work and is only explained briefly. We conduct the retrieval of relevant tweets by using the Twitter Streaming API (Bifet and Frank 2010). We collect every instance (tweet) which contains at least one word of a predefined keyword list. The list is reasoned on the opinion of professionals as part of a workshop and popular electric vehicles in Germany. It consists of eight German[1] and five English[2] generic terms which are supplemented by ten electric vehicles.[3] From March 2015 to May 2016 and from November 2016 to February 2017, over 2 million tweets are collected.

Based on the language information of Twitter, all non-German tweets are sorted out—which reduces our dataset to 107,441 instances. After the identification of user-generated content (Anonymized 2016), the dataset amounts to 6996 possibly relevant tweets. After multiple lab labeling sessions of our independent participants (50 business engineering students), we finally end up with 1093 remaining instances containing needs, which are only identified as such if at least 2 out of 3 participants agree on the tweet containing a need. This resembles the dataset of the work at hand.

The labeling and clustering of the tweets regarding their needs and major need categories starts with our first look at the 1093 instances. Researchers with knowledge about e-

[1] e-tankstelle, eauto, elektroauto, elektrofahrzeug, elektromobilitaet, elektromobilität, ladesaeule, ladesäule

[2] ecar, electric mobility, EV vehicle, e-mobility, emobility

[3] bmw i3, egolf, eup, fortwo electric drive, miev, nissan leaf, opel ampera, peugeot ion, renault zoe, tesla model s

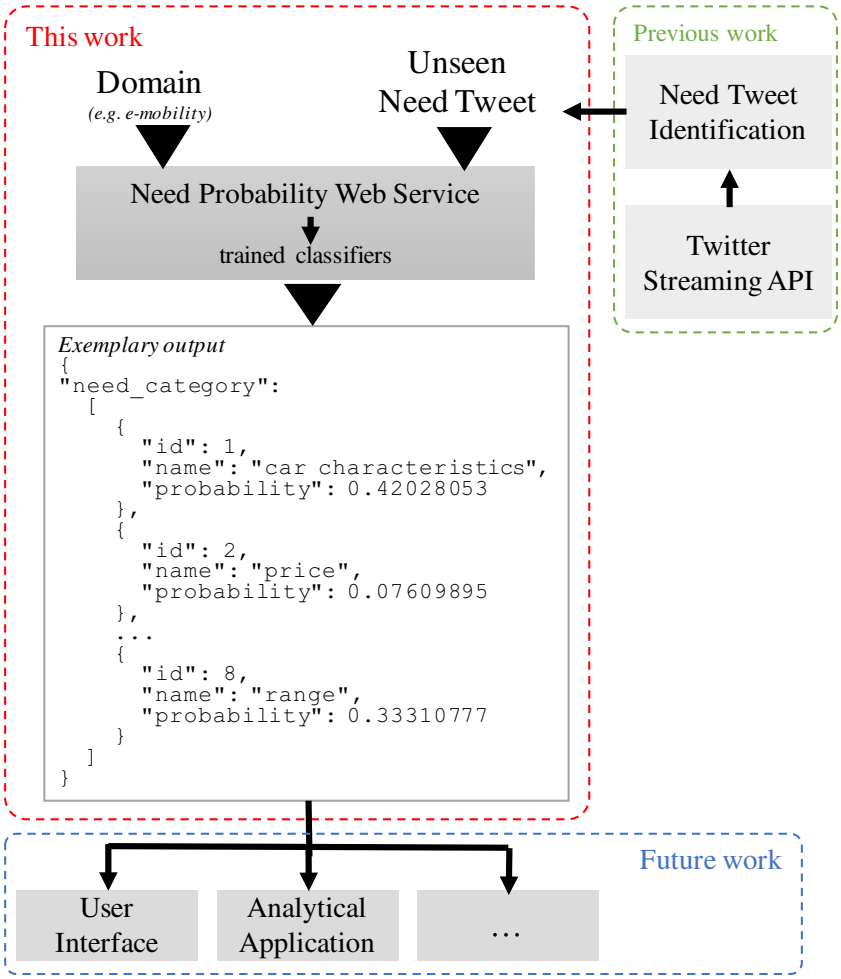

Fig. 4 Positioning of the web service in a holistic tool among other components of previous and future research

mobility, who were not part of the previous labeling process, conduct the descriptive coding (Saldaña 2015) as described in the subsection "Data Acquisition & Labeling". We perform five iterations of clustering until we reach saturation. In the end, we reveal seven major need categories, the category *Other* and 28 different needs which are depicted in Table 3.

To decide if we train our machine learning algorithm on the need labels or the major need category labels, we have to take the amount of instances for every label into account. Whereas some needs like *Car price* and *Politics* have a large amount of instances, six needs have only a single-digit amount of instances. The lowest number of instances for a major need category is 71, which represent slightly over 5% of all identified needs. Therefore, we decide to continue to train our machine learning algorithm on major need categories.

Choosing Preprocessing & Algorithm

In order to implement the previously suggested models, we use the Python programming language and the well-established machine learning package *scikit-learn* (Pedregosa et al. 2011). Once we have gathered and labeled the tweets, we develop a program to systematically evaluate and choose the best suited kind of preprocessing steps. We propose several different kinds of possible language processing methods: For word stemming, we use the *SnowballStemmer* (Porter 2001) for German of the package *NLTK* (Bird et al. 2009), a natural language toolkit for Python. Lemmatizing is done using *TextBlob* (Loria 2017), a Python package built on top of *NLTK*. We also develop own methods to replace emoticons, URLs and usernames with the words "emoji", "url", and "name" using regular expressions. Apart from the language processing, we take different tokenization methods into account. We use the standard *scikit-learn* tokenizer, which by default removes all tokens that consist of one character only and considers any kind of punctuation characters as word separators. This is the reason why we replace emoticons, URLs and usernames, instead of just removing them—as they would get removed by the tokenizer anyway. In addition to the standard tokenizer, we implement two variations of this tokenizer: One which does not remove tokens with the length "1" and another one which always removes tokens with less than "3" characters. Moreover, a

Table 3 Quantitative share of the major need categories and the needs for the regarded tweets, $n = 1093$

Major Need Category	Amount (share)	Need	Amount (share)
Price	202 (14.8%)	Car price	154 (11.2%)
		Electrical price	22 (1.6%)
		Price (other)	22 (1.6%)
		Oil/fuel price	4 (0.3%)
Car characteristics	145 (10.6%)	Car characteristics (other)	66 (4.8%)
		Car design	28 (2.0%)
		Car sound	19 (1.4%)
		Driving experience	15 (1.1%)
		Car comfort	7 (0.5%)
		Car performance	6 (0.4%)
		Car smell	4 (0.3%)
Charging infrastructure	305 (22.3%)	Charging infrastructure existence	191 (14.0%)
		Charging infrastructure availability (technical)	45 (3.3%)
		Charging infrastructure (general)	43 (3.1%)
		Charging infrastructure availability (physical)	26 (1.9%)
Range	135 (9.9%)	Range	135 (9.9%)
Charging technology	119 (8.7%)	Charging interfaces and technologies	55 (4.0%)
		Charging speed	30 (2.2%)
		Battery (other)	29 (2.1%)
		Range extender	5 (0.4%)
Environment & health	71 (5.2%)	Environmentally friendly car usage	39 (2.8%)
		Environment & health (other)	29 (2.1%)
		Environmentally friendly car production	3 (0.2%)
Society	283 (20.7%)	Politics	171 (12.5%)
		Desire for e-mobility	112 (8.2%)
Other	109 (8.0%)	Other (miscellaneous)	60 (4.4%)
		Definable	39 (2.8%)
		Joke	10 (0.7%)

TweetTokenizer from *NLTK* is used. When it comes to the removal of words containing very little information, we work with the German *NLTK* stop words. Other variables we take into consideration are the lowest and highest threshold of frequency removal, or the value for n in n-grams. Over– and undersampling is done using the Python package *imbalanced-learn* (Lemaître et al. 2017). For the tf-idf transformation of the feature vector, we use the built-in *tfidfTransformer* of *scikit-learn*.

Our goal is to determine which of these methods promise the best overall model performance. We, therefore, perform a 3-fold CV (with equal class distributions for each fold) with every reasonable combination of all of these methods, including the possibility to not use a step, such as no stop word removal. Since we cannot yet determine which classification algorithm to use, we perform it for both an SVM and a RF classifier—without tuning their hyperparameters at this point. Generally working with separate binary classification models, we run this process for each need category individually. As described in the Suggestion section, we split this part into two runs. We use random seeds whenever it is necessary, in order to be able to shuffle or randomly split the data, while maintaining comparability among the two runs.

In a first run, we consider whether to remove stop words, downcasing, removing words that appear in less than 1% or 5%, and in more than 50% or 75% of the documents, uni-gram and bi-grams, the *NLTK TweetTokenizer* against the *scikit-learn* default tokenizer, stemming, lemmatizing and different combinations of emoticon-, url- and name-replacement. For each need category and for each classification algorithm, we make a list of the ten best performing combinations in regard to the F_1-score. Based on a majority of occurrences in the top ten lists of the first run, we decide to always use stop words, lowercase, stemming, no lemmatizing, uni-gram, as well as emoticon- url- and name- replacement. The results are not consistent in terms of the classification algorithm, tokenizer and frequency removal, which we keep in the set of possible methods for the second run.

In the second run, we additionally try over- and undersampling as well as tf-idf/no tf-idf transformation of the feature vector. Again, for every need category, we calculate the performance (F_1-score) of all possible combinations and rank them. As we consider each need category as one binary classification problem, we have to perform this task for each need category separately.

Interestingly, the rankings are slightly different for each need category, meaning that each need category has its own set of preprocessing steps that leads to the best scores. Nevertheless, we are able to single out the best preprocessing combinations they have in common, in order to have a general preprocessing that we can apply to every tweet, no matter which need category it is assigned to. Clearly, as we generalize the preprocessing across all need categories, we cannot use the individual preprocessing that would be best for each need category on its own.

Table 4 shows, for each need category, the best scores achieved in the second run if the tweets of one need category were treated with their individually best preprocessing, in comparison with the scores achieved with the finally chosen, generalized preprocessing. In addition to the steps selected based on the first run, our pre-treatment in the end includes a tf-idf transformation and oversampling. We do not perform any frequency removal, however the best tokenizer removes all tokens with a single character. With this preprocessing, an SVM in terms of algorithm choice yields to better scores.

Model error estimation

To gain evaluation scores of the upcoming grid search, we define a parameter search space and perform a nested CV. This allows us to estimate the model error of the following grid search performed to tune hyperparameters of the SVM. Concretely, we optimize the C-value, the γ, the kernel and the degree (in case of a polynomial kernel). Like Hsu et al. (2003) suggest, we begin with a coarse parameter grid with exponentially growing values, and iteratively repeat the whole nested CV with a finer grid in order to explore the parameter space and to fine tune the model. After two runs of such a coarse data exploratory, we define the parameter set as $C \in \{0.5, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, $\gamma \in \{1.8, 2, 2.2, 2.4, 2.6, 2.8, 3\}$ and the kernel $\in$ {'sigmoid'}.

We consider the preprocessing elements from the previous step as given and use the same implementations. In an outer loop, we split the data into ten outer folds and perform a grid search on nine folds with three inner folds. This results in a *best parameter set*, which we use to train a new classifier. This classifier is then tested on the hold-out set of the outer loop. We measure the F_1-score of the trained classifier on the hold-out data set and store these scores in an external file, before we iterate through the outer loop. Ten outer folds result in ten iterations through the outer loop, each iteration with its own results on the hold-out data set. Lastly, we use the external file to determine the mean, the standard deviation and the minimum and maximum values of these ten scores.

Model deployment

After the model error estimation step, we perform a final grid search with the same parameter grid as in the previous step using the built-in *scikit-learn* grid search (Pedregosa et al. 2011) and train the final classifier on the whole data set. As we use an SVM as classification algorithm, this classifier would only be able to determine whether a tweet contains a need category or not. We hence take advantage of *scikit-learn*'s option to additionally carry out the Platt scaling algorithm (Platt 1999) in order to be able to calculate the *probability* of a tweet belonging to a need category. To store the classifiers for each need category persistently, *scikit-learn* suggests an additional package called *joblib* (Jovic et al. 2014), which makes it easy to store a fitted classifier into an external file. This file can then be loaded into any other Python program and be unpacked to exactly the same instance of the fitted classifier, with the same functions and attributes.

As we want to make the classifiers publicly available for integration into other tools, we store the classifiers on a web server. To make them accessible, we implement a RESTful

Table 4 F_1-score comparison of the individualized or generalized pre-treatment

Need	Best F_1-score with best pre-treatment for the individual need category	Best F_1-score with generalized pre-treatment
Price	0.595	0.583
Car characteristics	0.374	0.311
Charging infrastructure	0.727	0.707
Range	0.717	0.572
Charging technology	0.284	0.213
Environment & health	0.401	0.330
Society	0.538	0.418
Other	0.261	0.101

web service, using the Python *Flask* web framework. According to the REST architectural pattern, this web service provides an endpoint other tools can access via a simple HTTP GET request. The implemented web service provides an endpoint *GET /clustering* with two parameters *tweet* and *domain*. A client can then send a tweet and a domain to the web service, which will internally use the trained classifiers stored on the server, preprocess the tweet, and instantly return the need categories of that domain and the probability of which they contain this particular need category. An additional *GET /status* endpoint is provided, for the sake of delivering information about the current status web server and of errors that may have occurred.

The answers of the server are formatted in *JSON*, since it is an open and standardized format almost every other tool and database can easily deal with, while maintaining human-readability.

Evaluation

The evaluation is divided into three parts: First, we focus on the performance measures of the model error estimation and the final grid search. Second, we point out the real-world usability of the web service. Finally, we discuss the generalizability of the artifact for research and industry.

Model error estimation

As stated before, we conduct a nested CV for each need category to obtain a model error estimation of the possible classifiers. The ten outer folds of this nested CV result in ten different F_1-scores of different classifiers on unseen data. Table 5 shows the minimum, maximum, mean and standard deviation of these ten scores for each need category. The minimum and maximum values compose the confidence interval for the F_1-score of the finally deployed classifier. Table 5 also shows the baseline scores achieved by a random guess classifier. The scores of the grid search which determine the parameters for the final training on the whole dataset are depicted in Table 6. Indeed, as in the final training, we use the same parameter space as in the nested cross validation, the scores for each need category lie within the confidence intervals of the nested CV.

The scores for C*harging infrastructure, Range, Price* and S*ociety* show that it is possible to automatically allocate major need categories from tweets. Being the harmonic mean of precision and recall, a high F_1-score means that the classifiers are in most cases able to correctly identify whether a tweet contains a need category. This can certainly add substantial value to companies and help marketing managers in their decision making.

The results differ notably between the individual need categories. One influencing factor is that the class distributions for each need category vary: For the need category E*nvironment & health* for example, only 66[4] tweets have been labeled as containing a need category, in contrast to 298[4] for C*harging infrastructure*.

Different class distributions among the need categories are also one reason why they have (slightly) different *best parameter sets*, since the nested CV and the grid search in the model deployment phase are conducted for each need category separately. Indeed, the parameters are optimal for each need category, but—in addition to differing class distributions—different classifiers with different (optimal) parameters might behave differently and are another reason why the scores for each need category are heterogeneous. Nevertheless, further investigation on the influencing factors responsible for the varying results is required. The low score for the category *Other* is consistent with the fact that this category is of very diverse content—making it difficult for the classifier to find the right patterns which determine whether a tweet belongs to this category.

Model deployment

After manually performing the necessary steps like data gathering, labeling and deciding upon the preprocessing, we successfully implement a stand-alone web service that is capable of automatically allocating a tweet to the probabilities of each need category in the specified domain. The web service is running steadily and can be accessed on a web browser under a public url. An example of a JSON response can be found in the Appendix (Fig. 5 on page 33).

Based on this web service, new tools can be developed in the future. With the parameter *domain*, we allow future extensions to other domains besides e-mobility. In the future, classifiers for other domains are trained and can simply be stored on the web server. For other services or applications that invoke this web service, no adjustments are required since the parameter is already part of the API. The REST API makes it convenient to embed this service into more sophisticated analytical tools. A user interface for marketing managers, allowing them to analyze their costumer needs, could be developed. Interesting aspects, such as the distribution of costumer needs over time, could then be visualized. As the web service calculates the probabilities of containing a need category, an element empowering the user to define a "probability threshold" could be placed on the user interface. The

[4] In case one tweet contains multiple needs that belong to the same need category, only one instance is used in the implementation. Therefore, the effective frequencies of tweets containing need categories is slightly different than the total amounts shown in Table 3.

Table 5 F1-scores of the nested CV for model error estimation

Need	Min	Max	Mean	Standard Deviation	Baseline	Improvement
Price	0.524	0.737	0.642	0.059	0.264	+143.18%
Car characteristics	0.308	0.600	0.471	0.089	0.199	+136.68%
Charging infrastructure	0.719	0.871	0.783	0.043	0.353	+128.13%
Range	0.538	0.917	0.721	0.122	0.199	+262.31%
Charging technology	0.222	0.444	0.360	0.078	0.174	+106.90%
Environment & health	0.125	0.800	0.543	0.203	0.107	+407.48%
Society	0.452	0.746	0.543	0.080	0.333	+63.01%
Other	0.171	0.457	0.278	0.079	0.167	+66.47%

managers could then decide by themselves, "how certain" the estimates must be in order to be taken into account for the visualization. For them, this might be much more meaningful than the F_1-scores discussed in the previous section.

Discussion

While we show that supervised machine learning is feasible for automated need quantification in the domain of e-mobility, it remains to be discussed how this artifact could be applied in a more general way, especially in domains other than e-mobility. The underlying process model, preprocessing and algorithms of the artifact can be easily applied to other domains to generate other domain-specific machine learning models. However, there are several manual steps that need to be undertaken before a new artifact can be instantiated. First, experts have to agree on a list of keywords that is used to stream tweets for the domain of interest. Then, the tweets containing a need and subsequently the determination of the actual needs, as described in the "Data Acquisition & Labeling" section, is necessary. As a next design cycle of this overall project, the manual need identification and labeling could potentially be outsourced to crowd workers. This would allow automation capabilities to instantly instantiate new artifacts for a domain of interest, as the scalability and speed of crowd workers allow for fast labeling of data (Haas et al. 2015). First studies already show impressive results with semi-automated labeling of Twitter data (Finin et al. 2010; Garimella et al. 2016). Once the data is labeled, an automated model training and deployment could be easily implemented by providing an additional endpoint for the web service which has automated training and deployment capabilities (Grinberg 2018). Such an endpoint could accept the labeled tweets as an input and then trigger a process of automatic preprocessing, model training including the parameter tuning, as well as deploying the resulting artifact. This would result in a complete automation of the web service for any new domain of interest.

Besides generalizing the supervised approach, other solutions we applied could be meaningful for other purposes. Not only do we show a feasible approach to detect *m* characteristics ("needs") in *n* instances ("tweets"), also the instances itself can contain multiple characteristics. Comparable problems can be found e.g. in the field of object recognition, where a picture can also contain multiple objects of interest. We solve this issue by dividing the high-dimensional clustering problem into single, binary clustering problems. These tasks can be solved independently, which simplifies the clustering and enables parallel execution. Additionally, the identified useful preprocessing steps can be used as a starting configuration for other problems in the field of natural language processing (NLP) of Twitter data.

At its current state, the artifact can already be helpful for marketing managers. If integrated into a larger social media analytics tool for the elicitation of customer needs, it can continuously monitor the needs of customers or potential customers for the German speaking region in the domain of e-mobility. Based on these insights, it could be determined how existing products or services might be marketed in another way to address the expressed needs. As an example, our dataset shows a high frequency of tweets for *Charging infrastructure*—but with the expansion of charging stations and home charging (Nair et al. 2017), this specific customer need could become less relevant in the future. A marketing—

Table 6 Scores of the grid search that determined the parameter for the final training on the whole dataset

Need	Best F_1-score	C	γ	Kernel
Price	0.621	0.25	1.8	sigmoid
Car characteristics	0.462	0.50	1.8	sigmoid
Charging infrastructure	0.742	0.40	1.8	sigmoid
Range	0.717	0.10	1.8	sigmoid
Charging technology	0.386	0.10	2	sigmoid
Environment & health	0.513	0.35	1.8	sigmoid
Society	0.535	0.15	2.6	sigmoid
Other	0.284	0.10	2.6	sigmoid

or innovation manager—could see such changes immediately (or even use the data to forecast) on a long-term, mid-term or day-to-day basis and could quickly adopt marketing activities or even the products or services themselves accordingly. For instance, the effectiveness of marketing activities could be measured to track if certain need expressions decrease after, e.g., a new campaign ran. This could be a step change for the field of marketing management, since most traditional approaches of customer need identification and quantification are performed less often since they lack in scalability and automation capabilities. However, it always has to be considered that only needs expressed on Twitter are captured and the representativeness of the overall society is not necessarily granted (Barberá and Rivero 2015).

Besides marketing, which originally motivated the work at hand, continuous monitoring of social media streams for customer needs with an automated data collection, labeling, model training and deployment for any given domain could be used in many departments across a company. Tracking the numbers of customer needs in a specific domain helps the research department to set priorities for upcoming research. The product department could receive input on how to satisfy customer needs by looking deeply into suggestions of customers, expressed on social media. It would be also easily conceivable to use the number of unsatisfied customers for a specific product to predict future returns. Additionally, the warranty department could use data about customer needs which indicate a product failure to act before the customer claim their warranty rights.

Conclusion

In the work at hand we explore the option to automatically quantify customer needs from a predefined set. To achieve that, we first code over 1000 German tweets containing customer needs in the field of e-mobility with a descriptive coding approach. With this labeled data at hand, we choose the preprocessing, estimate the model error and train different supervised machine learning models for predicting the need category of incoming, unseen tweets. We encapsulate this functionality into a web service, which allows an automatic prediction of eight need categories for the e-mobility domain. If implemented into a larger analytical information system, this web service can assist marketing managers and alike in their daily operations of researching and prioritizing different marketing ventures. In conclusion, we can answer our originally-stated research questions by stating that the resulting artifact is well-feasible in terms of successful design implementation (RQ1) and statistical performance of the classifiers (RQ2).

This work has several limitations. We only explore a supervised approach and only do so for the field of e-mobility. Consequently, the customer needs are required to be identified once before the automatic identification can be enabled and non-identified categories cannot be revealed. This also results in the limitation that newly emerging needs cannot be automatically discovered in the continuous monitoring process. To address this major shortcoming, future work needs to investigate unsupervised approaches with large corpuses of data, preferably billions of tweets (Kalchbrenner et al. 2014). Additionally, new classifiers need to be trained if other domains are regarded outside of e-mobility. As the approach is of supervised nature, the manual need identification and labeling itself can be a costly endeavor—and possibly mitigate the efficiency gains of the application. Future work needs to address the economic feasibility of such an approach by comparing the costs of manual quantification with the automation gains of the approach. Another limitation is the small data set; for single categories the results of the error estimation show a high variance and, therefore, the possibility of model overfitting. Especially for unstructured text data from social media, the choice of the appropriate preprocessing steps (including feature selection and dimensionality reduction) is critical but also challenging. Although we evaluate a broad range of preprocessing methods, a more detailed evaluation on how to tailor the preprocessing of unstructured Twitter data to the particular use case is needed. In any case, this requires manual evaluation and decision-making.

The use of Twitter has limitations on the content and on the technical side. The used dataset might not reflect the opinion of a whole society because it might be biased to people with a more open-minded mindset towards technology or to younger people in general (Marshall et al. 2015). Apparently, the dataset is also biased to people using Twitter (Correa et al. 2010). On the technical side, we utilize the Twitter Streaming API, which allows us to receive all tweets containing the mentioned keywords as long as the fetched data does not exceed more than 1% of all tweets (Twitter 2018). However, when receiving the tweets, our results are limited to the selected keywords as well as the ability of Twitter to identify the language—since our language identification relies on the received meta data. By doing so, we cannot capture tweets which are written in mixed language (e.g. English and German) as well as tweets from Germans tweeting in a different language (e.g. English). Furthermore, as we only examine tweets in German, future work needs to explore the adaptability to other languages.

Nonetheless, we are able to show the feasibility of supervised machine learning to provide a solution for the challenge of automatable need quantification, which can provide businesses additional insight into the needs of their customer in the future. The exploration of further options, including unsupervised approaches of larger data sets, yields an interesting and promising field of research.

Appendix


```
{
  "need_category": [
    {
      "id": 8,
      "name": "Andere",
      "probability": 0.29098089526762805
    },
    {
      "id": 6,
      "name": "Umwelt & Gesundheit",
      "probability": 0.0007088399520436969
    },
    {
      "id": 2,
      "name": "Fahrzeugeigenschaften",
      "probability": 0.07239459132908911
    },
    {
      "id": 3,
      "name": "Ladeinfrastruktur",
      "probability": 0.051732475305165504
    },
    {
      "id": 5,
      "name": "Ladetechnologie",
      "probability": 0.4266037674424961
    },
    {
      "id": 7,
      "name": "Gesellschaft",
      "probability": 0.18743264918578326
    },
    {
      "id": 4,
      "name": "Reichweite",
      "probability": 0.2378019950835607
    },
    {
      "id": 1,
      "name": "Preis",
      "probability": 0.016416791142959127
    }
  ]
}
```


Fig. 5 Example of a JSON response from the deployed REST API

References

Andrews, N. O., & Fox, E. A. (2007). Recent developments in document clustering.

Balakrishnan, V., & Lloyd-Yemoh, E. (2014). Stemming and lemmatization: A comparison of retrieval performances. *Lecture Notes on Software Engineering, 2*(3), 262.

Barberá, P., & Rivero, G. (2015). Understanding the political representativeness of twitter users. *Social Science Computer Review, 33*(6), 712–729.

Bergstra, J. S., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems* (pp. 2546–2554).

Bermingham, A., & Smeaton, A. F. (2011). On using twitter to monitor political sentiment and predict election results.

Bifet, A., & Frank, E. (2010). Sentiment knowledge discovery in twitter streaming data. In B. Pfahringer, G. Holmes, & A. Hoffmann (Eds.), *Discovery science* (Vol. 6332, pp. 1–15). Berlin Heidelberg: Springer.

Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python: Analyzing text with the natural language toolkit. " O'Reilly Media, Inc."

Blindheim, J., Wulvik, A., & Steinert, M. (2016). Using secondary video material for user observation in the needfinding process for new product development and design. In *Proceedings of the DESIGN 2016 14th International Design Conference*.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition, 30*(7), 1145–1159.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Cavnar, W. B., & Trenkle, J. M. (1994). N-gram-based text categorization. *Ann Arbor MI, 48113*(2), 161–175.

Cawley, G. C., & Talbot, N. L. C. (2010). On over-fitting in model selection and subsequent selection Bias in performance evaluation. *Journal of Machine Learning Research, 11*, 2079–2107 Retrieved from http://jmlr.csail.mit.edu/papers/v11/cawley10a.html%5Cn, http://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf. Accessed 2019-02-01.

Chawla, N. V. (2005). Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook* (pp. 853–867). Springer.

Chen, R., & Lazer, M. (2013). Sentiment analysis of twitter feeds for the prediction of stock market movement. Stanford. Edu. Retrieved January, *25*, 2013.

Chen, H., & Zimbra, D. (2010). AI and opinion mining. *IEEE Intelligent Systems, 25*(3), 74–80.

Christopher, M., Payne, A., & Ballantyne, D. (1991). Relationship marketing: Bringing quality customer service and marketing together.

Cieliebak, M., Deriu, J., Egger, D., & Uzdilli, F. (2017). A twitter Corpus and benchmark resources for German sentiment analysis. *Fifth International Workshop on Natural Language Processing for Social Media*, (April), 45–51. https://doi.org/10.18653/v1/W17-1106

Correa, T., Hinsley, A. W., & de Zúñiga, H. G. (2010). Who interacts on the web?: The intersection of users' personality and social media use. *Computers in Human Behavior, 26*(2), 247–253. https://doi.org/10.1016/j.chb.2009.09.003 .

Cuthbertson, R., Furseth, P. I., & Ezell, S. J. (2015). *Innovating in a service-driven economy: Insights, application, and practice*. Palgrave Macmillan UK Retrieved from https://books.google.de/books?id=kXWkCgAAQBAJ. Accessed 2019-02-01.

Edvardsson, B., Kristensson, P., Magnusson, P., & Sundström, E. (2012). Customer integration within service development - a review of methods and an analysis of insitu and exsitu contributions. *Technovation, 32*, 419–429. https://doi.org/10.1016/j.technovation.2011.04.006 .

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining* (Vol. 96, pp. 226–231).

Finin, T., Murnane, W., Karandikar, A., Keller, N., Martineau, J., & Dredze, M. (2010). Annotating named entities in twitter data with crowdsourcing. Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's mechanical Turk.

Garimella, K., Weber, I., & De Choudhury, M. (2016). Quote rts on twitter: Usage of the new feature for political discourse. In Proceedings of the 8th ACM conference on web science (pp. 200–204).

Gerber, M. S. (2014). Predicting crime using twitter and kernel density estimation. *Decision Support Systems, 61*, 115–125.

Gollapudi, S. (2016). Practical machine learning. Packt Publishing. Retrieved from https://books.google.de/books?id=WmsdDAAAQBAJ. Accessed 2019-02-01.

Gregor, S., & Hevner, A. R. (2013). Positioning and presenting design science research for Maximim impact. *MIS Quarterly, 37*(2), 337–355. https://doi.org/10.2753/MIS0742-1222240302 .

Grinberg, M. (2018). *Flask web development: Developing web applications with python*. O'Reilly Media, Inc.

Haas, D., Wang, J., Wu, E., & Franklin, M. J. (2015). Clamshell: Speeding up crowds for low-latency data labeling. *Proceedings of the VLDB Endowment, 9*(4), 372–383.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating (ROC) Curvel characteristic. *Radiology, 143*(1), 29–36. https://doi.org/10.1148/radiology.143.1.7063747 .

Harding, J. A., Popplewell, K., Fung, R. Y. K., & Omar, A. R. (2001). Intelligent information framework relating customer requirements and product characteristics. *Computers in Industry, 44*(1), 51–65. https://doi.org/10.1016/S0166-3615(00)00074-9 .

Hauser, J. R., & Griffin, A. (1993). The voice of the customer. *Marketing Science, 12*, 1–27.

Hirt, R., Kühl, N., & Satzger, G. (2017). An end-to-end process model for supervised machine learning classification: from problem to deployment in information systems. In *Designing the Digital Transformation: DESRIST 2017 Research in Progress Proceedings of the 12th International Conference on Design Science Research in Information Systems and Technology. Karlsruhe, Germany. 30 May-1 Jun.*

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). A practical guide to support vector classification.

Hu, M., & Liu, B. (2004a). Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining (pp. 168–177).

Hu, M., & Liu, B. (2004b). Mining opinion features in customer reviews. 19th National Conference on Artifical intelligence, 755–760. https://doi.org/10.1145/1014052.1014073

Hu, M., & Liu, B. (2006). Opinion feature extraction using class sequential rules. In *Proceedings of the AAAI spring symposium: Computational approaches to analyzing weblogs* (pp. 61–66).

Hull, E., Jackson, K., & Dick, J. (2010). *Requirements engineering*. Springer Science & Business Media.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning. New York: Springer. Retrieved from https://link.springer.com/content/pdf/10.1007%2F978-1-4614-7138-7.pdf. Accessed 2019-02-01.

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science, 349*(6245), 255–260.

Jovic, A., Brkic, K., & Bogunovic, N. (2014). An overview of free software tools for general data mining. In *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on* (pp. 1112–1117).

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. In Proceedings of the 52nd annual meeting of the Association for Computational Linguistics *(Volume 1:* Long Papers*)*. https://doi.org/10.3115/v1/P14-1062

Kaplan, A. M., & Haenlein, M. (2010). Users of the world, unite! The challenges and opportunities of social media. *Business Horizons, 53*(1), 59–68. https://doi.org/10.1016/j.bushor.2009.09.003 .

Khalid, H. M., & Helander, M. G. (2006). Customer emotional needs in product design. *Concurrent Engineering, 14*(3), 197–206.

Kietzmann, J. H., Hermkens, K., McCarthy, I. P., & Silvestre, B. S. (2011). Social media? Get serious! Understanding the functional building blocks of social media. *Business Horizons, 54*(3), 241–251.

Kotler, P., & Armstrong, G. (2001). Principles of marketing. World wide web internet and web. *Information Systems, 42*, 105. https://doi.org/10.2307/1250103 .

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica, 31*, 249–268. https://doi.org/10.1115/1.1559160 .

Kuechler, W., & Vaishnavi, V. (2012). A framework for theory development in design science research: Multiple perspectives. *Journal of the Association for Information Systems.* https://doi.org/10.1201/b18448-6 .

Kuehl, N., Scheurenbrand, J., & Satzger, G. (2016). "*Needmining: Identifying micro blog data containing customer needs*". Research Papers. 185. https://aisel.aisnet.org/ecis2016_rp/185.

Kuhn, M., & Johnson, K. (2013). Applied predictive modeling. In *Applied predictive modeling*. https://doi.org/10.1007/978-1-4614-6849-3 .

Kühl, N., Goutier, M., Ensslen, A., & Jochem, P. (2018). Literature vs. Twitter: Empirical insights on customer needs in e-mobility. *Journal of Cleaner Production*. https://doi.org/10.1016/j.jclepro.2018.12.003.

Kvaløy, O., Nieken, P., & Schöttner, A. (2015). Hidden benefits of reward: A field experiment on motivation and monetary incentives. *European Economic Review, 76*, 188–199.

Lee, D., Jeong, O.-R., & Lee, S.-G. (2008). Opinion mining of customer feedback data on the web. Proceedings of the 2nd international conference on ubiquitous information management and communication ICUIMC 08, 230. https://doi.org/10.1145/1352793.1352842

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research, 18*(17), 1–5.

Limehouse, D. (1999). Know your customer. *Work Study, 48*(3), 100–102. https://doi.org/10.1108/00438029910262518 .

Ling, C. X., Huang, J., & Zhang, H. (2003). AUC: A statistically consistent and more discriminating measure than accuracy. In IJCAI international joint conference on artificial intelligence (pp. 519–524).

Loria, S. (2017). TextBlob.

March, S. T., & Smith, G. (1995). Design and natural science research on information technology. *Decision Support Systems, 15*(4), 251–266 Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=epref&AN=DSS.AE.BEA.MARCH.DNSRIT&site=ehost-live. Accessed 2019-02-01.

Marshall, T. C., Lefringhausen, K., & Ferenczi, N. (2015). The big five, self-esteem, and narcissism as predictors of the topics people write about in Facebook status updates. *Personality and Individual Differences, 85*, 35–40. https://doi.org/10.1016/j.paid.2015.04.039 .

Maynard, D., Bontcheva, K., & Rout, D. (2012). Challenges in developing opinion mining tools for social media. Proceedings of the@ NLP can u tag# Usergeneratedcontent, 15–22.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space, 1–12.

Mikolov, T., Yih, W., & Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In Proceedings of the annual conference of the north American chapter of the Association for Computational Linguistics: Human language technologies (HLT-NAACL) (Vol. 13, pp. 746–751).

Misopoulos, F., Mitic, M., Kapoulas, A., & Karapiperis, C. (2014). Uncovering customer service experiences with twitter: The case of airline industry. *Management Decision, 52*(4), 705–723.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). Foundations of machine learning. MIT Press.

Montgomery, D. C. (2013). *Design and Analysis of Experiments (Eighth Edi)*. Hoboken: Wiley.

Müller, V. C., & Bostrom, N. (2016). Future progress in artificial intelligence: A survey of expert opinion. In Fundamental issues of artificial intelligence (pp. 553–570). Springer.

Nair, S., Rao, N., Mishra, S., & Patil, A. (2017). India's charging infrastructure — Biggest single point impediment in EV adaptation in India. In 2017 IEEE transportation electrification conference (ITEC-India) (pp. 1–6). https://doi.org/10.1109/ITEC-India.2017.8333884

Oke, A. (2007). Innovation types and innovation management practices in service companies. *International Journal of Operations & Production Management, 27*(6), 564–587.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In Proceedings of the ACL-02 conference on empirical methods in natural language processing-*Volume* 10 (pp. 79–86).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in {P}ython. *Journal of Machine Learning Research, 12*, 2825–2830.

Peffers, K., Rothenberger, M., Tuunanen, T., & Vaezi, R. (2012). Design science research evaluation. *Design Science Research in Information Systems. Advances in Theory and Practice*, 398–410. https://doi.org/10.1007/978-3-642-29863-9_29 .

Perrin, A. (2015). Social Media Usage: 2005–2015: 65% of Adults Now Use Social Networking Sites–a Nearly Tenfold Jump in the Past Decade. *Pew Research Center*, (October), 2005–2015.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers, 10*(3), 61–74.

Porter, M. F. (2001). Snowball: A language for stemming algorithms.

Powers, D. M. (2011). Evaluation: From precision, recall and F-measure to ROC, Informedness, Markedness and correlation. *Journal of Machine Learning Technologies, 2*(1), 37–63.

Saldaña, J. (2015). The coding manual for qualitative researchers. The coding manual for qualitative researchers. https://doi.org/10.1017/CBO9781107415324.004

Scheffler, T., Gontrum, J., Wegel, M., & Wendler, S. (2015). Mapping German tweets to geographic regions. Working Paper - University of Potsdam.

Scheurenbrand, J., Engel, C., Peters, F., & Kühl, N. (2015). Holistically defining E-mobility: A modern approach to systematic literature reviews. *Karlsruhe Service Summit*, 17–27. https://doi.org/10.5445/KSP/1000045634 .

Schlagwein, D., Fischbach, K., & Schoder, D. (2011). Social information systems: Review, framework, and research agenda. *International Conference on Information Systems*. https://doi.org/10.12980/APJTB.4.2014C1020 .

Sheldon, K. M., Elliot, A. J., Kim, Y., & Kasser, T. (2001). What is satisfying about satisfying events? Testing 10 candidate psychological needs. *Journal of Personality and Social Psychology, 80*(2), 325–339.

Silva, C., & Ribeiro, B. (2003). The importance of stop word removal on recall values in text categorization. In *Neural networks, 2003. Proceedings of the international joint conference on* (Vol. 3, pp. 1661–1666).

Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull., 24*(4), 35–43.

Srivastava, A. N., & Sahami, M. (2009). *Text mining: Classification, clustering, and applications*. CRC Press.

Srivastava, R. K., Shervani, T. A., & Fahey, L. (1999). Marketing, business processes, and shareholder value: An organizationally embedded view of marketing activities and the discipline of marketing. *The Journal of Marketing*, 168–179.

St Louis, C., & Zorlu, G. (2012). Can twitter predict disease outbreaks. *BMJ, 344*, e2353.

Steinwart, I., & Christmann, A. (2008). Support vector machines. Springer Science & Business Media.

Stieglitz, S., Dang-Xuan, L., Bruns, A., & Neuberger, C. (2014). Social media analytics. *Business & Information Systems Engineering, 6*(2), 89–96.

Subramaniam, L. V., Faruquie, T. A., Ikbal, S., Godbole, S., & Mohania, M. K. (2009). Business intelligence from voice of customer. In IEEE international conference on data engineering (pp. 1391–1402).

Timoshenko, A., & Hauser, J. R. (2018). Identifying customer needs from user-generated content.

Tumasjan, A., Sprenger, T., Sandner, P., & Welpe, I. (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment. Proceedings of the fourth international AAAI conference on weblogs and social media, 178–185. https://doi.org/10.1074/jbc.M501708200 , 280

Turney, P. D. (2002). Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the 40th annual meeting on association for computational linguistics (pp. 417–424).

Twitter (2016). About Twitter. Retrieved August 2, 2016, from https://about.twitter.com/de/company/press/milestones, last accessed 2016-08-02.

Twitter (2018). Twitter Streaming API. Retrieved March 22, 2018, from https://developer.twitter.com/en/docs/tweets/filter-realtime/guides/connecting. Accessed 2019-02-01.

Varma, S., & Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics, 7*(1), 91.

Webb, G. I., Pazzani, M. J., & Billsus, D. (2001). Machine learning for user modeling. *User Modeling and User-Adapted Interaction, 11*(1), 19–29.