FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

# Crowd Data

## Ein datenanalytischer Ansatz zur Crowdfunding Projektanalyse

DIPLOMARBEIT

zur Erlangung des akademischen Grades

**Diplom-Ingenieur**

im Rahmen des Studiums

**Business Informatics**

eingereicht von

**Lukas Grömer BSc**
Matrikelnummer 0927316

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber

Wien, 17. Jänner 2016

| Lukas Grömer | Andreas Rauber |
|---|---|

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

# Crowd Data

## A data analysis approach to crowdfunding project analysis

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

**Diplom-Ingenieur**

in

**Business Informatics**

by

**Lukas Grömer BSc**
Registration Number 0927316

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber

Vienna, 17th January, 2016

Lukas Grömer

Andreas Rauber

# Erklärung zur Verfassung der Arbeit

Lukas Grömer
Lederergasse 22, 1080 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ort, Datum

Unterschrift Lukas Grömer

# Acknowledgements

I would like to express my appreciation and thanks to my advisor Professor Dr. Andreas Rauber, for supervising this thesis and giving me the opportunity to pursue research in two fascinating fields of my choice – crowdfunding and data mining.

Also, I want to thank David Heberling and David Holetzeck from the Table of Visons GmbH, who created one of the first crowdfunding platforms in Germany – pling.de – and helped me in developing the idea behind this thesis.

I also feel incredibly thankful for the endless support i received from my family, not only during the process of writing this thesis, but throughout my whole life in general. I want thank my father Herbert for guiding me on my way to become an engineer and my mother Margarete, who consistently made my life easier in so many different ways.

Last but not least, my girlfriend Sophie. I want to thank her for her tireless support and for being so incredibly patient and helpful, even during long nights of writing and work-dominated weekends.

# Kurzfassung

Ein großes Problem von Crowdfunding ist, dass mehr als die Hälfte aller Kampagnen scheitern. Dies hat zur Folge, dass einerseits Projektgründerinnen und Projektgründer ihre Ideen nicht in die Wirklichkeit umsetzen können und andererseits, dass Crowdfunding-Plattformen keine Provision bekommen. Das bedeutet einen Verlust für beide Seiten. Das Ziel dieser Arbeit ist es, Methoden vorzustellen, die Projektgründerinnen und Projektgründern sowie Plattformen helfen, die Erfolgsaussichten von Projekten zu verbessern. Die Analyse wurde mit Data-Mining Algorithmen durchgeführt und basiert auf über 14.000 Kickstarter-Projekten, gesammelt im Zeitraum zwischen Juni und Oktober 2015.

Zu Beginn wurden rein statische Attribute von Projekten untersucht. Das sind jene Attribute, die bereits vor dem Projektstart erschließbar sind, wie zum Beispiel das Projektziel, die Projektbeschreibung, Belohnungen und das soziale Netzwerk der Gründerin beziehungsweise des Gründers. Mit dieser Analyse konnten die Projekte bereits mit einer Genauigkeit von 78.5% als erfolgreiche oder gescheiterte Projekte identifiziert werden. Indem man die dynamischen, sich zeitlich verändernden Projektattribute in das Modell aufnimmt, konnte diese Genauigkeit auf einen Wert von 83.4% nach einem Prozent der Projektdauer gesteigert werden. Möglich macht das die rapide Projektentwicklung von erfolgreichen-Kickstarter Projekten. Die gefundenen Modelle wurden zum Abschluss noch in ein CrowdData Framework verpackt, welches Projektgründerinnen und Projektgründer in der Findung des optimalen Projektsetups unterstützt und sie den Projektverlauf überwachen lässt.

# Abstract

A major problem of crowdfunding is, that more than half of all campaigns fail. This is not only bad for founders who then cannot realize their ideas but also for crowdfunding platforms who draw commissions from successfully funded projects. The goal of this thesis is to give platform providers and founders data based methods for improving the success of their projects. The analysis was performed with data mining algorithms and is based on over 14,000 Kickstarter projects between June and October 2015.

Initially, static project attributes were analyzed, which are present even before a project has been launched. Such attributes concern the project‘s general setup, the description, rewards and the founders social media network. With this method the success or failure of a project could be predicted with an accuracy of 78.5%. By including some dynamically changing attributes the accuracy could even be boosted to a value of 83.4% after one percent of the campaigns lifetime. The rapid development of successful Kickstarter projects makes this possible. Finally, these models were incorporated into a novel CrowdData framework, which supports founders in finding the ideal project setup and allows them to monitor the success probability over time.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

CHAPTER 1

# Introduction

## 1.1 Motivation

The 2008 financial crisis hit the banking sector very hard and consequently it became noticeably difficult for entrepreneurs to receive funding from finance institutes in the traditional way [Kir+14], while simultaneously the technological advance of and possibilities gained through internet platforms and online payment paced forward. This resulted in a perfect breeding ground on which crowdfunding could develop and prosper. Over the past years crowdfunding has even evolved into a serious alternative to traditional financing. Massolution [Mas15] calculated the worldwide crowdfunding market for 2014 of $16.2 billion, experiencing a boost of 167% up from $6.1 billion in 2013. In 2015, they even expect a value of $34.4 billion, meaning another increase of the market by more than double within one year. Some projects raised millions of dollars alone, for example the Pebble E-Paper Watch with $10.3 million, the Ouya open source gaming console with $8.5 million, or Oculus Rift raising $2.3 million [Sch14]. This should underline the seriousness and growing importance of such financing approaches.

At the same time data storage continuously got cheaper and cheaper[1], obviously leading, in many cases, to a store-first-process-later attitude. Hence, huge collections of data piled up. With the new collection standards new requirements for analysis arose. Traditional statistics and the creation of representative samples as one of its major tasks, became more or less obsolete in such exercises, since suddenly the whole data population was present and ready to analyze. Under these circumstances, the big data hype emerged and data mining leaped forward.

While data analysis, data mining and big data prosper and boom in many industries, it is still in its infancy in the area of crowdfunding. Only few scientific papers covering

[1] More precisely the price of storage media, hard disks in particular, dropped by about 5 orders of magnitude (100,000:1) since the 1980s [Kov05]

this subject have been published until now, nonetheless the number of contributions to crowdfunding analysis and conducted project analysis are increasing. To sum it up, crowdfunding has got a huge market size and is causing high interest a variety of media. Additionally it incorporates a big number of stakeholders and users, who generate hundreds of thousands of data instances. Plus it has its field of activity in the internet, which results in perfectly accessible and easy-to-process resources. All this inevitably leads to data mining problems. Despite these facts, crowdfunding data analysis was unattended for a long time (and still is). The lack of research in this field makes it very interesting to conduct an early attempt at data analysis and this thesis aims to provide valuable information to crowdfunding platforms, project owners and backers.

## 1.2 Problem Statement

In order to get a clean understanding of the problems that crowdfunding platforms, project owners and backers face, we need to go one step back and examine the basics of crowdfunding. Howe [How06] observed a new trend in companies in the mid 2000s to let external people ("the crowd") take part on internal processes through internet platforms and created the term "crowdsourcing", and with it the foundation for crowdfunding. The major goal of crowdsourcing is to tap external resources. Crowdfunding, as a branch of crowdsourcing, gives entrepreneurs a platform to present themselves and their project in front of a community in order to acquire funding. The community or the specific supporters get rewards in return. Hence, crowdfunding is all about pooling financial resources through the internet. This argument is underlined by Belleflamme et al. [Bel+10], who define crowdfunding as "an open call, essentially through the Internet, for the provision of financial resources either in form of donation or in exchange for some form of reward and/or voting rights in order to support initiatives for specific purposes" and by De Buysere et al. [Buy+12], who see crowdfunding "as a collective effort of many individuals who network and pool their resources to support efforts initiated by other people or organizations" which is "usually done via or with the help of the Internet".

With this in mind, a typical setup for reward-based crowdfunding platforms, which this thesis aims to provide ultimately, looks as follows: Normally, project owners share their ideas and projects with an online community — a crowd — and define a certain monetary goal that must be fulfilled. The community pledges money for those projects and receives rewards or shares in return. With an all-or-nothing policy, a very common strategy on crowdfunding platforms, the funders only have to pay their pledged amount, if the goal was accomplished at the end of the funding period. Therefore project owners have the strong need to best present their project, in order to be successful and receive funding. Crowdfunding platforms like Kickstarter, Indiegogo, RocketHub, FundRazr, to name a few, charge, according to Forbes Online [Tay13], a fee of 4–9% of the gathered amount plus transaction fees to the project owner, of course highly dependent on the chosen funding plan. Thus it is obvious that the success of the platforms is directly proportional to the success of the presented projects, and so it's not only project owners who have

great interest in making their project successful. Consequently platform providers must take steps towards achieving better and more successful funding projects. According to TechCrunch [Eth13] at Kickstarter only 44% and at Indiegogo only 34% of all projects get funded successfully, which emphasizes the need and the potential for improvement for the projects and their success.

## 1.3 Research Questions and Goals

In an effort to help project owners receive funding in order to realize their ideas, and platform providers to focus their efforts for hosting more and more successful projects, this thesis tries to highlight success factors and driving variables found in projects. There are two classes of possible success drivers. The first class is static factors, which are success drivers that are setup once, and before the initial start of the project. These are assumed to be non-volatile throughout the project‘s lifetime. Examples of such are social network factors like Facebook friends, content factors like pictures and description text variables and also attributes describing the rewarding strategy. All of these are defined or predetermined before the project launches. Therefore the first set of research questions can be defined as such:

**RQ1**: What are static success factors for crowdfunding projects and which factors influence the project‘s success? To which extent can the campaign‘s success be predicted even before it has started?

Furthermore one has to keep in mind that projects evolve while they are online. Hence, it is not sufficient to only perform a static analysis of the project‘s success. Time dependent project variables that influence the potential success of a project are called dynamic factors and mark the second class of success drivers. In association with the dynamic project factors, the following questions arise:

**RQ2**: How do projects develop over time? What patterns within the project‘s development emerge? What could a project forecasting model look like?

At this moment, there is already some existing research examining crowdfunding with data mining technologies. Static success factor analysis and also the dynamics of projects are no completely new topics. Those papers present some interesting drivers for success, but have a shortcoming in translating the findings into concrete business applications. Based on these deficiencies a third complex of questions arises:

**RQ3**: What could a crowd data framework look like, in order to best support project owners and platform operators in their project analysis? How can one transform the findings into a useful support tool?

## 1.4 Methodology

The analysis will be conducted based on projects present on the crowdfunding platform Kickstarter. The database is wrapped by a website and an API, consequently all necessary data needs to be gathered and brought into analyzable form first. Defining the data structure in cube form is beneficial, due to its solid handling of dimensions and the possibility to combine and aggregate data. The development process for the collection application is inspired by design science with its phases analysis, design and evaluation. The analysis step‘s biggest task is to identify requirements for the data cube. Normally this first phase also includes a relevance analysis, which is trivial in this case, due to the obvious need of this application to fulfill the data analysis task. Alongside the definition of the data cube, the design phase also includes the generation of the module and class structure. Last but not least, the evaluation checks if the initial requirements of attributes and structures are fulfilled which in this case is an evaluation through extensive testing. However, the main focus of this thesis lies on the data mining task and the business applications of the results, the data collection algorithm is only a means to an end and is intentionally kept short.

The CRISP DM - Cross Industry Standard Process for Data Mining - model from Chapman et al. [Cha+00] serves as a basic model for data mining tasks. As its name obviously suggests, it is applicable for any data mining task independent from its field of origin. It breaks down the data mining task into the phases business understanding, data understanding, data preparation, modeling, evaluation and finally deployment. Without understanding the domain and its data, it is ultimately impossible to conduct any legitimate examination, therefore in the first phases a general understanding for the domain and its data need to be established. Data preparation accommodates the selection of table, attributes and records and constructs the final data set from the raw data, ready for analysis. Afterwards a model is created for the outlined data mining problem and eventually evaluated. The deployment of the evaluated data mining models will only happen to the extent of including them into the generic crowd data framework. Clearly, the business and data understanding part of the CRISP DM process and the design of the collection algorithm have mutual dependencies, therefore iterations of those processes are more than likely.

The actual mining and analysis methods for this thesis are wrapped by the CRSIP DM model and include mainly a range of supervised learning and classification algorithms to identify drivers and important attributes for completed historical projects. But, especially for the dynamic development, clustering algorithms help to get a grasp of evolving development patterns.

For the data mining tasks listed above a huge number of data mining tools are applicable. One precondition for the tool selection is its required open source origin. It is, however, not hard to find suitable, open-sourced tools, since a poll by Gregory Piatetsky [Pia14] of KDNuggets – a well recognized online news platform covering business analytics

and data mining – identified, that the majority of the top 10 popular data mining tools are in fact open source. This obviously implies the existence of big communities and therefore decent community support, and mostly prevents premature and buggy software. The actual poll results can be observed in figure 1.1, which depicts the top 10 most used data mining tools from 2014, compared with the ones from 2013. In this thesis the data mining tasks are performed mostly in Weka. Additionally Python, R and Excel are used, mostly for data preprocessing, pivoting, exploratory statistics and visualizations. All three of them appear in the data mining survey under the top ten.

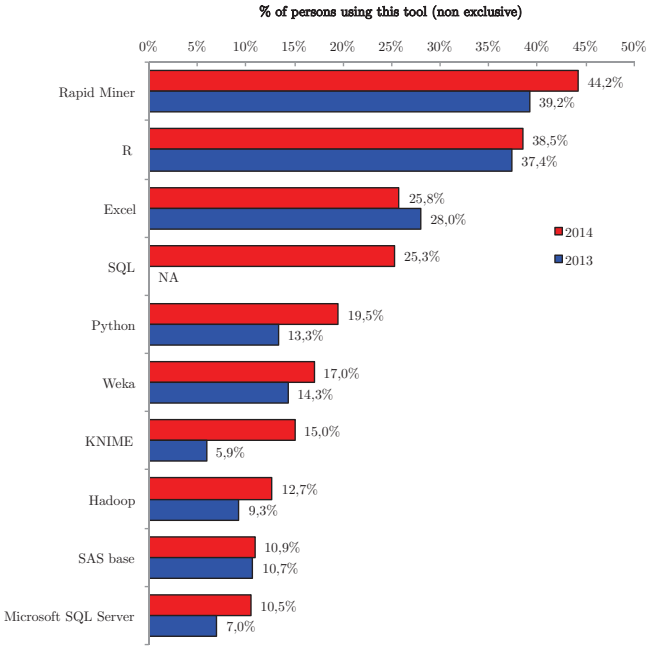



Figure 1.1: Data analysis tools in use [Pia14]

Once again, this thesis is an examination of real world phenomena with a strong focus on practical usage and commitment to implementation possibilities in business applications. Therefore the focus of the data mining tasks lies on finding beneficial insights for project owners and platform providers. Parameter experiments on data mining algorithms are inevitable, but parameter optimization in extreme is not the declared intention of this work. Additional boundaries relate to the availability of data. Clearly Kickstarter offers only a “small” amount of data for each project. Some very interesting and possibly important attributes concerning user statistics and user behavior, page views, etc. are unavailable to the public. The analysis is therefore restricted to open data.

CHAPTER 2

# Theoretical Background

This chapter introduces the theoretic foundations for this thesis and its related tasks. This includes the CRISP Data Mining Framework, outlining the task independent mining framework to which this thesis is aligned, but also relevant definitions and descriptions of basic data mining algorithms. Additionally, data preparation, attribute selection and other necessary preprocessing mechanisms will be discussed, in order to achieve a satisfactory overview on theoretic basics closely connected to the thesis. Finally, it attempts to give a brief overview of the current state of the art in the niche of crowdfunding analysis.

## 2.1 CRISP DM Framework

The widely-used and popular [Aze+08] Cross Industry Standard Procedure for Data Mining from SPSS [Cha+00] or simply CRISP DM Framework, serves as the meta-model and starting point for this thesis. It provides a clear and structured approach for handling industry independent, generic data mining tasks. The framework urges the user to follow the steps in generating general business understanding, establishing data understanding, data preparation, modeling, evaluation and deployment as depicted in figure 2.1. In the following section the framework will be briefly examined.

Basically, the framework is organized in four layers. On top, one can find a generic, application independent phases layer. The procedure outlined in figure 2.1 simply describes the sequence of those top level phases. Placed right beneath them are more specific, but still generic jobs in the "generic tasks" layer, creating a hierarchical structure. Following the hierarchy down, one passes the specialized tasks layer and reaches the process instances, as leafs of the hierarchy. Specialized tasks and process instances are created for each application differently. In the following section, the individual phases shall be discussed and it will be examined how the data mining problem can be aligned according to the framework.

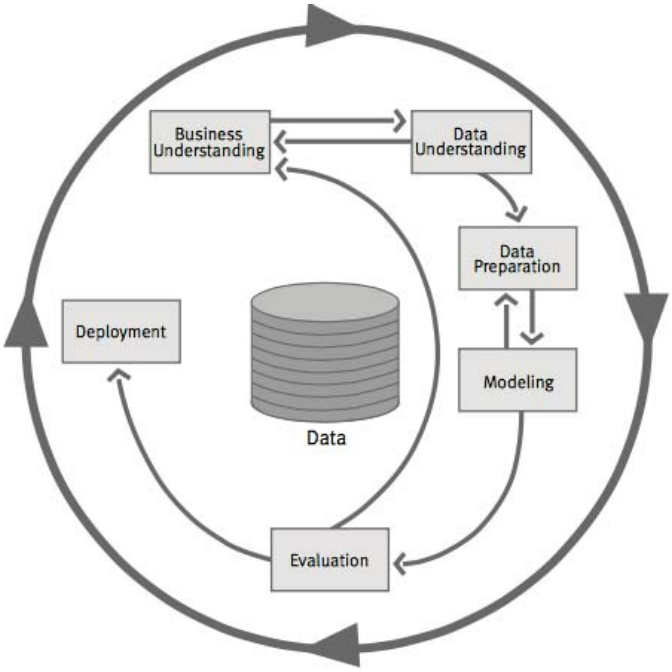



Figure 2.1: The CRISP DM 1.0 framework [Cha+00, p. 10]

**Business understanding**

In general, these tasks involve the determination of business objectives, but also the identification of boundary conditions concerning resources, assumptions and other factors, that have an influence on the data mining project. Additionally, risk and cost-benefit assessments are resided in this step. At the end of this phase, tools and possible data mining techniques shall be identified, that suit the data mining tasks best. [Cha+00, p. 30pp]

The perfect understanding of the domain and the clear definition of business objectives and success criteria is surely intuitive, but nonetheless essential for the data mining project. The parcel service UPS can serve as an example - they faced problems from the start with unexpected car failures and resulting delivery delays. Consequently, the idea was born to predict car failures of any type before they even occur, through analyzing of monitored car components data [May+13, p. 78]. However, without clean objectives behind the data mining task, the analyst could have predicted lifetime span of certain car types in almost the same manner – for example for fleet composition decisions – instead of the actual car failure prediction. This trivial example should emphasize the importance of this phase, which is to prevent the analyst from giving the right answer to the wrong question. With this method UPS could in fact save costs amounting to a sum of several millions of dollars.

The concrete “business understanding” step for our data mining problem will

mostly be covered in section 3.1 where the business objectives will be identified and occurring problems will be discussed. Since the project at hand is a rather small data mining project, not all listed generic tasks need to be mapped on concrete tasks. The focus will be put on the domain description.

**Data understanding**

As a first step however, before deep data understanding can be established, the data needs to be collected. Following the acquisition, a first assessment shall be conducted, including a volumetric and attribute analysis. Subsequently the data should be explored initially, leading – in the best case – to initial hypotheses and exploratory statistics. As a last step within this task the data quality needs to be judged, answering questions related to completeness, errors and missing values. [Cha+00, p. 37pp]

The term "data understanding" in the context of the CRISP-DM Framework refers to the establishment of crucial knowledge about the basic structure and the nature of data. It does not necessarily need to produce perfect insights yet, but rather collect and describe meta information and quality of data, from the predefined resources within the business understanding phase. The data collection and exploratory statistics, both part of this phase, are discussed in sections 3.2 and 3.3.

**Data preparation**

The data preparation phase produces analyzable and ready to model datasets. It contains the generic tasks "data selection", "data cleansing", "data construction", "data integration" and "data formatting". The selection step is all about deciding which data is intended to be used for analysis. Selection criteria include relevance in regards to data mining goals, quality and technical constraints. Following the data selection process, a cleaning step may be required. This could encompass the selection of a clean data subset or the insertion of suitable defaults or estimations. In other words the data cleansing is required to deal with all kinds of noise. Data construction is about deriving completely new attributes or transforming existing ones. The data integration task is responsible for combining data from multiple sources, and merging data instances. Finally, formatting data, which means applying syntactic modifications to the data, can be mandatory for certain mining tools. [Cha+00, p. 42pp]

**Modeling**

Chapman et al. suggest that the modeling phase in the CRISP data mining framework include the selection of the modeling technique, generation of test design, model building and model assessment. In the first step the actual initial modeling technique will be selected, but one has to keep in mind, that not all techniques are applicable to every task. Therefore one has to make sure, that assumptions about the data, which are presumed within a given technique, do in fact hold up. The generic task "generation of test design" then urges the user to define methods to test the model's quality and validity. A very common validation method

for classification problems is to use error rates. After having chosen a modeling technique and having defined validation criteria, the modeling tool is run on the prepared datasets to create one or more models. In a last step, the model is assessed according to the test and success criteria. [Cha+00, p. 48pp]

Clearly the data preparation tasks are tightly connected to the modeling phase, which is why they are discussed here together. Especially merging tables, the creation of derived attributes and ranging attributes seem very model specific, thus the loopback connection between the phases modeling and data preparation is more than justified. This feedback loop is also supported by the definition of Everitt [Eve98], who describes data mining as "the process of considering a large number of models including many which are 'data-driven' in order to obtain a good fit". The large number of models mentioned is not only produced by the application of many possibly suitable learners to the data, but also trying differently prepared data sets. In general, both phases are very straight forward and do not leave much room for misunderstandings. For the crowdfunding data analysis the concrete specialized tasks and process instances will be discussed in chapter 4.

**Evaluation**

While previous evaluations dealt with accuracy and generality, the evaluation phase rather concentrates on the examination of the degree of fulfillment of the business objectives, as well as the description and interpretation of findings. Of course if model deficiencies are discovered – for whatever reason – this is the place to discuss these matters. For quality assurance reasons the process needs to be reviewed in this phase, while the following steps also need to be considered. Possible analysis follow-ups could be: another next iteration in the CRISP DM lifecycle or setting up a new data mining task. [Cha+00, p. 51]

The whole CRISP data mining framework is obviously an adjusted "Plan-Do-Check-Act" cycle. In this context, the evaluation phase contains the check and act activities of the process. Check activities mean primarily the assessment of business objective fulfillment. Based on the deviation within the planned results, reactions towards a better performance are then defined. The modeling and evaluation phase of the CRISP DM within the crowd data analysis data mining task, specific to this thesis, can be examined in chapter 4.

**Deployment**

The deployment planning – as first generic task within the deployment phase – aims to apply the results into the business and should result in a deployment plan, which summarizes the strategy and necessary steps to "launch" the model. Additionally, the CRISP DM framework suggests for this phase, to plan maintenance and monitoring activities. These activities shall guarantee the correct usage of the data mining results and record the performance of the model over time. At the end of the project, a final report shall be created, containing a process description, findings, project costs, plan deviations and future works. As the final task of every

project an overall assessment needs to be given, containing information on what went wrong, what went right and possible improvements. [Cha+00, p. 52pp]

The project findings of the crowdfunding data mining tasks at hand, shall not be implemented and deployed in a business environment entirely. Nevertheless, the results are viewed in a practical business context and chapter 5 attempts to sketch possible ways to support project owners, their projects and crowdfunding platforms.

Ashby, one of the pioneers of cybernetics points out that "variety can destroy variety" [Ash56]. Since variety is a measure of complexity, his statement leads to the conclusion, that a complex matter can only be handled by other complex systems. Data complexity arises, inter alia, from the different kinds of data, the diversity and distribution of resources and the data dimensionality [Ras+07]. On the other hand data mining and knowledge discovery complexity in general "stems from a variety of tasks that can be performed to analyze the data and from the existence of several alternative ways to perform each task" [Cor+13]. The CRISP DM framework is organized in a cyclic manner, with several feedback loops and therefore proves its ability for handling complex data and complex data mining tasks simultaneously. Approaching complex data is done via the feedback loop from modeling to data preparation and complex data mining tasks are managed with the general feedback loop after evaluation.

## 2.2 Data Mining

Data mining has its core in the derivation of knowledge and creating an understandable output from a set of data. This can be observed by taking a look at different definitions. Chakrabarti et al. [Cha+06] define data mining as the science of extracting useful knowledge and information from huge data repositories. Hand et al. [Han+01] suggest that "data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner." Because of the requirement for processing big datasets a focus on the algorithm is obviously unavoidably. This is one major distinction to traditional statistical methods, which have their foundation in pure mathematics, resulting in the processing of large data heaps and data mining problems being too slow, as it is simply not intended [Han99]. This should not create the impression that data mining and traditional statistics have nothing in common. The opposite is true, but the focus is simply another.

### 2.2.1 Preprocessing

This section provides a brief theoretic overview on data preprocessing, with the main points being removal of outliners, handling of missing values and dataset imbalances. Together, these subtasks can be categorized as data cleansing. Normally the preprocessing additionally contains data selection and data construction tasks, but within the scope of this thesis they play a minor role and thus can be omitted. Data selection can be crucial

for the analysis of huge feature sets containing thousands upon thousands of attributes. Text classification with words of entire dictionaries as features is only one example for such an enormous amount of variables. This can be deadly when the learner‘s time complexity is an exponential function of the number of attributes[1]. In our case, however, we only examine a maximum of 30 – 40 features, hence a feature selection is not that crucial here, because – as can be seen in chapter 4 – the used learning algorithms can cope with such a “small” number of attributes very well.

All in all, missing value handling, outlier detection and the management of data set imbalances merit examination in dedicated, separate theses. Nevertheless, for the crowdfunding data analysis task at hand it is important to understand the implications of these concepts in order to find suitable machine learning algorithms. One example of such a robust learning algorithm are random forests.

### Outlier detection

An outlier is a data point “that appears to deviate markedly from other members of the sample in which it occurs” [Gru69]. An often used definition for outliers comes from Hawkins [Haw80], taking a similar line and stating an outlier as “an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism”. Aggarwal [Agg15] sums up other definitions describing outliers in data mining and statistics literature as abnormalities, discordants, deviants, or anomalies. One thing is clear, values that seem to differ from most of the values or produce a long tail within an assumed normal distribution need to be examined closely. Furthermore in most cases a multivariate outlier detection can make sense. For example, when examining the input of height and weight in a survey, the input of 190 cm and a weight of 50kg both are within their attribute distributions. However, the two values in combination for one person most likely mark an outlier.

The possible reasons for outliers are either measuring and data errors (whether produced by intentional misinformation or by accident), sampling errors or simply wrong assumptions about normality[2]. In this context normality does not mean normally distributed, but what the analyst assumed to be normal behavior and normal outcomes. If no measuring error is present and the analyst is confident of the outliers‘ natural source, their removal is hard to argue, because they inevitably belong to the population. Osborne and Overbay [Osb+04] sum up that, although the removal of extreme scores can lead to undesirable outcomes, in most cases the removal leads to beneficial results.

For outlier handling the thesis mainly uses a density-based method, namely the calculation of local outlier factors (LOF). These method assigns an outlier factor to each data instance based on the density of neighboring instances. Intuitively, LOF compares

[1] For example the time complexity for Top-Down Induction of Decision Tree (TDIDT) learners like ID3 and C4.5 is according to [Mar+96] in $O(A^2N)$ with $A$ the number of attributes and $N$ the number of instances

[2] An exhaustive list of reasons for outliers can be found in the e-paper article of Osborne and Overbay [Osb+04]. The above presented categorization is more or less a condensed version of their outlined enumeration.

the reachability of one node to the reachability of the k-nearest neighbor nodes. If, however, the neighbor nodes are better reachable than the specific node itself, it can be considered as an outlier. Formally LOF is defined as follows [Bre+00]:

$$\text{Sampleset} \quad S \tag{2.1}$$

$$\text{Data vectors} \quad p, o \in S \tag{2.2}$$

$$\text{Distance metric (e.g. Euclidean)} \quad d(p, o) \tag{2.3}$$

$$\text{Distance of the k-nearest neighbor of p} \quad \text{k-dist}(p) \tag{2.4}$$

$$\text{Neighbors within k-distance} \quad \text{N}_\text{k}(p) \tag{2.5}$$

$$\text{reach-dist}_k(p, o) = \max\{\text{k-dist}(o), d(p, o)\} \tag{2.6}$$

$$\text{lrd}_k(p) = 1/\left(\frac{\sum_{o \in N_k(p)} \text{reach-dist}_k(p, o)}{|N_k(p)|}\right) \tag{2.7}$$

$$\text{LOF}_k(p) = \frac{\sum_{o \in N_k(p)} \frac{\text{lrd}_k(o)}{\text{lrd}_k(p)}}{|N_k(p)|} \tag{2.8}$$

Additionally Breunig et al. [Bre+00] suggest a LOF heuristic for outlier identification where LOF is not only generated for one $k$, but a range of $k$ with $MinPts \leq k \leq MaxPts$. The maximum factor for each instance then determines the LOF factor.

Finally, Aggarwal describes "An attempt to use the wrong model for a given data set is likely to provide poor results. Therefore, the core principle of discovering outliers is based on assumptions about the structure of the normal patterns in a given data set. Clearly, the choice of the 'normal' model depends highly upon the analyst's understanding of the natural data patterns in that particular domain." [Agg15] This can be seen as the moral of outlier detection. It is essential that the analyst has the right model assumptions in order to be able to identify reasonable outliers.

### Missing value handling

Missing values are unavoidable in practice. According to Schafer and Graham [Sch+02] missing values can be categorized in "missing completely at random" (MCAR), "missing at random" (MAR) and "missing not at random" (MNAR). The first type MCAR refers to missing values, which are dependent on unobservable attributes only. MAR "missingness" is more general and indicates, that the missing variables can be dependent on other observed variables. The third type, MNAR, additionally allows a "self-dependence", meaning that the "missingness" itself can have information about the missing value. A good example for MNAR type is the observation of earnings, where persons with high

income are systematically hiding information. The type of "missingness" is important to mention, because there are big implications on the treatment strategies concerning biases and predictability.

Gelman et al. [Gel+06] categorize missing value strategies in methods discarding data and imputation methods. Case deletion and available-case analysis belong to the first category. It is self-evident however, that omitting instances or whole attributes could lead to information loss or biases if the type is not MCAR [Acu+04]. The other general approach to missing data management is to fill in or impute values. The range extends from very simple approaches, where the analyst inserts predefined values, means or median; to more complex models like regression and multiple imputation. It should be mentioned, that these methods are mostly designed for MAR assumptions only. Besides the missing value cleansing discussed in the preparation step, it is worth noting, that some learning algorithms are able to handle missing data themselves. C4.5 for example splits attributes with missing values in fractions, which are then used for case subsets [Grz+10].

#### Dataset imbalance

A problem often faced in data mining is dataset imbalance. It occurs when class labels are not distributed according to their relevance and importance. More precisely, the class of interest is underrepresented within the dataset. The following trivial example shall highlight some of the problems with imbalanced data sets: Let us assume we want to identify spam emails. We have a dataset containing 100 labels for emails with 10 "ham" and 90 "spam", no more information. Consequently a weak learner would produce a classifier, which simply takes the probabilities into account and would classify a newly incoming mail with a probability of 90% as "spam". Now, 10 "ham" emails are received, but the classifier obviously discards 9 – which is pretty bad. We need another classifier. Thus the number of "spam" emails is cut down to 10, while the 10 "ham" mails are kept unchanged. Consequently, the classifier will adapt and identifies 50% as "spam" and suddenly accepts 5 out of 10 "ham" emails that are coming into the mailbox. The tradeoff is, that the mail program now accepts 50% from 90 incoming "spam" mails as well. The data analyst needs to be aware of this tradeoff, but sampling techniques provide a solid way to raise the importance of underrepresented classes.

Having a look at performance metrics of a classifier like above, accuracy would be at a level of 91%. This might seem not bad, but obviously the classifier performs pretty poorly. Therefore it is self-evident, that predictive accuracy might not be appropriate when the data is highly imbalanced. This should underline, that the accuracy is not the best performance metric for unbalanced datasets and must be used with care. This argumentation is of the same tenor as He [He+09] who suggests the "Receiver Operating Characteristics" as good measure to overcome distribution-dependent metrics, like accuracy to a high extent, but also the F-measure. The suspicion, that our model above is not very good is stressed by the area under the ROC curve discussed in section 7, which

is 0 in both cases, which matches the minimum value on the scale.

Above we already examined one strategy for overcoming dataset imbalances. In literature the observed mechanism of cutting down the less relevant class is referred to as "undersampling". There are three general strategies to overcome those deficiencies: Undersampling, oversampling and weighting or cost-aware learners [He+09]. Undersampling refers to the method of reduction of the class(es) with more values. This may be very intuitive, but there is the obvious problem that reduction can lead to loss of potentially important data points. Additionally, the highly cost-intensive data collection was, in part, a waste. The other option is oversampling, which means the generation of additional data points based on the existing ones. It can be seen as the exact opposite of undersampling. This can be done simply throughout copying data points of the underrepresented class, or by using more sophisticated algorithms like SMOTE (Synthetic Minority Oversampling Technique), which introduces synthetic examples by joining k nearest neighbors of the minority class [Cha+02]. Obviously these techniques aim at penalizing mistakes in classifications of the minority class more than those from the majority. A third technique is the usage of cost-sensitive methods. There are three major approaches to this: introducing weights to the dataset, cost-minimizing techniques as ensemble methods to standard classifiers and the direct incorporation of cost-sensitive functions or features into classification paradigms [He+09].

### 2.2.2 Supervised Learning

Supervised learning can be defined as "methods that attempt to discover the relationship between input attributes and a target attribute." [Mai+10] More formally a supervised learning algorithm seeks a function $f : X \to Y$ over an example set $S = ((x_1, y_1), \ldots, (x_m, y_m)) \in (X \times Y)^m$, with $x_1, \ldots, x_m$ drawn i.i.d according to a fixed but unknown distribution $D$ and $X$ representing the input space, whereas $Y$ represents the output space respectively [Moh+12]. Classification and supervised learning find application in diverse scenarios, like spam detection [Awa+11], customer relations management [Hua11; Bal+12], credit ratings [Kha+10] and many more.

Essentially, supervised learning algorithms can be categorized into classification and regression methods. The difference between those two lies in the output space. Where in classification $Y$ is a set of predefined, known labels, in the field of regression the output space $Y$ is $\mathbb{R}$ and the task is to find predictions close to the correct ones. Clearly, finding the exact values as in classification is in fact utopistic. Representatives of both methods will be discussed briefly in the following section.

#### Naive Bayes

One of the simplest methods for supervised learning, which is yet very powerful, is the Bayesian classification. Before diving into the Bayes' classifier directly, the basics and the statistical foundation need to be discussed. First of all, the conditional probabilities

of events A and B are defined as follows [Gup06]:

$$P(A|B) = P(A \cap B)/P(B) \tag{2.9}$$
$$P(B|A) = P(A \cap B)/P(A) \tag{2.10}$$

By dividing the first equation by the second we can then derive the Bayes' theorem.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.11}$$

For a classification problem, the probabilities of an attributes vector belonging to a certain class is of interest. The class with the highest probability shall be selected. To put it more formally, one of the classes from the following candidate set $S = \{c|c \in C \wedge \forall k \in C \setminus c : P(c|X) >= P(k|X)\}$ must be selected randomly. $C$ represents the set of different classes, to which the data instances need to be assigned. The calculation of $P(c|X)$ is ad hoc impossible, but with the Bayes theorem workaround of $P(c|X) = P(X|c) \cdot P(c)/P(X)$ it is sufficient to simply calculate $P(X|c)$, $P(c)$ and $P(X)$. For the attributes of the $X$ vector the algorithm needs to assume independence, otherwise the decomposition of the probability $P(X|c) = P(x_1|c) \cdot \cdots \cdot P(x_n|c)$ with the feature vector $X = (x_1, \ldots, x_n)$ would be invalid. Although the independence assumption is quite restrictive and unrealistic, the algorithm produces astonishingly good results in practice [Ris+01].

### Support Vector Machines

Support vector machines try to find a hyperplane that best separates the data instances by their class. In other words the hyperplane linearly separates the classes from each other. By definition a general hyperplane $\mathcal{H} = \{x|\langle w, x\rangle + b = 0\}$ with $w \in \mathbb{R}^N$ and $b \in \mathbb{R}$. This is fit into the feature space in a way, that the margin between the instances of both classes becomes maximal. Figure 2.2 shows a classification problem with two attributes $a_1$ and $a_2$, where a hyperplane separates the two classes "balls" and "diamonds". In order to find suitable values for $w$ and $b$ the following optimization problem needs to be solved [Den+12]:

$$\max \frac{2}{||w||} \tag{2.12}$$
$$\text{with } y_i\langle w, x\rangle + b \geq 1 \tag{2.13}$$

The objective function represents the margin between the two classes, that needs to be maximized and the constraint origins in the hyperplane definition with slightly re-scaled $w$ and $b$. Since SVM plays only a little role in this thesis, a more detailed examination and as well as the the mathematical solution to this problem are intentionally omitted.

Abe [Abe10] states in order "[...] to enhance linear separability, the original input space is mapped into a high-dimensional dot-product space called the feature space" Such a separability problem is illustrated in figure 2.3. The left picture shows the initial feature

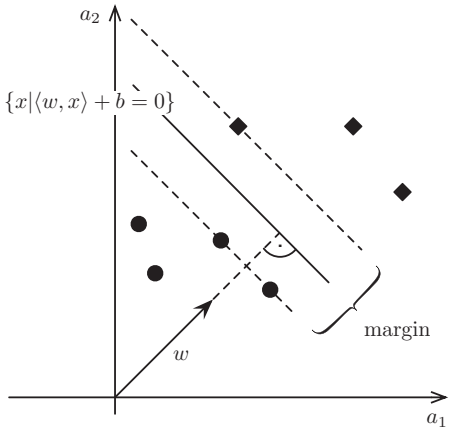



Figure 2.2: A 2-dimensional classification toy problem[Hea+98]

space, where it is obviously impossible to draw a hyperplane (in two dimensional space a line) that reasonably separates the classes. The mapping to the high dimensional feature space on the right side is, however, able to compensate this handicap. More formally the mapping can be described as $\Phi : \mathbb{R}^n \to \mathcal{H}$ and with $\mathcal{H}$ as the fictional features space and $n$ the number of attributes. A great advantage of this approach is, that it is not necessary to bring the whole feature space to the higher dimension. Instead, it is sufficient to introduce a function $\mathcal{K}(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ which is capable to calculate the dot product for the higher dimension. These functions are called kernels and are "located" in the original attribute space, but still act like scalar products in the other. Or as Abe [Abe10] states "advantage of using kernels is that we need not treat the high-dimensional feature space explicitly"

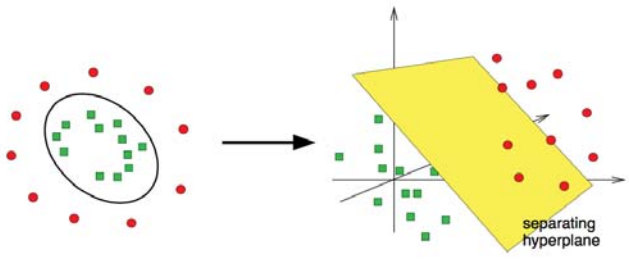
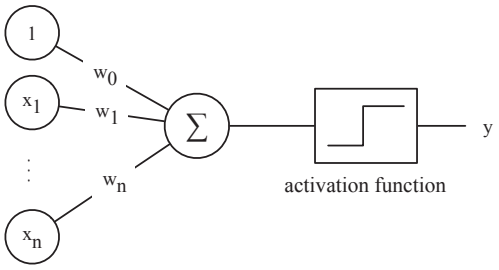



Figure 2.3: Fitting a high dimensional, fictive, separating hyperplane into a point cloud [Mar03]

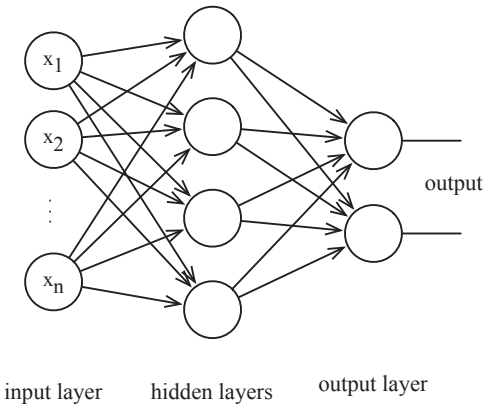
### Neural Networks

One of the early works in the field of artificial neural networks (ANN) was contributed in 1943 by Warren McCulloch and Walter Pitts, studying mathematical properties of

artificial neural networks [Roj13]. In their publications they introduced neurons which are able to produce a binary output based on some binary inputs, toggling "1" on the output if an input threshold is exceeded. These neurons are called McCulloch–Pitts units and can be used to produce simple logic gates. A more general computational model compared to McCulloch–Pitts units is the perceptron, extending them by a simple but yet crucial improvement of numerical weights and a back-propagation mechanism [Roj13]. The structure of one perceptron is illustrated in figure 2.4a. Multiple inputs and a bias, which is simply a reformulation for the threshold as input, are fed to the perceptron. They are weighted, summed up and connected via the activation function to the output. The activation function is responsible for producing a binary output. It is obvious that this type of weighted summation can only achieve linear separability can be achieved. By connecting several perceptrons in series, non-linear classifiers can also be generated. This mechanism is depicted in 2.4b.




(a) The structure of a single perceptron




(b) A multilayer perceptron

Figure 2.4: An overview to perceptrons

Until now, we have only discussed the general structure of perceptrons, no word was lost about how they can be trained. The magic word here is back-propagation. Figuratively speaking, one must tell the weights they need to change, if the overall output is not suitable. More formally, the weights are adapted in a way, such that the

overall cost function of the output level decreases in the opposite direction of the gradient towards a (hopefully global) minimum. Since only the desired value at the output level is given and no intermediate results, the weights of any neuron within the network depend on the intermediary neurons between itself and the output. The calculation for deep structures could possibly mean a computational hazard, and thus a trick can be applied. By choosing the activation function smartly, the current weights are only dependent on neighboring neurons and already calculated values.

### Regression

Linear regression serves as an important method in the field of econometrics, but still fits nicely in the context of classification and data mining. Its core task is, as in other supervised learning methods, to discover the relationship between input attributes and a target attribute. Figure 2.5 illustrates the idea of regression, which is generating a hyperplane that best describes the relationship between input and output values. The standard linear regression model describes the linear relationship from $x_1, \ldots, x_k$ inputs with some coefficients $\beta_0, \ldots, \beta_k$ with an error term $\epsilon$.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \epsilon$$

Now an OLS (ordinary least squares) estimation for the coefficients is applied which minimizes the error term $\epsilon$ for the observed $y$ and $x_i$. This regression model works only under several assumptions, like the independence of input attributes, the error homoscedasticity, or the absence of autocorrelation for the errors [Guj12]. Clearly it is also important that the training set contains at least $k + 1$ instances, otherwise the arising equation system cannot be solved.

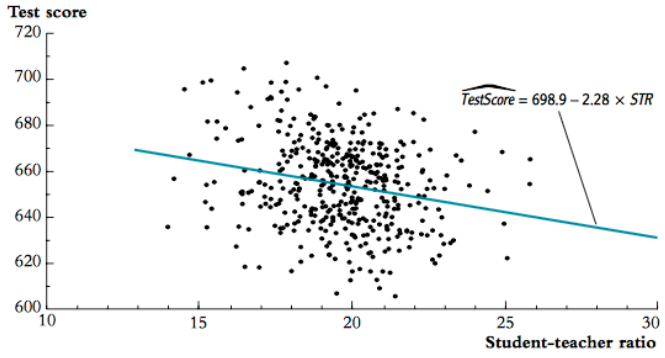



Figure 2.5: Example regression for test score and teacher-student ratio [Sto+11]

Classification aims on the prediction of certain classes. However, the discovery of exact values in linear regression is pretty unrealistic. Instead the algorithm rather concentrates on finding reasonable results close to the actual value to be and needless to say, regression primarily aims on prediction of non-discrete values. This shortcoming for matters of classification can be handled with logistic (logit) regressions, where for each class a

separate linear model is produced, setting $y$ to 1 if the instance belongs to the class, 0 otherwise [Wit+11]. The logit regression model is defined as follows [Hai+10]:

$$log\ \frac{p(x)}{1-p(x)} = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k$$

It is then solved for $p$, for receiving the dependent part with its range between 0 and 1. When classifying a new instance, the data instance at hand needs to be run through all models and – simply enough – the class with the largest probability value is selected.

### Decision Trees

When thinking of tackling complex issues, in our case complex data sets and complex data mining tasks, it is only logical to explore divide-and-conquer approaches. In the field of data mining such methods directly lead to decision trees, as [Wit+11] suggests with the statement "A 'divide-and-conquer' approach to the problem of learning from a set of independent instances leads naturally to a style of representation called a decision tree". An example of a decision tree is depicted in figure 2.6. The actual classification of unknown samples happens by "feeding" data instances to the decision tree and traversing through the tree nodes, beginning at the root node. At each node an instance is matched against predicates, who decide on which path to continue, until a leaf node is found. Finally the leaf node indicates either the predicted class label for the instance, or class probabilities instead.

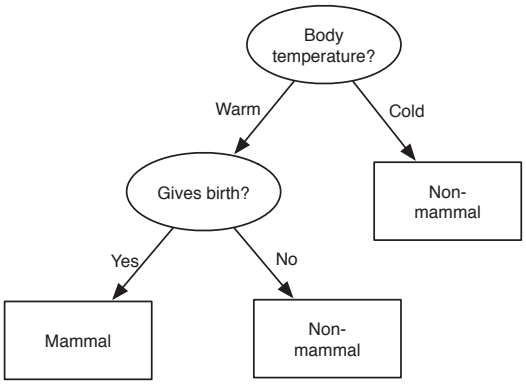



Figure 2.6: Example for a decision tree classifier based on [Tan+05, p. 151]

The outlined induction algorithm 2.1 presents a generic method for creating decision trees based on [Kot07]. First it evaluates possible stop criteria. This could be a reached maximum tree depth or simply finding a finite class label. Afterwards, the information gained – or another ranking criteria – is calculated for each attribute in the feature vector. This is the basis for the split decision. To put it differently, it selects the attribute, which leads to a viable result in the fastest way possible. For the best attribute a new decision node is generated. The sample set is then partitioned according to the split criteria and

further sub decision trees are generated recursively on those sets.

**Algorithm 2.1:** A generic decision tree algorithm [Kot07]

```
1 Check for base cases;
2 forall the attribute a do
3 |   Find the feature that best divides the training data such as information gain
  |   from splitting on a
4 end
5 Let a_best be the attribute with the highest normalized information gain;
6 Create a decision node node that splits on a_best;
7 Recurse on the sub-lists obtained by splitting on a_best and add those nodes as
  children of node;
```

The methods used for splitting a node or an attribute highly depends on the attribute types. In the following section, the different splitting methods shall be discussed briefly according to [Tan+05].

**Binary attributes**
Attributes that can only take two different values – binary attributes – result in the simplest splitting method and obviously in two sub trees, one for each attribute class.

**Nominal attributes**
These attributes can either result in a multiway-split or by grouping the values that $2^{k-1} - 1$ one-against-all possible binary splits arise. This is highly dependent on the algorithm used.

**Ordinal attributes**
Ordinal attributes receive the same treatment as nominal attributes, but requiring additional effort to be put into grouping values, since it makes no sense when grouping attributes $A = \{a_1, a_2, a_3, a_4\}$ with ordering $a_1 < a_2 < a_3 < a_4$ in the split groups $S_1 = \{a_1, a_3\}$ $S_2 = \{a_2, a_4\}$.

**Continuous attributes**
For continuous attributes there are again the possibilities of a binary split resulting in two classes $(-\infty, v)$ and $[v, \infty)$; or a multi-way split which produces a range of intervals with $k$ split values and the following classes $\{(-\infty, v_1), [v_i, v_{i+1}), [v_k, \infty)\}$ for $i = 1, \ldots, k-1$ accordingly.

The C4.5 algorithm and its predecessor ID3 can be considered as the standard methods for decision tree induction. Both use the (normalized) information gain as main split criteria. Weka for example uses a open source java implementation of the C4.5 algorithm — called J48.

One problem of classification trees is that a derived model may be highly reliant and perfectly tailored to the given sample set, but not to the reality. To put this in more formal terms, the induced decision tree might suffer from a lack of generalization. This refers to the term "overfitting". Gupta states [Gup06] that the simpler model is more likely to be the better one and therefore the pruning technique can be applied, which makes over-fitted trees simpler.

Safiavian et al. [Saf+91] outlined inter alia the following advantages and disadvantages of decision tree induction: Due to the divide-and-conquer principle of the decision tree classifiers, complex global decisions can be approximated by multiple smaller local decisions at various levels of the tree. This comes handy in high-dimensional spaces. Consequently, the tree classifier induction algorithm is tested only against certain subsets of the sample set and therefore has improved efficiency, due to the elimination of unnecessary computations. The drawbacks are however, that errors may accumulate from level to level in a large tree and that the performance of a decision tree classifier strongly depends on how well the tree is designed.

### Random Forests

Actually, random forests – introduced by Leo Breiman – do not mark a new classification technique but rather form an ensemble technique. They practically generate a number of decision trees – a forest – with a high amount of randomization. For one bagging is applied in a way, such that a random subsample of the data is drawn with replacement for each tree to be generated. Additionally the learning algorithm performs the node split on a random feature subset. The random forests induction then result in a probabilistic output and not just a single class point prediction but an entire class distribution [Cri+11]. This probabilistic model can be described as follows:

$$p(c|v) = \frac{1}{T} \sum_{t}^{T} p_t(c|v) \tag{2.14}$$

The probability of any tested data instance $v$ belonging to a specific class $c$ with $T$ random trees, is responsible for the final class selection.

Figure 2.7 depicts the leaf selection within every tree that produces a conditional probability for class $c$ and an input vector $v$ which is then averaged over all trees as we have seen above. According to Breiman [Bre01] the advantages of random forests are an equally good or sometimes better accuracy compared to boosting, the robustness to outliers and noise and runtime performance improvements to bagging and boosting, to name just the most important ones.

### Evaluation of classifiers

After a model is generated, it needs to be evaluated. Clearly a separation of training and testing instances need to be done, because otherwise – when using the training set for

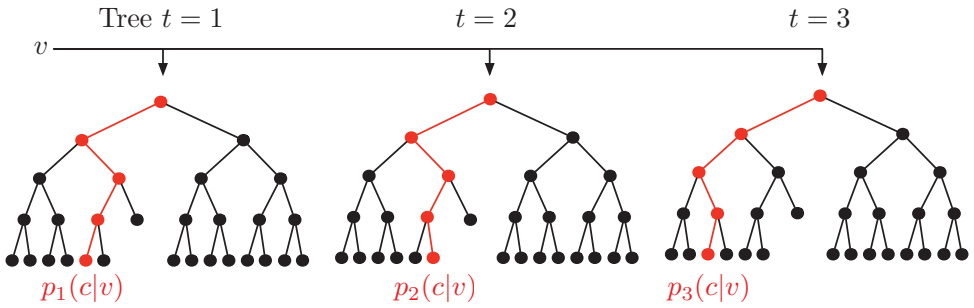



Figure 2.7: The random forest leaf selection [Cri+11, p. 29]

| | actual positive | actual negative |
|---|---|---|
| predicted positives | TP | FP |
| predicted negatives | FN | TN |

Table 2.1: A confusion matrix for binary classification [Dav+06]

evaluation as well – one would risk a high bias. To derive such an unbiased estimate of a model performance Gupta [Gup06] describes the following common evaluation methods: holdout method, random sub-sampling, k-fold crossvalidation, leave-out-one method and bootstrapping. K-fold cross validation, for example, splits the dataset in $k$ parts. Each of the $k$ parts acts as a test set once, while the other $k - 1$ sets are used for training independent models. The performance measures are then averaged. In the section to follow, some important and widely-used measures for model adequacy and competence are presented. The nomenclature of the following examples refer to the confusion matrix in figure 2.1.

**Accuracy/Classification error**

Intuitively one can compare all correctly classified instances from the output estimation $\hat{y}$ to the actual class $y$. This metric is called accuracy and using the confusion matrix nomenclature from figure **??** we can derive following formula:

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

**Precision and Recall**

Precision is defined as the fraction of retrieved objects that are relevant [Han+01]. Figuratively speaking, how many of the predicted positives are actually positive. Recall on the other hand is defined as the proportion of relevant objects that are retrieved relative to the total number of relevant objects in the data set [Han+01]. Recall indicates how many of the actual positives can found. Recall is sometimes also called sensitivity or hit rate.

$$Recall = \frac{TP}{TP + FN}$$
$$Precision = \frac{TP}{TP + FP}$$

**Specificity**

Specificity is the true negatives rate. It measures which of the negatives are actually classified as such. Specificity can be seen as a counterpart to the precision.

$$Specificity = \frac{TN}{TN + FP}$$

**F-Score**

Is the harmonic mean of precision and recall and results in the following formula.

$$F1 = 2\frac{precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN}$$

A more general F-Score also includes a weighting between precision and recall. When precision and recall are equally weighted, the above formulas arise. For imbalanced datasets the F-Score provides much more representative results than accuracy does, but is still sensitive to class label distributions [He+09].

**Receiver operating characteristics ROC**

The ROC curve is a useful classifier evaluation tool and visualizes sensitivity (true positive rate) and specificity (true negative rate). The ROC area is a single scalar that represents the area beneath the curve. A big ROC area implies a good model fitness. [Faw06]

The ROC curve arises, when thresholding the test set [Faw06]. Such a threshold or scoring parameter could be the probability of the instance belonging to a specific class. This means, if the classifier wrongly predicts instances, where it assumed a high certainty to be of some class, the curve turns out to be more linear. The drawing of the curve starts at point $(0, 0)$, with the instance with the highest score. In the worst case, every instance is predicted erroneously and the ROC curve is completely linear. Figure 2.8 depicts three examples of ROC curves. Curve A predicts nearly all instances in the test set right, while curve C contains no right predictions. For matters of completeness, true positive rate can be derived from the above mentioned specificity or true negative rate.

The previously listed validation metrics can be categorized as measures for "model goodness". Those only mentioned only form the tip of the iceberg. Of course, other criteria for model evaluation can be taken into account like speed, robustness, scalability, interpretability, flexibility and time complexity [Gup06].

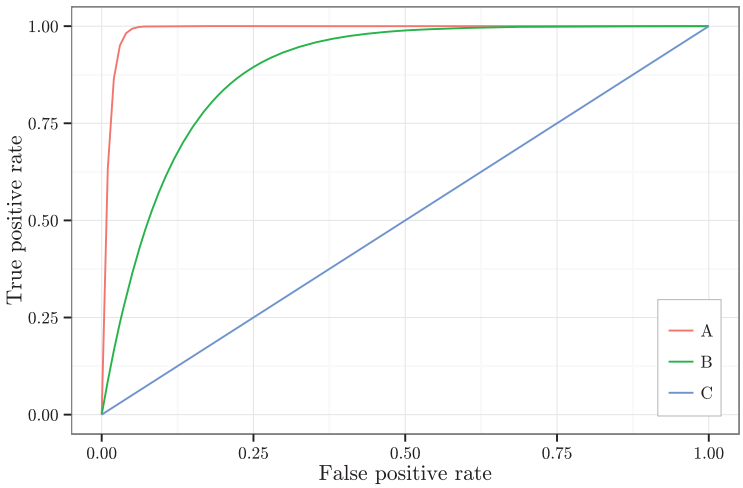



Figure 2.8: ROC curves

### 2.2.3 Unsupervised Learning

So far only classification methods that build and train models based on instances with classes known a priori were discussed. If, however, classes and class labels are unknown at the beginning, a new kind of data mining technique steps into breach – clustering. With clustering "one does not know what classes or clusters exist, and the problem to be solved is to group the given data into meaningful clusters" [Gup06]. Guptas statement emphasizes the major task and problem perfectly, and shows the clear distinction to classification tasks, which is obviously the a priori presence of class labels at data instance level. Most common methods are hierarchical and partitional methods, but there exist also grid- and model-based approaches. The complete taxonomy of existing clustering techniques is illustrated in figure 2.9. However, most of the clustering or unsupervised learning algorithms use distance metrics for grouping nearby data points. Widely used distance metrics are the Euclid distance, Manhattan distance, Chebychev distance but also the categorical data distance for distance measurement of categorical attributes. A detailed description on the depicted distance metrics, as well as general properties and usage scenarios can be looked up in [Gup06, p. 169p]

#### Partitional Methods

The K-means clustering technique is a classical partitions approach and its ease to use and simplicity makes it a powerful ally in clustering problems. The parameter $k$ indicates the number of clusters, which the algorithm will produce. To begin $k$ seeds have to be picked (randomly or with insights to the data) from the data points, which serve as initial clusters. Each data instance is then allocated to its nearest cluster, i.e. to the nearest centroid. Then a new centroid is calculated for each cluster and a new iteration starts until a stop condition is met – e.g. the clusters do not change and therefore the

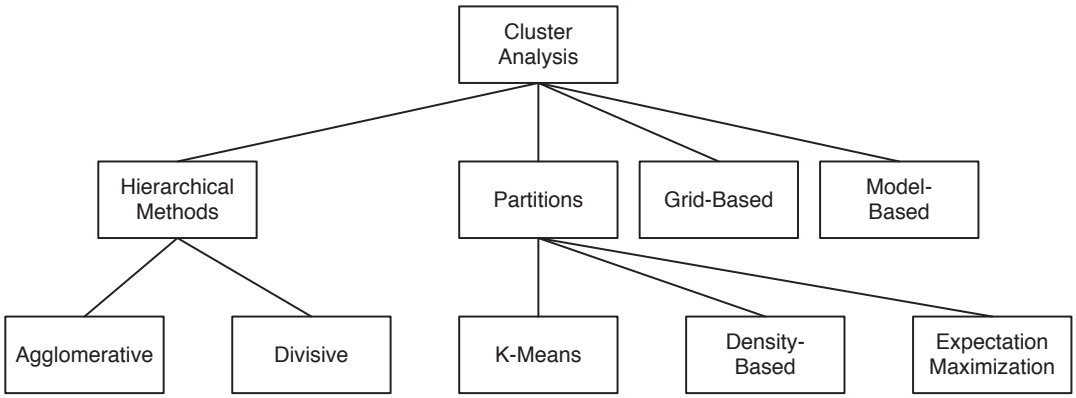



Figure 2.9: Taxonomy of cluster analysis methods [Gup06, p. 171]

algorithm has converged to a local optimum. The algorithm is called k-means because the cluster is represented by the means of the containing objects. [Gup06]

**Hierarchical Method**

Hierarchical clustering produces a nested series of clusters, while partitional methods produce only a flat set of clusters. It can be categorized in agglomerative and divisive methods. Agglomerative clustering starts with $n$ clusters which are merged step by step into larger clusters – they use a bottom-up strategy. The merging mechanism is based on inter cluster distances. On the other end, we have divisive clustering techniques that break down big clusters into several small ones. [Gup06]

Gupta [Gup06] lists the additional insights into the data by the presentation of hierarchies, the simpler concept compared to partitional clustering, as advantageous when only proximity data is available and different granularities of clusters as pros of hierarchical clustering. Since data mining, or more general knowledge discovery in databases (KDD) refers to the overall process of discovering useful knowledge from data [Fay+96], the additional insights into the data and presentation of different granularities can be crucial, and are considered as the most important advantages. Last but not least, disadvantages include the lack of cluster reassignment, the time and space complexity as well as the sensitivity to different distance metrics [Gup06].

## 2.3 Approaches to Crowdfunding Analysis

Basically crowdfunding analysis as field of study is quite originary. One of the few representative pieces of work is Ethan Mollick's "The dynamics of crowdfunding", published in the Journal of Business Venturing [Mol14]. In his work, he examines crowdfunding with methods from venture capitalism and entrepreneurial research. He discovered that mostly

projects fail by large amounts. The mean funding amount of failed projects is 10.3% of the goal. On the other hand successful projects succeed by small amounts, i.e. 25% of the successful projects are 3% or less above their goal, and only 50% are about 10% above their goal. Secondly Mollick states, that signals of quality and the preparedness of the founder are essential for the project success. Therefore he examined quality metrics like video presence, Facebook friends and the founder‘s network, updates after launch, and the absence of spelling errors. Another important finding is, that in crowdfunding geographical effects play a non-negligible role, although he also observed that a reduced importance of geographic constraints is present. Mollick‘s article delivers useful basic insights into the matter of crowdfunding dynamics and gives a well-presented guideline for further project analysis.

Greenberg et al. [Gre+13] use a variety of classification algorithms to predict project success or failure by merely static factors. They used pre-scraped Kickstarter projects from thekickbackmachine.com with the attributes project goal, category, reward count, duration, Twitter and Facebook connection, Facebook friends and Twitter followers, video presence, sentiment, Flesch-Kincaid readability metrics, number of sentences as independent variables and the project success as label. The data was applied to a range of different classifiers for their analysis, including several decsion tree induction algorithms, random forests and support vector machines. Their analysis topped at 68% accuracy, which is 14% above their baseline, which was predicting all projects as failed. Additionally, they found that random forest are a very good classification algorithm for this specific data mining task. Since the paper gives no information about data preprocessing and cleansing, it can be assumed, that not much was done in this respect. In general this concise paper gives a good first insight on possibilities of data mining algorithms in this area of research.

Mitra and Gilbert [Mit+14] took a very interesting approach by assuming the success of crowdfunding projects depends highly on used phrases in the project and reward description. Their thought is, that project owners that are using persuasion techniques are more successful. Additionally they use different additional control variables – in fact 59 additional variables including project goal, project duration, number of pledge levels, video presence, video duration, categories and many more – to predict the success of crowdfunding projects. By only including those 59 control variables they could improve their error rate from 48.47% baseline to a value of 17.03%, and, following with the inclusion of the phrases, another improvement down to an error rate of merely 2.24% could be achieved. Their baseline was assuming all projects as successful. With solely the paper at hand, and thus without a replication of the model, a percentage of more than 97% accuracy by only including the phrases approach, however, seems a little bit too good to be true. A possible reason for this could be found in biases within the data model. Nevertheless, this innovative approach would take the same line as Mollick indicating yet another measure for signal quality.

The papers on data mining for crowdfunding mentioned earlier solely concentrated

on crowdfunding data analysis based on static attributes. Those can be very handy for predicting success, even when the project has not yet been launched. Etter et al. [Ett+13] introduced a novel dynamic way of crowdfunding project analysis, by examining projects over time. With this approach they want to find out at which time in a projects online time the projects success or failure can be foreseen. By using the k-nearest-neighbors classifier and Markov chains they are able to predict the projects outcome state after 15% of the projects time with an accuracy of 85% based on the pledged money. Additionally they created a social predictors model and a combined model as well. The latter led to an accuracy increase of 4% within the first moments of the campaign.

## 2.4 Summary

For the following analysis, a deeper knowledge of a range of supervised and unsupervised learning algorithms is needed. This chapter therefore provides descriptions of the algorithms used within the crowd data analysis tasks. Those algorithms, after having been examined theoretically, will then be used later on to properly answer the research questions defined earlier. The existing literature, on the other hand, did provide a solid starting point for the analysis feature selection process and gave useful information in regards to which algorithms work properly.

Following the successful establishment of a common theoretic foundation for this thesis, data needs to be collected from different sources, integrated, cleansed and analyzed by means of initial exploratory statistics. The following chapter covers exactly these unavoidable and essential tasks.

CHAPTER 3

# Domain and Data

This section attempts to bring the domain and data understanding phases of the CRISP data mining model to life. Since business objectives are already defined within the introductory chapter, this section starts out with a short revision of the crowdfunding domain, followed by an examination of boundary conditions and general assumptions. Generally, this section shall attempt to provide a cursory examination of the domain and the possible implications on data mining. In its following thematic block, this section deals with data collection. This is obviously necessary to provide a first look at the data and have initial exploratory statistics for the purpose of building up data understanding. Finally, the data quality and necessary data preparations will be discussed within this section.

## 3.1 Domain

For simplicity reasons, this thesis only examines reward-based crowdfunding, and within this category only the campaigns from Kickstarter. Other crowdfunding fields like equity crowdfunding or donation-based crowdfunding are excluded, since an examination beyond the borders of reward-based crowdfunding would obviously introduce additional complexity and completely new research questions. The selection of Kickstarter as data base for the analysis has two pragmatic reasons: first, Kickstarter as popular reward-based crowdfunding platform provides a reasonable big population of campaigns to analyze and secondly the projects on Kickstarter are partly accessible via a JSON API, granting a structured and easy access.

The terms "'project" and "campaign" are used synonymously within in this thesis and the term "project" solely relates to the activities concerning crowdfunding, not on the real world matter which is sought to be funded. For example, if a user wants to receive funding for a comic book, the process of conception, drawing, funding and publishing can

be interpreted as a project as well, but the widely used project term relates to the sub project of crowdfunding only. In short, it means "crowdfunding project" in the context of this work.

The principal of reward-based crowdfunding is that a backer – who is a person supporting a project by donating money – receives a non-financial reward in return. The typical case of reward-based crowdfunding starts with the creation of a campaign. The creation covers the preparation of the project page and settings, namely goal, video, pictures, rewards, description and many more. Before the campaign is launched the project or campaign owner has usually the possibility to present a preview version to selected users, in order to conduct a small test run. The time between creation and launch will be called lead time. Following the publication of the campaign, persons have the possibility to supply the campaign with financial resources; this process, however, is called pledging. The crowdfunding project ends on a specific date, after a predefined period. With the end date, the time window for funding closes. In practice, two types of funding strategies have emerged: "All-or-Nothing" (AON), and "Keep-it-All" (KIA) [Cum+14]. With "All-or-Nothing" strategies, the founder only gets the pledged money, if the predefined goal was met. With "Keep-it-All" the project owner gets the resources anyway and can decide whether the funding is sufficient or to pay everything back otherwise.

While the description of crowdfunding was intentionally kept very general, let us now have a closer look at the Kickstarter peculiarities and characteristics. The platform selects a live duration of 30 days per default, which is assumed as an optimum. The scale goes up to a maximum of 60 days. The default selection of 30 days and the maximum of 60 obviously leads to a skimmed distribution within this attribute. Additionally, Kickstarter offers only an "All-or-Nothing" funding strategy to its users. This means, users must reach their goal. This could possibly lead to the effect of self-funding, if the goal would be missed by a small amount [Mol14]. Other preconditions for the creation of projects are a founder of full age and permanent residency in the country named as project country, with a verified address, bank account and ID. This reduces possible regional biases, and one can assume that the presented project location is correct. Additionally there are a number of other rules, which project owners need to take into account, for example that neither financial rewards are allowed, nor the publication of charity projects. For the interested readers, a more detailed view on Do's and Don'ts is provided at the rules page of Kickstarter[1].

Kickstarter has mechanisms to directly intervene with the project. For one, they are able to negatively affect projects by suspending them if they are contradicting to the community guidelines. On the contrary, positive actions can be set by featuring special projects. This mechanism is called "staff picks" on Kickstarter. A Kickstarter dedicated blog post describes influencing factors on "staff pick" selections [Abe15], like picture

[1] https://www.kickstarter.com/rules

selection, quality of media in general, expressiveness of descriptions and the absence of spam. These selection criteria are kept very vague, because of the understandable reason, that it is nearly impossible to give a hard facts-based answer to a subjective selection process. However, one needs to keep in mind, that such a selection mechanism entails the potential of unwanted biases. The homepage of Kickstarter again displays a subset of the "staff picked" projects, and even selects one as "project of the day". According to Kickstarter, those decisions are made within daily editorial meetings. In general these "staff picks" are assumed as influencing factors too, since one can see them as an indication of preparedness which was evaluated by the Kickstarter team.

Beside the campaign activities directly on the Kickstarter platform, there exist external influence factors on projects. These include for example activities on Facebook and Twitter. The related papers recognized this fact and included such factors in the analysis as well. These external factors can again be mapped to the concepts of preparedness and signal quality from entrepreneurial and venture capitalism theory. The close relationship of crowdfunding to entrepreneurship is underlined by Belleflamme's [Bel+10] statement: "The basic idea is always the same: instead of raising the money from a very small group of sophisticated investors, entrepreneurs try to obtain it from a large audience, where each individual will provide a very small amount".

## 3.2 Data Collection

The data serves as a fuel for the data analysis engine. Without the right and enough data the mining becomes obsolete and the engine cannot produce any insights to the topic under examination. Evidently, the data collection step is quite as essential as the performed analysis itself. In this analysis the "fuel" is crowdfunding projects from Kickstarter. The projects are collected by tapping the free for non-commercial-use online resources of the platform.

The online database contains over 200,000 projects. They are accessible by list or by detail. A list page contains 20 projects, with some basic information. The detail page on the other hand is the actual project page and contains more detailed information. The HTTP requests for the listing and the detail are outlined in figure 3.1. The bold parameters in the represented GET requests are the default API parameters. The possible values for the parameters "goal", "pledged" and "raised" are masks for value classes. The "raised" parameter for example, represents the percentage of goal achievement and has got classes of "smaller than 75%" with label 0, "between 75% and 100%" represented by label 1 and "more than 100%" with label 2. A very important parameter is the format parameter, with which one can select the desired output format. Luckily the project can be delivered in the easily processable JSON format. A positive side effect of the JSON format for lists is, that additional information is delivered compared to HTML.

```
GET /discover/advanced[?
          [&format=(html|json)]
          [&sort=(magic|newest|end_date|
               popularity|most_funded)]
          [&state=(all|live|successful)]
          [&goal=(all|0|1|2|3|4)]
          [&pledged=(all|0|1|2|3|4)]
          [&raised=(all|0|1|2)]
          [&woe_id=<location_id>]
          [&page=<page_number>]
     ] HTTP/1.1
Host: www.kickstarter.com
```

(a) The request for accessing the project list

```
GET /projects/<user_id>/<project_slug> HTTP/1.1
Host: www.kickstarter.com
```

(b) The request for accessing the detail page

Figure 3.1: Used HTTP requests for accessing kickstarter projects

According to the defined research objectives, two data collection tasks arise: the dynamic project factors collection task and the collection task for static factors. In order to derive the dynamic project developments, the state of all live projects must be recorded over time. The current state is a vector containing the number of pledges, backers, Facebook likes, shares and comments, and Twitter tweets. The time lag between two data records for one project was chosen one day. More precisely all states of live projects were gathered at one specific time on one day. The interval of one day was considered sufficient since points in between can be linearly interpolated. Assuming the project development has the form of a square root function then the error[2] of interpolation with 30 observation points, one per day for a project duration of 30 days, is below 2%. When assuming a logarithmic behavior the error is even lower. Luckily all Kickstarter related information that is needed for the project state is covered within the list request. All social media data is gathered through the Facebook graph API and Twitter API. The dynamic state collection algorithm is depicted in algorithm 3.1, only imagine the

[2]For the sake of completeness, the area between the square root function and the interpolation rated by the total area under the function was used as a quick and dirty error measure

algorithm without the last loop:

**Algorithm 3.1:** Project collection algorithm

```
   Data:
   Let L and P be sets
   Let M be a set of collected live projects from the last run
 1 P = ReadNextLivePage();
 2 while |P| > 0 do
 3 |   for Project p ∈ P do
 4 |   |   f = ReadFacebookStats(p);
 5 |   |   t = ReadTwitterStats(p);
 6 |   |   StoreState(p, f, t);
 7 |   |   L = L ∪ p;
 8 |   end
 9 |   P = ReadNextLivePage();
10 end
11 for m ∈ M \ L do
12 |   p = ReadDetailPage(m);
13 |   f = ReadFacebookStats(p);
14 |   t = ReadTwitterStats(p);
15 |   StoreState(p, f, t);
16 end
   /* Only needed for the statics collection                    */
17 for n ∈ L \ M do
18 |   p = ReadDetailPage(n);
19 |   StoreStaticAttributes(p);
20 end
```

Although algorithm 3.1 presents a simplified version of the real algorithm, it outlines the main concepts. First one live project page after the other is read and the states including the Facebook- and Twitter-stats are stored. The last part of the dynamic collection algorithm iterates through all projects that were live on the last collection run and are not anymore, and reads the project page in order to determine the final state. Here again the social media stats are read and stored. With the dynamic algorithm in mind, the collection of static project attributes, for the static success analysis is now very easy. By a slight adaption at the end of algorithm 3.1, not only the project state of newly found project is stored, but at the end all other necessary information is stored as well.

As mentioned slightly above, the actual algorithm is clearly more complex. Its general structure can be observed in figure 3.2. The collection algorithm is built around the resource interface with a get method. Resources in this sense represent (remote) information data stores. In this case, resources are mostly accessors to crowdfunding projects and project pages, which are more or less wrappers for the Kickstarter API

calls, but also resources for categories or users are imaginable. The resources above formulated postconditions, in terms of the design by contract software design paradigm, are therefore intentionally held very unspecific to get the concepts of different resources to reconcile. While concrete crawlers access real resources, like webpages, web services, or even databases, to gather relevant data, KckMultiPageResource acts as an operator, accessing multiple page resources until an empty page is delivered. From the class diagram it becomes obvious, that some kind of decorator pattern is used for this concept.

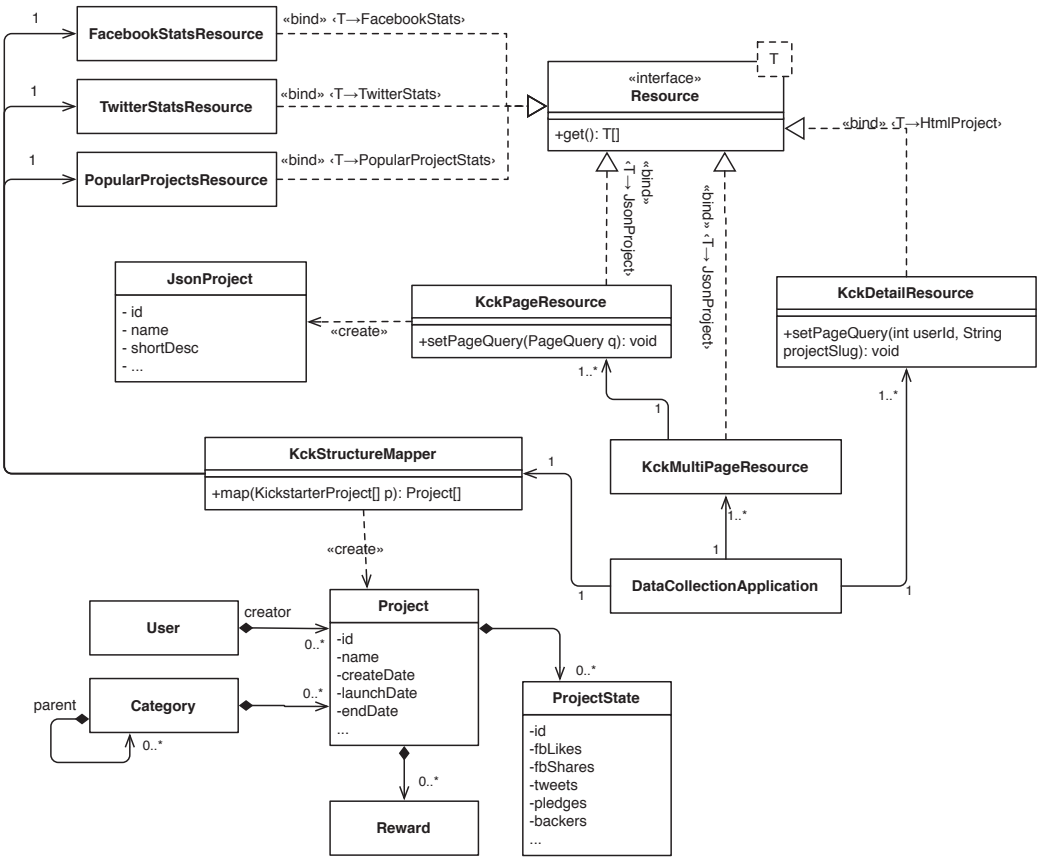



Figure 3.2: Class diagram of the data collection engine

Until now, the algorithm has only read the JSON files from the API. This is where the KckStructureMapper component steps in and "translates" those JSON files into common crowd data projects. Unfortunately, the page resource cannot receive all necessary attributes from one source, therefore the mapper not only transforms the projects, but also extends the projects with attributes gathered from other resources. Examples of such attributes are Facebook likes, shares and comments; or Twitter tweets. Additionally, this mapper takes care over some data cleansing functionality, for example by transforming

time-stamps to date formats.

As is very common in online APIs, Kickstarter prohibits from overloading the servers with requests and restricts the access to 600 requests per hour, before the applications IP address gets a timeout. In order to get reasonable performance in the application a request control unit was implemented, which allows the to user control the time intervals of requests going out. This mechanism is depicted in figure 3.3. The request control unit is plugged into a "Requestor unit" and regulates the outflows. This is especially important since KckMultiPageResource is used in a multi-threaded environment. The dynamically gathered projects over time ranged from 5700 to 6200 per day. With 20 projects per page about 285 to 310 requests were generated for the live projects part. On average around 200 projects were started per day, which is also the number of final states to collect. The dynamic collection algorithm needed about 485 to 510 requests per day and therefore never exceeded the 600 requests threshold. Since an additional 200 requests per day are needed for the collection of the static project attributes and after the dynamic process nearly exhausted all 600 permitted requests per hour, the static collection will be executed at a later time on the same day. The reason for not just reading the static projects together with the final state is that, with a combined approach possible biases could be introduced. This should be emphasized by the following example: Facebook friends of the project creator are, if presented, scraped from the projects detail page. Obviously a crowdfunding campaign could influence the number of Facebook friends. More formally $\frac{\partial f}{\partial c} \neq 0$, where $f$ is a function determining one's Facebook friends and $c$ the campaign. As a result the Facebook friends on the final state may be biased by the campaign itself, while the number of friends at the beginning are not.

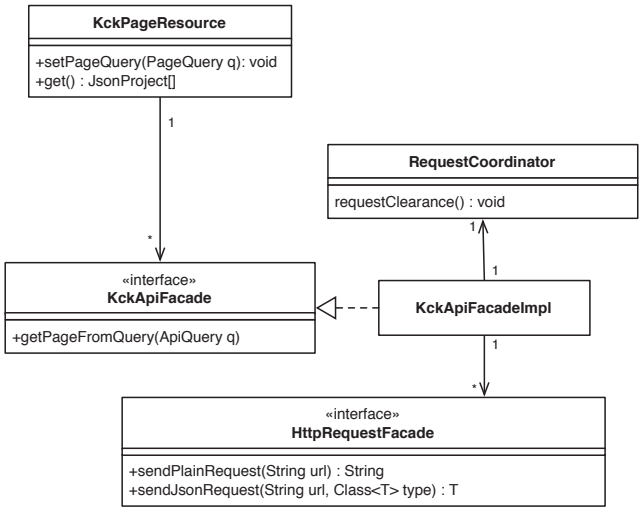



Figure 3.3: Classes for request regulation

The issue of tapping different data sources was already raised above, by mentioning the necessity of deriving Facebook and Twitter stats. An additional external data sources is Genderize.io[3], an API that assigns gender to user names. The idea of including gender factors in crowdfunding was taken from Greenberg and Mollick [Gre+14] and was extended by identifying the user type. A user type can be a person, a company or other. A company is characterized by the inclusion of the type of the business entity. However, it was differentiated between limited and public limited business entities as well as incorporation types of topmost countries within the data set.

The collection algorithms were written in pure Java, using Apache Maven as build management tool, with Hibernate as an object-relational-mapper, Guice for dependency injection, Log4j for logging, Gson for JSON to class mappings and Jsoup API for HTML scraping. Additionally Xuggler was used for video processing in order to analyze video duration and bitrate and Pushover for error reporting. Last but not least JUnit was selected for unit testing with Mockito as a mocking framework. The application is executed through a CRON-job, invoking an executable Java jar-file on an Amazon EC2 Micro-instance. The EC2 Micro with one virtual core of up to 3.3 GHz, 1 GB main memory and 8 GB EBS (Elastic Block Storage, which is a Amazon specific persistent block based storage) is perfectly sufficient for this purpose. The collection algorithm is started each day at 7:00 UTC, where the least traffic is estimated on Kickstarter[4], in order to least affect the platforms functionality. In case of errors the application sends out status reports via the smart-phone push API of Pushover.

## 3.3 Data Structure and Statistics

The basic data structure from the collected data, illustrated in figure 3.4, has a lot in common with the snowflake star schemes and can therefore be viewed as a two-dimensional data cube, with the dimensions project and time. For the purposes and the goals of this thesis the two dimensional approach is sufficient. Nevertheless, the introduction of additional dimensions could provide a more detailed view on the data and could potentially open possibilities for a deeper examination. For example by installing a user dimension, the analyst could identify which user pledged how much money on which day and consequently could discover detailed pledge behavior patterns. The structure in general arose from some iterations through collection and examination phases. The new iterations meta data and settings were based on the knowledge from the crowdfunding literature and previous examinations and experiments with the data. A complete list of all attributes can be found in appendix A.1.

Before diving deeper into a first crowdfunding analysis, it should be mentioned that these initial examinations and exploratory statistics should serve to build up a solid data understanding to start from. Additionally, the explorations were repeated several times until the final dataset was found, which makes it possible to answer the research

[3] https://genderize.io/

[4] Since the traffic could not be measured directly the launch time of projects was taken as an estimate.

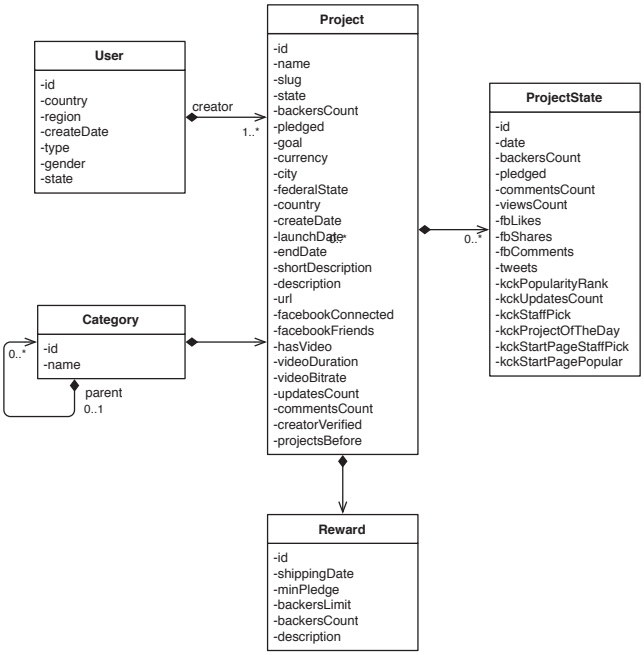



Figure 3.4: The data cube used for the analysis

questions reliably. Also, these exploratory statistics already partly answer to the first research question, where success and failure drivers are being sought.

The crowdfunding data set contains in total 14,369 crowdfunding campaigns with $156,432,029 money pledged. The projects were collected in the time span between June and October 2015. A general overview is provided in table 3.1. Table 3.2 on the other hand depicts the success drivers vs. drivers of failure. According to the table, a lot of content metrics are ranked very highly, additionally rewarding strategy is very important, as is experience in form of the attribute "projects before". Interestingly enough, the final number of likes of a project and the number of backers do not play as essential of a role as standalone factors as assumed. According to the attributes‘ correlations, the most important blocker is the duration. Mollick [Mol14] suspects that "longer durations are a sign of lack of confidence" and therefore have a negative effect on campaigns. Additionally, some categories like technology and food act as blockers in this simple scenario. Also, regional aspects seem to play a big role for projects originating from Southern or Western Europe. Last but not least, the correlation matrix identifies persons, especially with male gender, as less likely to be successful.

### 3.3.1 Categories and their Implications

The first examination of the top attributes most correlating with success has already shown, that categories play a non-negligible role. Table 3.3 illustrates that the categories

| | Successful | Failed | Total |
|---|---|---|---|
| Projects | 4,757 | 9,612 | 14,369 |
| Proportion | 33.1% | 66.9% | 100% |
| Backers (non unique) | 1,517,384 | 139,721 | 1,657,105 |
| Pledged | $144,496,949 | $11,935,080 | $156,432,029 |
| Likes | 2,662,630 | 667,741 | 3,330,371 |
| Shares | 1,161,987 | 468,068 | 1,630,055 |
| Tweets | 1,135,070 | 411,783 | 1,546,853 |

Table 3.1: Dataset overview

| Top success correlations | | Top failure correlations | |
|---|---|---|---|
| rewardCount | 0.33* | duration | -0.14* |
| picCount | 0.28* | category_Technology | -0.11* |
| hasVideo | 0.26* | gender_MALE | -0.11* |
| outOfDict | 0.26* | currency_EUR | -0.10* |
| wordCount | 0.24* | type_PERSON | -0.09* |
| sentenceCount | 0.24* | subRegion_Southern Europe | -0.07* |
| paragraphCount | 0.22* | category_Food | -0.07* |
| videoBitrate | 0.18* | subRegion_Western Europe | -0.06* |
| projectsBefore | 0.17* | country_IT | -0.06* |
| videoDuration | 0.17* | sent_neg | -0.06* |

* $p < 0.05$

Table 3.2: Top success and failure correlations

music, film & video, games, art, comics, theater and dance have (significant) positive correlations to success while fashion, technology, photography, journalism, crafts, publishing and food mark negatives ones. The reasons for these effects are manifold. There would be enough substance to base several complete theses about just analyzing the categories and their certain success drivers. Although the thesis‘ objectives are located in different corner of crowdfunding analysis, I intend to provide at least some possible explanations to differences of success correlations for categories, because they seem rather important, according to the top success and failure correlations.

Although the projects final state decides whether a campaign failed or succeeded in the end, success is not as black and white as one might think. If a projects fails to reach the funding goal but still collects a big amount of Facebook likes or shares and attracts fans over the whole world, it should not be considered as a complete failure. Consequently, the correlations concerning end state, pledged money and supporters, as well as social media factors are analyzed, which are all assumed as indicators for success and included into the correlations in table 3.3.

| | Success | Pledged | Backers | Likes | Shares | Tweets |
|---|---|---|---|---|---|---|
| Art | 0.03* | -0.02 | -0.02* | -0.01 | -0.02* | -0.01 |
| Comics | 0.11* | -0.01 | 0.00 | 0.00 | 0.00 | 0.01 |
| Crafts | -0.03* | -0.01 | -0.01 | -0.02* | -0.02* | -0.01 |
| Dance | 0.03* | -0.01 | -0.01 | -0.01 | -0.01 | -0.00 |
| Design | 0.01 | 0.05* | 0.05* | 0.05* | 0.08* | 0.00 |
| Fashion | -0.06* | -0.01 | -0.02 | -0.02* | -0.02* | -0.01 |
| Film & Video | 0.02* | -0.01 | -0.02* | 0.04* | 0.02* | -0.00 |
| Food | -0.07* | -0.01 | -0.02* | -0.01 | -0.02* | -0.01 |
| Games | 0.07* | 0.04* | 0.09* | -0.00 | 0.01 | 0.01 |
| Journalism | -0.03* | -0.01 | -0.01 | -0.01 | -0.02* | -0.00 |
| Music | 0.05* | -0.02* | -0.03* | -0.01 | -0.03* | -0.01 |
| Photography | -0.02* | -0.01 | -0.01 | 0.00 | -0.01 | -0.00 |
| Publishing | -0.03* | -0.02* | -0.02* | -0.02* | -0.02* | -0.01 |
| Technology | -0.11* | 0.02* | 0.01 | 0.00 | 0.02* | 0.02* |
| Theater | 0.09* | -0.01 | -0.01 | -0.01 | -0.01 | -0.00 |

* $p < 0.05$

Table 3.3: Success correlations of categories

**Content factors:** As we have seen, several of the top success correlations concern the project description and content metrics. Therefore, figure 3.5 illustrates the differences in content presentations. While comics, design and fashion use a high amount of pictures compared to text, journalism, music and crafts are very text-heavy. Also the success correlations change according to the category. While design success correlates to the number of pictures with $r_{\text{success,picCount}} = 0.35$ the correlation to sentences is much lower with "only" $r_{\text{success,sentenceCount}} = 0.22$. With journalism this is slightly upside down, with $r_{\text{success,picCount}} = 0.25$ and $r_{\text{success,sentenceCount}} = 0.26$. This shall emphasize, that when performing an analysis with the CrowdData framework as in chapter 5 the user shall always analyze the content attributes, in order to find the optimal setting for them within the defined category.

**Regional factors:** In order to examine regional factors, the sub regions are plotted against the categories. The resulting figure (3.6) shows the success rate for each category and region. For example fashion and journalism have higher success rates, and technology and fashion at least equal ones, if they have originated from Western Europe instead of Northern America.

The calculation of significance for success rates being superior to others was omitted, since the intention of this graph is to only give a brief overview. Therefore the results must be assessed critically and should not be taken at face value. Additionally, some outlier values must be examined with care. This outlier introduced "blurriness" stems from

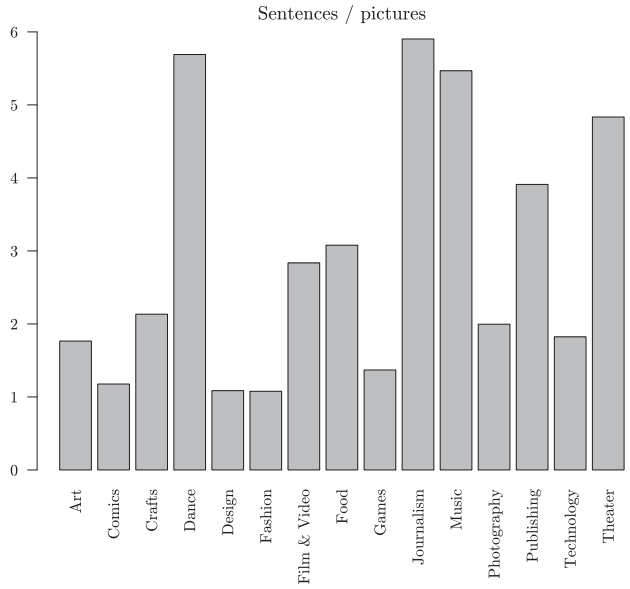



Figure 3.5: Presentation differences across categories

the reduced number of projects outside of the US. This is especially true for Southern Europe, where only a little number of campaigns originate from.

**Audience factors:** Since there are possible influencing factors on the projects creator origin, it is assumed that similar factors exist on the backer side as well. Not only regional audience factors, but also platform biases are imaginable, where users with certain interests prefer certain platforms. To put it differently, design-oriented users might prefer other crowdfunding platforms than technology-oriented users do. A possible indicator could be differing correlations in likes and shares across the categories, as can be observed in table 3.3. A more detailed analysis of such effects would need a detailed examination of the backers on a specific platform, for example by determining general interests of backers.

As seen above, one of the top failure correlations are the technology and food categories. With the knowledge generated in this section one cannot precisely determine the factors for these highly negative – as compared to others – success influences. However, it is more likely that the reason is a combination of all mentioned factors.

### 3.3.2 Content Quality Signals

Clearly, a lot of calculated content metrics have a high impact on the project success. This justifies a brief separate examination. While a positive correlation of number of pictures,

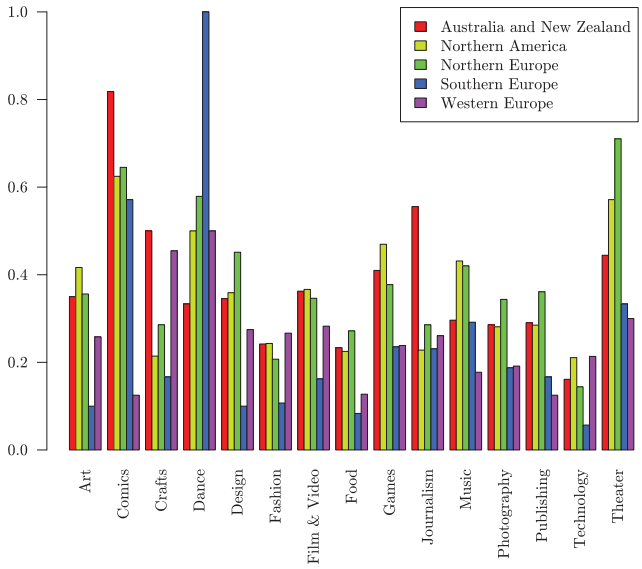



Figure 3.6: Regional differences in campaign success across categories

words, sentences and paragraphs can be expected, the fact that out-of-dictionary words are also positively correlated is peculiar (table 3.4). The out-of-dictionary measure was originally introduced to incorporate effects of spelling mistakes into the model, therefore this attribute was expected to be negatively correlated. Of course, not only spelling mistakes are included into this measure, but also proper nouns and words invented by project owners. Apparently the presence of such proper nouns are a quality signal and a sign of preparedness and displace the spelling mistakes impacts. Clearly, writing a generic text with no proper names would result in a low out-of-dictionary metric, but would also signal more uncertainty to the backers. In other words, project owners that come up with creative names for their product, might be rewarded.

Figure 3.7 shows the histogram and density function of the top success correlation factors. Clearly, normal distributions are hard to argue for this content attributes, since the distributions are highly skewed. This is supported by Micceri [Mic89], who underlines that in real world applications true normality is a rather rare phenomenon.

### 3.3.3 Rewarding

As we have seen in table 3.2, the correlation matrix indicates a highly positive correlation between number of rewards and success. Furthermore there are also significant correlations to other success indicators ($r_{rewards,pledged} = 0.12^*$, $r_{\text{rewards,backers}} = 0.16^*$, $r_{\text{rewards,likes}} = 0.22^*$, $r_{\text{rewards,shares}} = 0.22^*$ and $r_{\text{rewards,tweets}} = 0.06^*$ with * indicating $p < 0.05$).

| | Success | Pledges | Backers | Likes | Shares | Tweets |
|---|---|---|---|---|---|---|
| wordCount | 0.24* | 0.12* | 0.15* | 0.14* | 0.16* | 0.06* |
| sentenceCount | 0.24* | 0.13* | 0.16* | 0.13* | 0.16* | 0.06* |
| paragraphCount | 0.22* | 0.11* | 0.13* | 0.12* | 0.14* | 0.05* |
| picCount | 0.28* | 0.21* | 0.24* | 0.18* | 0.22* | 0.10* |
| hasVideo | 0.26* | 0.06* | 0.08* | 0.10* | 0.11* | 0.03* |
| outOfDict | 0.26* | 0.14* | 0.17* | 0.15* | 0.17* | 0.09* |
| sent_neg | -0.06* | -0.01 | -0.02 | -0.01 | -0.02* | -0.01 |
| sent_neutral | -0.03* | -0.01 | -0.01 | 0.02 | 0.01 | 0.01 |
| sent_pos | 0.06* | 0.01 | 0.02* | -0.01 | -0.00 | -0.01 |
| langEn | 0.05* | 0.01 | 0.01 | 0.02 | 0.02* | 0.01 |
| langDe | -0.05* | -0.01 | -0.01 | -0.01 | -0.01 | -0.00 |
| gunningFogIndex | -0.05* | -0.02* | -0.02* | -0.02 | -0.02* | -0.00 |
| fkReadabilityScore | -0.01 | -0.01 | -0.00 | -0.02* | -0.02* | -0.01 |

* $p < 0.05$

Table 3.4: Correlations of content factors

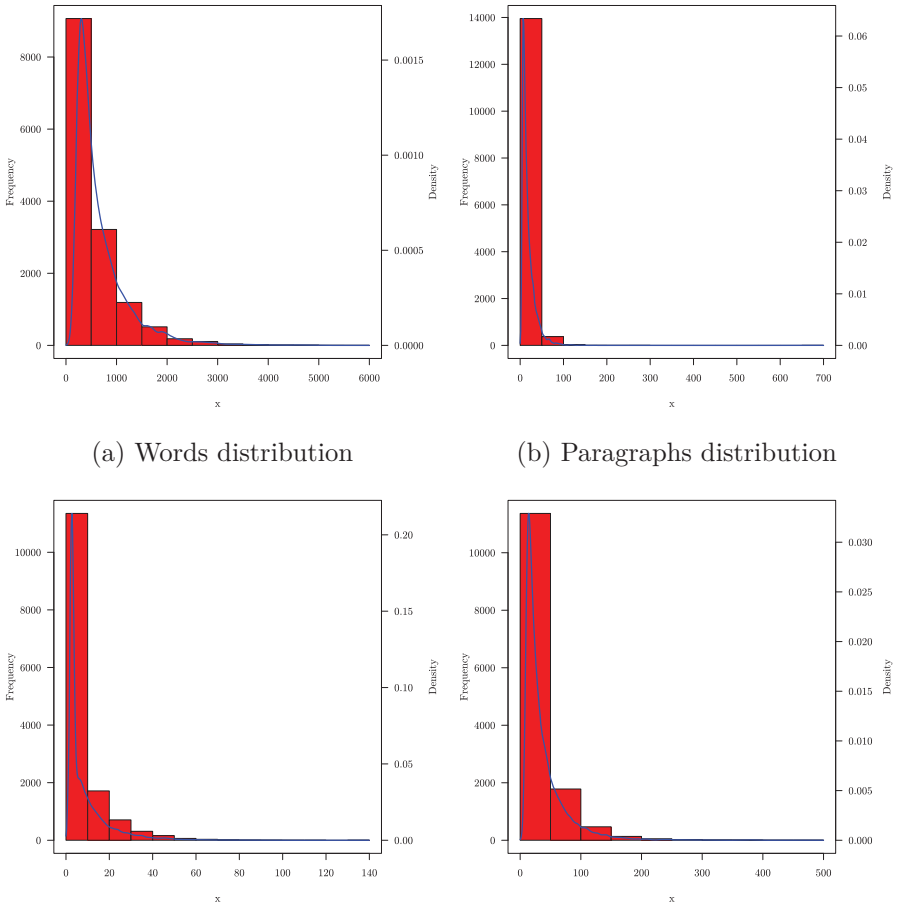
This clearly indicates the importance of rewarding strategies. This initial examination was responsible for the introduction of additional rewarding attributes like minimum, maximum, mean and standard deviation reward levels. For simplicity reasons the analysis will be performed with these rewarding strategy representatives only. Other imaginable rewarding attributes are: a flag indicating the presence of early bird rewards, content metrics for rewards and delivery dates.

### 3.3.4 Crowdfunding Goals

The fact that crowdfunding goals are slightly correlated to failure ($r_{\text{success,goalUsd}} = -0.03$ with $p < 0.05$) is not very surprising. It is interesting though, that there are no apparent correlations to the projects amount pledged and the number of supporters. This may be an indicator, showing that the users do not take the goal into the decision process, whether to support or not support a campaign. The quartiles in table 3.5 indicate that the majority of projects is located in the goals segment below \$10,000 – in fact 64% of all projects. Nearly one quarter of all project goals are located even below \$2,000. The mean for goals is, however, very biased with a value of \$67,967.94 through some outlier goals of over \$1 million. The histogram for project goals and the density function is depicted in figure 3.8. The histogram shows a highly skewed density with a long tail reaching out for the 1,367 campaigns (9.5%) with goals over \$50,000.

### 3.3.5 Companies vs. Persons, and Genders in Crowdfunding

As mentioned earlier, Greenberg and Mollick [Gre+14] took a look on the performance of genders in crowdfunding, and suggested that “the presence of female backers explains




(a) Words distribution

(b) Paragraphs distribution

(c) Pictures distribution

(d) Out-of-dictionary distribution

Figure 3.7: Distributions for content metrics

some of the comparative success of female-founded projects. However, rather than being driven purely by better numerical representation of women in a given industry category, the success of female founders seems to require a subpopulation of female backers that disproportionately support women founder in areas in which women are historically underrepresented – activist choice homophily". These finding were reason enough to include gender attributes into the model.

The top failures in table 3.2 somewhat underline Greenberg and Mollick's conclusion, by depicting male gender as one of the top drivers for failure within correlation analysis. Testing female against male success with $\mathcal{H}0$ that they are equal, against $\mathcal{H}1$ female outperform male results, leads to a clear rejection of $\mathcal{H}0$ in favor of $\mathcal{H}1$ hypothesis ($t = 8.5181$, $df = 4540.6$, $p < 2.2 * 10^{-16}$). This means, that in the dataset at hand, women are also significantly more successful than men, which, again, supports the results

| | 25% | 50% | 75% | 100% |
|---|---|---|---|---|
| Successful | 1,500.0 | 4,634.0 | 10,018.0 | 2,000,000.0 |
| Failed | 2,931.5 | 8,000.0 | 25,000.0 | 101,605,526.0 |
| Total | 2,225.0 | 6,425.0 | 20,000.0 | 101,605,526.0 |

in USD

Table 3.5: Goal quartiles

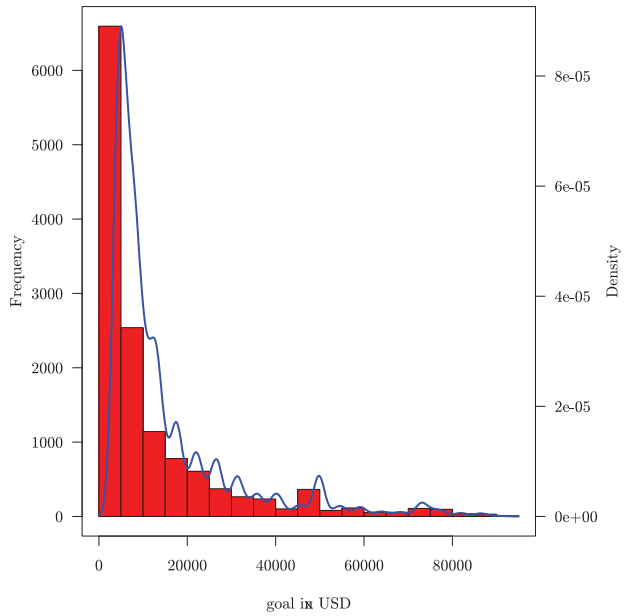



Figure 3.8: Goal histogram and density function

presented by Greenberg and Mollick.

Despite the fact that women are more likely to be successful, men seem to perform better than women according to the means of collected money of 22836.99 for men and 18763.77 for women. Although one cannot claim this with statistical significance since testing the amount of collected money from men compared to women with equality suggestion $\mathcal{H}0$, it cannot be rejected in favor of $\mathcal{H}1$, which hypothesizes men perform better. ($t = 0.65018$, $df = 2142.3$, $p = 0.2578$). A similar phenomenon can be observed in the differences of goal settings between men and women. The average goals for men are slightly higher than the average goals for women, but once more no statistical significance is present ($t = 0.38454$, $df = 2902$, $p = 0.3503$).

As already mentioned, an additional user type attribute was generated, which is able to distinguish between companies and persons. The project success of companies (these

are essentially usernames that contain a business entity type abbreviation) was therefore compared to the success of persons, with the result, that company projects significantly outperform projects from persons ($t = 7.1244$, $df = 805.5$, $p = 1.161e - 12$). They also outperform female project owners alone ($t = 3.3716$, $df = 1099.5$, $p = 0.0003867$).

Automatically, a third class of user types catches all non-classifiable user names. This class contains unknown first names, multiple names from teams and unidentifiable company names. Interestingly enough, even projects from users within this class outperform persons projects ($t = 7.1244$, $df = 805.5$, $p = 1.16 * 10^{-12}$). A possible explanation to this phenomenon could be, that this class assembles from a high amount of companies, which are already proven to be more successful and from teams, which may also be more trustworthy.

### 3.3.6 Data Quality

Last but not least, according to the CRISP DM framework, the data exploration needs to assess data quality, containing the assessment of inconsistencies, outliers and noise. Since the assessment of such phenomena is tightly connected to the preprocessing steps, the major part will be discussed in the section right below in more detail.

First of all, missing values are unavoidable. However, within the CrowdData set they have different origins. For example the content of projects may be inaccessible, possibly due to injunctions caused by lawsuits, or simply because of current request timeouts. In such a case, all dependent attributes like reward count, readability metrics and text metrics in general are set missing. Another reason for missing readability metrics comes from the usage of different languages. All metrics used were learned based on English texts. If a language other than English is detected, these measures must be set missing. It can be assumed, that these missing values are missing-completely-at-random.

Missing values also occur in the Facebook friends attribute, because not all users present their social network size on the campaign page. Mollick [Mol14] provides some evidence that campaign owners with a big social network are more likely to present it. Users with small networks, on the other hand, are less likely to be successful and consequently they exclude the Facebook friends from presentation. This would be a classic case for a missing-not-at-random attribute. To recall Schafer and Grahams [Sch+02] missing-not-at-random means that the "missingness" of an attribute depends on the attribute itself. It is though important to note, that an attribute is not missing-not-at-random, if there exist such self-dependencies, but the values can be derived from other attributes with no residuals. Assuming that the likes a campaign generates can reveal the number of Facebook friends (among other attributes), would transform the missing-not-at-random attribute to a missing-at-random one. However Schafer et al. [Sch+02] argument that "in most cases we should expect departures from MAR, but whether these departures are serious enough to cause the performance of MAR-based methods to be seriously degraded is another issue entirely" and they further state, that those erroneous assumptions may often only have a minor impact on estimates and standard errors. This means that even

if it would not be possible to derive the Facebook friends attribute entirely from other values, it would only have minor impact and it is possibly sufficient to use a simple MAR missing values handling method.

Another phenomenon observed is a slight dataset imbalance, since there are 66.9% failed projects and only 33.1% successful ones. But, as we have seen in the theoretical foundations section (2.2.1), these imbalances are only relevant if the distribution of classes do not match their relevance and importance. In this thesis, successful projects shall not be weighted as more important than unsuccessful ones. Ranking successful projects superior by means of dataset imbalance reduction methods would result in a higher false positive rate for them. This could cause the project owner to reduce his effort prematurely. It therefore reduces his success chances and bears high risks. Depending on the application, a ranking in favor of failed projects could be better instead.

## 3.4 Preprocessing

Even though some preprocessing tasks were already included in the data collection process, the majority happens afterwards and is implemented in Python. This decision is justified by good text analysis libraries and with "pandas", an intuitive data frame handling library. As a first step, derived attributes are generated. This includes a regions attribute, containing the values "Africa","America", "Asia", "Europe" and "Oceania", which are assigned to the data instances according to the country code. Essentially, the regions attribute represent the continent, except, that North and South America are combined to one region "America". The sub regions attribute is a bit more fine granular and splits Europe for example in Northern, Eastern, Southern and Western Europe. Closely related to the locations attributes is the currency problem concerning the goal. In the collected data the goals are presented in the currency of its country of origin. In order to make the goal comparable, a new attribute "goalUsd" needs to be introduced which rates the collected goals by the exchange rate at time of launch.

Several content quality signals are generated on basis of the collected HTML project descriptions. Therefore the html markup was cleaned[5] and a variety content metrics were calculated by the Python textstat library with help of the NLTK (natural language toolkit) library. The Flesch-Kincaid readability score – as first readability metric used – was developed to test the readability and complexity of technical material for naval military[Kin+75]. In the context of crowdfunding this could especially be handy with projects from the technology category. Secondly the Gunning-Fog score was calculated. Both are widely used readability metrics [Si+01]. Furthermore the Coleman-Liau readability index was also included in the metrics. Using more readability metrics could be beneficial, since they all stem from different scientific areas and were trained with different kinds of text. Therefore, this could deliver a broader view on readability in

[5]First HTML parsing was done with the library BeautifulSoup. Subsequently the sentences were cleaned from unnecessary punctuation by tokenizing sentences and words, until the pure text remains.

general within the context of crowdfunding. Together with the readability measures, attributes like picture count, paragraph count, number of sentences, words count, out of dictionary words and used languages are appended to the analysis model. For the sake of completeness, the languages are detected by the "langdetect" Python library. By splitting the text in several parts and performing a language detection on each of them, the usage of multiple languages could be determined.

As a second essential part of the preprocessing step, outliers are detected. As discussed in the theoretic foundations, an outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism [Haw80]. Outliers were detected by reason in the first place. This means that obviously wrong values were removed immediately. As an illustration, the Gunning-Fog and the Coleman-Liau grade level indicate the school grade necessary to read and understand the text. Since there exist no grade level below 0, it is only reasonable to accept levels between 0 and 20 – which would be levels slightly exceeding even the post-secondary level.

Using an interquartile range of three times the standard deviation leads, according to figure 3.9, to a high number of outliers. These attributes viewed separately obviously entail a high level of skewness. It is obvious, that some of the attributes are not normally distributed. An example is the distribution for project duration, where Kickstarter sets 30 days as default value which is used for most of them, but still a great amount of users select the maximum of 60 days for their campaign. With rewards the thing gets trickier. Here, possible reasons could be the following: there could be multiple different rewards at the same level, early bird promotions, etc. Another rather complicated example is the out-of-dictionary attribute. As discussed earlier, this measure does not only contain spelling mistakes but also proper names. Since there are multiple aspects included within this measure, a normal distribution is also less likely.

Due to the skewed distributions a big number of outliers arise, when the attributes are treated separately. Therefore the local-outlier-factors (LOF) are generated for all data point as a density-based, multivariate outlier indicator. For determining the LOFs, the heuristic described earlier was chosen, which uses a $MinPts$ range between 2 and 35 and selects the maximum LOF for each data instance. To recap, these $MinPts$ indicate the number of neighbors for each data instance, which are taken into account. After the calculation of LOFs, it was assumed that there exist 5% or outliers and consequently the top 5% data instances with the highest LOF were removed.

The preprocessing step incorporates three methods for missing values handling. Value, median and model imputation. Case deletion was excluded, since the dataset to analyze was not that big in size with about 14,000 campaigns. Value imputation was applied only for video duration and bitrate when no video was present. In such case, a value of 0 for both was imputed. Model imputation was done for missing Facebook friends, where a linear regression model was trained to predict the values based on the gathered likes.

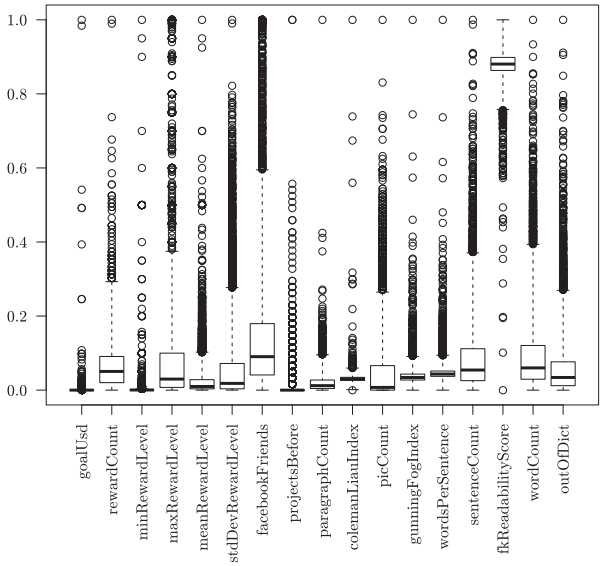



Figure 3.9: The interquartile ranges for several normalized attributes within 3 times the standard deviation

Median imputation is also very straight forward and is by standard applied to all other missing values.

A project‘s development over time is a set of $n + 1$ observations with $n$ online days. To compare projects with different duration, they need to be brought to the same length and same scale. Therefore the $n + 1$ observations are interpolated to an amount of $k$ points, with $O_i$ as the set of interpolated observations of campaign $i$, $k_i = |O_i|$ and $\forall k_i, k_j : k_i = k_j$. A value of $k_i = 100$ was selected. Additionally, the observations do not take place after equal time intervals for each project, but all at once. Consequently, the projects observations are rated by their time-stamps and launch date. It is important to mention, that for project pattern cluster analysis only projects that collected two times of their goal at maximum were taken into account, in order to receive more condensed clusters. Nominal attributes are encoded in multiple binary attributes.

In conclusion, a list of 39 attributes was taken into account with a cleansed dataset of 13554 instances. This is the foundation for the upcoming analysis. For the sake of completeness, the exhaustive list of static attributes is presented in appendix A.1 and includes, next to the attributes name, also the data type and a short description. The list of dynamic attributes is much shorter: pledged amount, number of backers, Facebook likes, shares and comments and Twitter tweets.

## 3.5 Summary

In conclusion, the thesis focuses only on Kickstarter only, which is a reward-based crowdfunding platform. Fortunately, Kickstarter provides a JSON API which permits the retrieval of structured data. The collection algorithm takes care of both static and dynamic data and also taps also external data sources like Facebook and Twitter. The collected data is then stored in a MySQL database in a snowflake-star scheme.

Additionally, the initial data discovery is presented in this chapter. To be more precise, several exploratory statistics for categories, content quality, rewarding, goals and genders in crowdfunding are introduced. The correlation tables show, that projects in music, film & video, as well as games, are more likely to be successful than fashion or technology projects. These statistics also reveal, that for different categories, different content factors are more important. For example in design projects pictures are more important than for journalism projects. Additionally, user aspects were examined, revealing that women are significantly more successful than men, and companies significantly outperform single persons.

Based on the exploratory statistics a set of features was identified, resulting in 39 static features and 5 dynamic features that were taken into account in the various analysis approaches. Finally, the data was integrated into a single dataset and, following a cleansing process, it was ready to go through the actual crowd data analysis.

CHAPTER 4

# Crowd Data Analysis

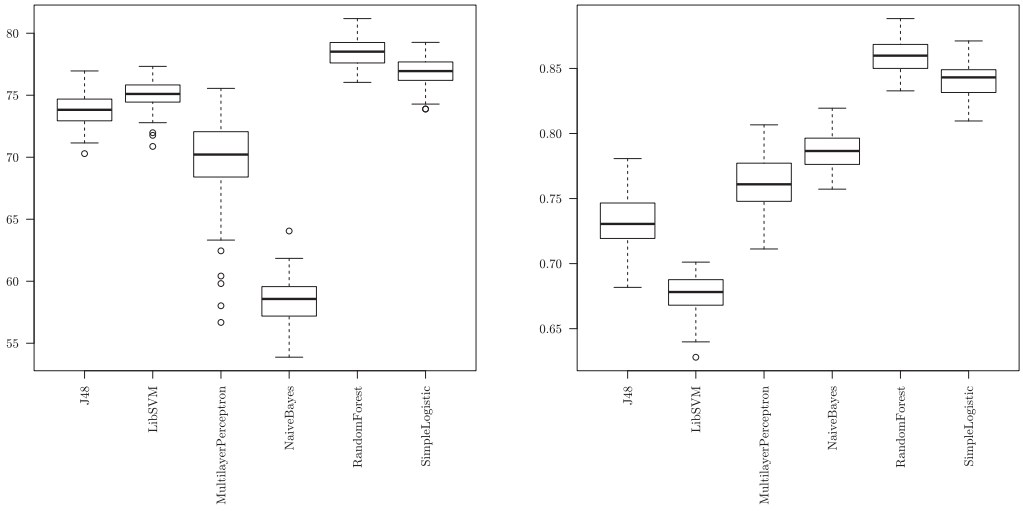
With the following analysis tasks it is attempted to produce answers to the initially defined research questions. Let us therefore recap the objectives of this thesis: First of all, static success factors of crowdfunding campaigns must be analyzed. They cover a range of attributes, which can be derived even before a project has stared. Secondly, the timely development of the campaigns shall be taken into account and implications on success and predictability are investigated. This includes the questions in regard to projects develop over time and what a dynamic forecasting model could look like.

## 4.1 Static Success Factors

As described in the problem statement, it is hard for project owners to know which factors are important for a successful crowdfunding campaign. Especially if one has never done any crowdfunding projects before. This section builds and evaluates basic classifiers that open the possibility to make early predictions on a projects success, in order to help founders in optimizing their campaign. It also compares the learned classifiers to state of the art models.

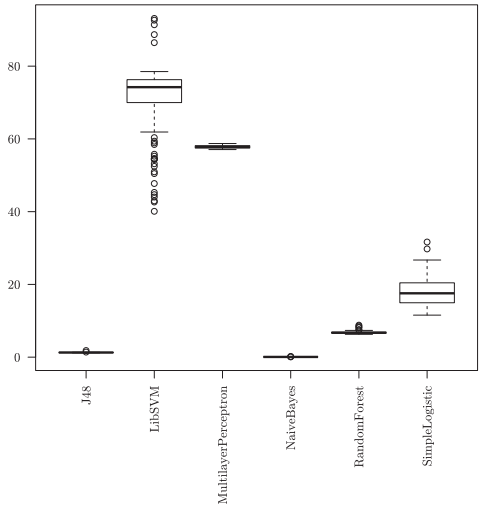
First of all, the technique selection includes an examination of several supervised learning algorithms. For all of them it is true, that the state, which is either "Successful" with the integer representation of 1, or "Failed" with 0 respectively, needs to be predicted. The selected algorithms have already been discussed in section 2 and include Naive Bayes, Support Vector Machines, Artificial Neural Networks, Logit Regression, C4.5 Decision Trees and Random Forests.

In order to find the best performing classifier, an experiment with said algorithms was conducted. These experiments were performed with 10-fold cross validation. Figure 4.1




(a) Accuracy

(b) Area under ROC




(c) Training time

Figure 4.1: Performance metrics of different algorithms compared

depicts the differences in accuracy, area under ROC and runtime (the complete model setup can be observed in appendix A.2.1). The random forest algorithm significantly outperforms ($p < 0.01$) all other algorithms in matter of accuracy and area under ROC. Of course, these learners have room for improvement and through extensive parameter optimization, better results could be derived. Nonetheless this analysis shall give a first impression and emphasizes that random forests deliver a very good accuracy within a reasonable amount of time. While the Naive Bayes classifiers have a severe lack in accuracy, the Perceptron and SVM tend to have tremendous time issues for this dataset.

In figure 4.1a we can observe a very high variance for the Multilayer Perceptrons, which possibly originates from the reduced learning iterations due to the otherwise unbearable runtime. It is therefore expected, that the accuracy tends more towards the upper bound than towards the lower. In general, the performance of random forests leads to a mean accuracy of 78.5% with a Receiver Operating Characteristic of 0.86. The kappa statistic, which compares the accuracy to the expected accuracy, results in a value of 0.49, which is a moderate value according to Landis and Koch [Lan+77]. Because of the superior accuracy and area under ROC in combination with an acceptable runtime, the thesis sets its focus on random forests.

The static success model incorporates several attributes from the work of Greenberg et al. [Gre+13]. To recall, they examined a total of 13 static attributes and tried to predict the success or failure before the projects have even started. According to them, their future goal is – which partly intersects with the goals of this thesis – to give the project owners tools to make their projects more successful. They were able to generate an accuracy of 68% for a balanced dataset of Kickstarter projects. Comparing the random tree results to a simple Greenberg baseline of 68% we have a boost of accuracy of over 10%. These improvements are wonderful; at least they seem to be.

There is only one problem, that such baseline comparisons are extremely dangerous. Nevertheless, they are very common in practice. Even in this case, where the population of crowdfunding projects is more or less the same for both approaches, one needs to be aware of possible pitfalls. For example when considering the attribute “facebookFriends”, there could be differences in their time of collection. The question is, whether they were collected at the beginning of the campaign or on some date when project was already finished. We simply have no information about that. So even with seemingly the same database, one has to be careful with baseline comparisons. According to Armstrong et al. [Arm+09] it would be necessary to have one generally accepted CrowdData set on which all analyses are performed, in order to be able to make such comparisons. This is clearly utopic, since a core task is always finding new, interesting and descriptive attributes that raise the model‘s performance. Also Kickstarter and the crowdfunding projects are continuously changing and thus, the usage of a possibly older comparable dataset would not be accurate. Therefore, the better solution is to rebuild the experiment on the actual data in order to derive meaningful comparisons. This is exactly the approach taken in this thesis.

Unfortunately, the experiment from Greenberg could not be rebuilt 1:1. For one, the Kickstarter project pages share too little information on Twitter accounts and the source of this attribute in the Greenberg experiment can only be guessed. Secondly, the issue with the collection date of certain attributes remains and also no information on data cleansing was presented. Therefore, the Greenberg experiment will be interpreted as a feature subset selection of the basic data (without outlier reduction and with no missing value imputation since none of these methods were described in the

model). Additionally, the Twitter URL, as well as the number of Twitter followers was omitted in the replicated experiment, because of the ambiguity of how those were collected.

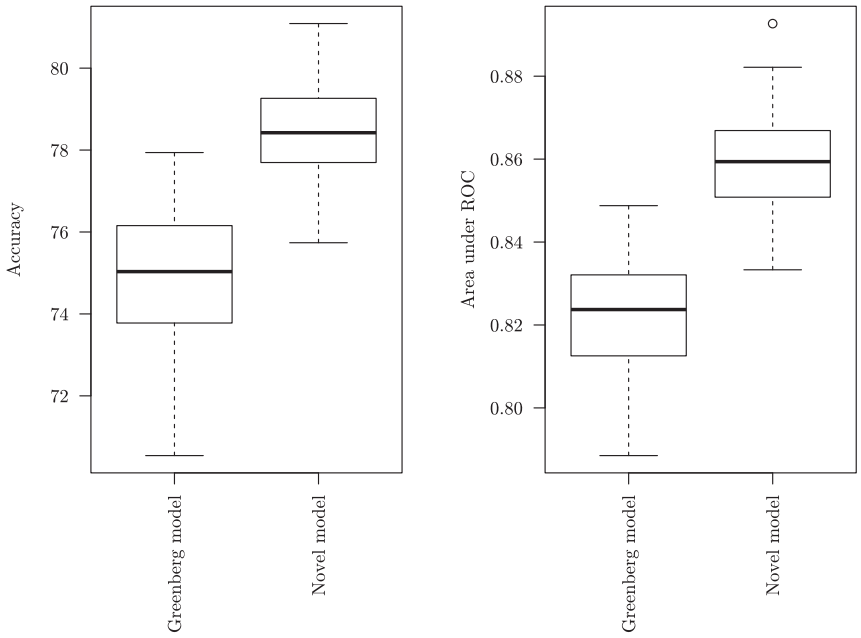



Figure 4.2: Comparison of accuracy and AUC of the replicated Greenberg experiment with the novel static model

The repeated Greenberg experiment, in comparison to the new model from the thesis, is illustrated in figure 4.2. For the purpose of the experiment, a random forest algorithm was used with 100 trees and each split considering 5 features. The replicated experiment resulted in an average accuracy of 74.9%, and an area under ROC of 0.82. This still means a significant improvement ($p < 0.01$) for the new static success model introduced previously, but it is not that much of a leap. Intuitively, two possible explanations for the improvement of accuracy between Greenberg original and replica pop up. First, the Twitter data introduced a high amount of noise into the model or the model performs better on the newer dataset.

**Hypothesis 1** *Noisy twitter data results to a lower accuracy at the initial model*

Claiming the improvement of the reevaluation is caused by noisy Twitter data can be rejected quite easily, because most of the algorithms used by Greenberg are very robust against noisy or random attributes. The used decision tree algorithm, for example, greedily chooses attributes which provide the highest information gain. Clearly a randomly generated – and therefore noisy – Twitter attribute has very little information gain and

therefore will be selected – if ever – very late in the learning algorithm and therefore has only minor impact, or no impact at all. Consequently, we can reject the hypothesis.

**Hypothesis 2** *The model performs better on the newer dataset.*

This hypothesis can be neither verified nor rejected, due to the non-accessibility of the original data set. Additionally, it cannot be ruled out that there are other reasons for the significant differences between the replicated Greenberg model and the original one. A further examination of the reasons is not part of the objectives of the thesis.

With the derived knowledge, the first research question can be partly answered already. The described model can predict the projects‘ success with an accuracy of 78.5% even before the project starts. The second part of the first research question is about finding the most influencing factors of crowdfunding success. The correlations examined earlier in chapter 3 generated first insights into this matter and indicated the number of rewards, pictures and other content metrics as the major success drivers. These insights shall be evaluated with the data mining model at hand. Therefore, a sensitivity analysis for data mining algorithms from Cortez [Cor+11] was used, which uses the produced data mining models as black boxes. The one dimensional sensitivity analysis produces the relative importance of each attribute. It uses a baseline vector with the median value of each numeric attribute and the mode value for nominal ones for the analysis. This baseline vector is altered for each attribute, such that the attribute takes several previously defined levels. Cortez et al [Cor+11] suggest that each attribute takes $L$ input levels, ranging from the minimum observed value to the maximum one. This approach was altered a bit and instead of taking the minimum and maximum values, the 5 and 95 percentile were utilized in order to keep the ranges more condensed. Consequently, the success was predicted with these $L$ alterations of the baseline vector. Quite intuitively, the wider the range of the resulting success probabilities is, the higher is the influence of the attribute in examination. The variance of the predicted probabilities was used as the sensitivity metric, which describes the operation range of outputs. The output variance of one altering attribute was then compared to the output variance of all others, in order to derive the relative importance. For more details to the sensitivity analysis in data mining, I refer to [Cor+11].

The top ten important factors on data mining are depicted in figure 4.3. The bar-plot shows quite similar results compared to the correlations. For example, reward count is top ranked in the correlation statistics and the sensitivity analysis. Some of the important content metrics identified in the correlations can be found in the relative importance statistic as well. Those are pictures count, video, out of dictionary and sentiment. Like in the correlations, here the pictures count is assumed to be more important than the presence of a video. Kickstarter on the other hand highlights the importance of a launch video in their “Getting started” FAQ [1], but as we have just seen, the presence of lot of

[1] https://www.kickstarter.com/help/faq/creator+questions

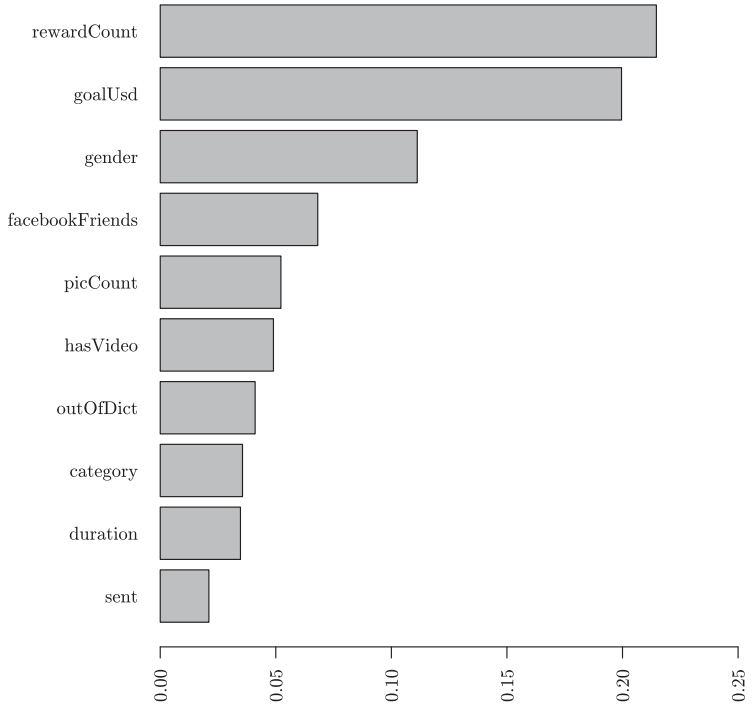



Figure 4.3: Top ten influencing factors according to the one dimensional relative importance

pictures is at least as important. This fact is probably not recognized accordingly by Kickstarter.

While the correlations indicated only a slight negative impact of the goal setting on the project‘s success, the sensitivity analysis sees the goal as nearly as important as the reward count and placed it on the second rank. This is actually not highly unexpected, but the interesting thing is that correlation statistics did not capture the importance of this attribute as much. A reason could be found in the high variance within the attribute itself, which reduces the correlation coefficient.

Last but not least, the relationships of multiple attributes to the success shall be examined. This can be done with the previously mentioned baseline vector, where – this time – more than one attribute is altered. The resulting success probabilities were then visualized on a surface plot. Figures 4.4, 4.5 and 4.7 illustrate surface and level-plots of several highly important attributes plotted against the success probability. First, figure 4.4 shows the goal, the reward count and their relationship to the project‘s success. After a certain threshold of project goals is exceeded, the success remains rather constant.

This threshold is somewhere beneath a value of $10.000. On the other hand, the project's success is influenced by the setting of the number of rewards, regardless of the goal size. However, the number of rewards also reach a plateau of a maximum success on about 10 rewards. The diagram also depicts the quite intuitive relationship, that for smaller goals the number of rewards can be smaller as well.

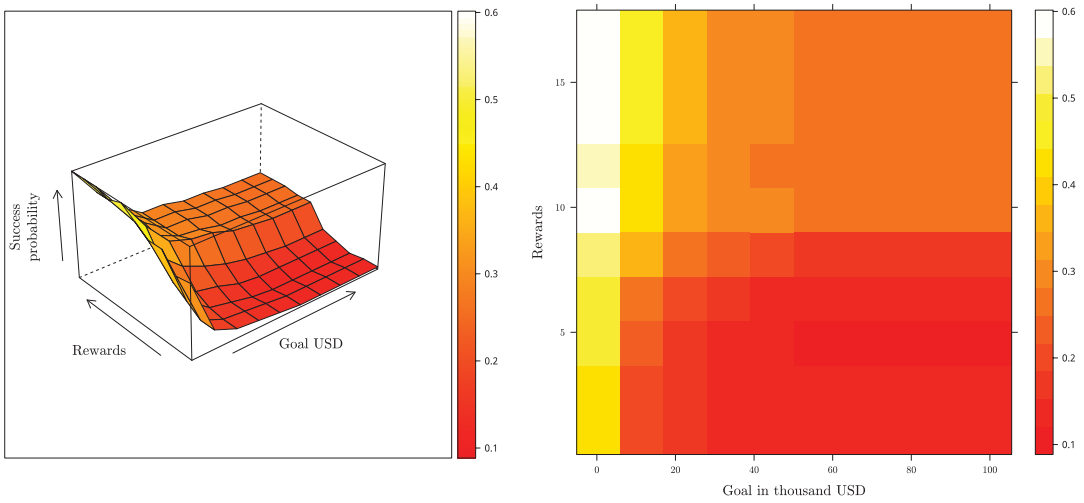



Figure 4.4: VEC surface level-plot of goals and rewards

The second surface plot in figure 4.5 connects the number of rewards with Facebook friends. It identifies a maximum success probability plateau, which (again) starts on about 10 rewards – this seems to be the optimal selection for rewards – and a number of about 800 Facebook friends. Additionally, there is a slight decrease occurring in success probability when hitting a number of more than 15 rewards. This indicates that a mere stubborn adding of more and more rewards is counterproductive. A certain level of rewards shall be provided, but the founder should take care not to overwhelm the user with the rewards.

The third pair of surface- and level-plots in figure 4.7 was picked because of the peculiar form. Obviously, there are two local optima for sentences. One on about 29 sentences and the other on 38 sentences. This local optimum gets clearer the more pictures are included in the description. In other words, when a lot of pictures are used, it is sufficient to have a reduced number of sentences. Additionally, after leaving a level of zero pictures behind, the success grows quite steadily with the number of pictures.

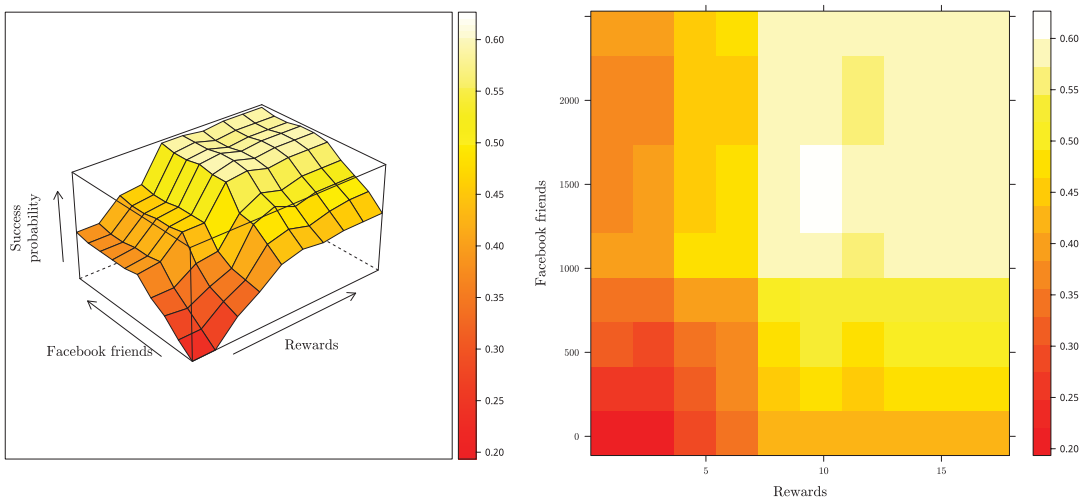



Figure 4.5: VEC surface level-plot of rewards and Facebook friends

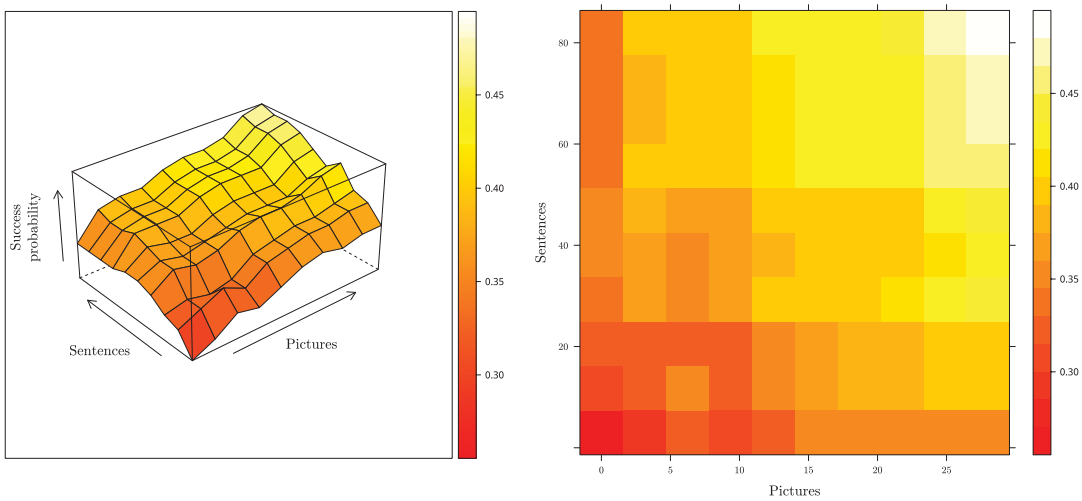



Figure 4.6: VEC surface level-plot of pictures and sentences

## 4.2 Project Development Patterns

The first partial goal of research question two is to identify project development patterns. This opens up the possibility for platform providers to identify important turning points and apply special marketing actions accordingly. Additionally, project founders could initially develop conditional promotional actions. This means a project owner can create multiple marketing strategies for different development scenarios and apply the right one when the actual evolving development path is known for sure.

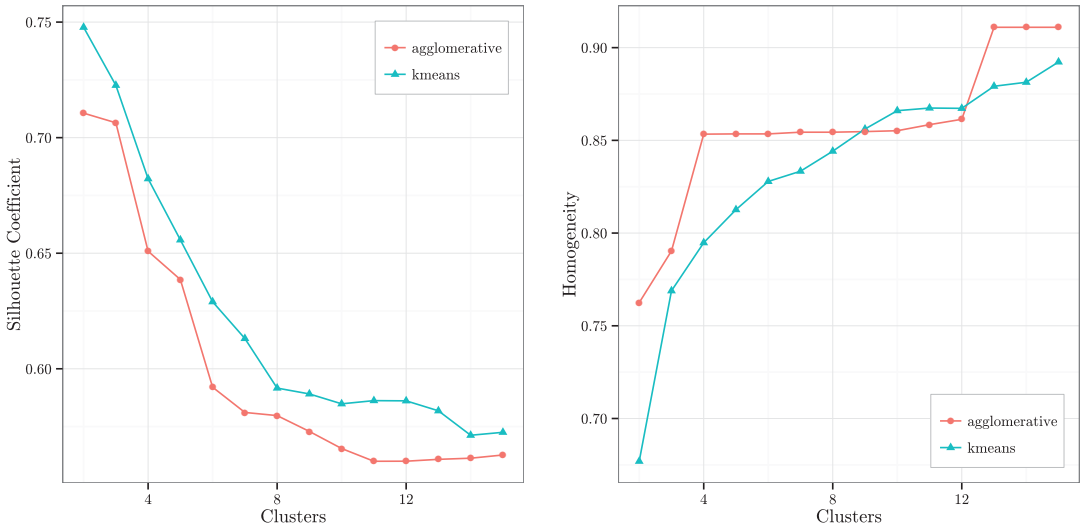



Figure 4.7: Comparison of Kmeans and Agglomerative clustering methods

The project patterns are identified by clustering the observed development patterns for the projects in the dataset. Kmeans and Agglomerative clustering algorithms are examined. Figure 4.7 compares the silhouette coefficients and the homogeneity with success or failure labels of the two different clustering techniques. Both algorithms are chosen from Python's scikit-learn machine learning library. The silhouette index measures how well a data point fits into the cluster to which it is assigned, by comparing the within-cluster cohesion, based on the distance to all entities in the same cluster, to the cluster separation [Amo+15]. According to the silhouette index, the clusters are naturally not quite separable since the coefficient decreases steadily. Another statistic used was the homogeneity, which means that "the class distribution within each cluster should be skewed to a single class, that is, zero entropy" [Ros+07]. Since the homogeneity measure needs class labels, the success and failed labels are used. The right diagram in figure 4.7 depicts the rising homogeneity with the increasing number of clusters. The hierarchical approach produces more homogenous clusters by means of success labels. For the further examination a cluster size of 6 was picked as a reasonable trade-off between silhouette index, homogeneity and explanatory power. Explanatory power means, that patterns shall be identified that actually make sense. Additionally, the Kmeans clustering

algorithm was selected due to the better performance on the silhouette index. The silhouette index in this application is more important than the homogeneity, due to the final state class labels used for the homogeneity. These class labels mark a less-than-ideal solution, since the purpose of the clustering is not the identification of success and failure clusters but the clustering of development patterns. The usage of these final class labels is, however, not completely wrong, since the clusters should separate good performing projects and their patterns from bad performing.

The selection of six clusters results in figure 4.8. While the thick line represents a cluster centroid, the shaded region marks the standard deviation of the clustered project. As described in section 3.4, only projects that reached a maximum of two times of their were taken into account. The green cluster at the bottom contains the majority of all projects. These projects fail by a big amount, where most of them not even reach 10% of the goal, with a gradient of nearly 0. The red cluster already contains better performing projects, but still the majority does not reach 50% of the goal. The most interesting clusters are the clusters 1, 4 and 5, which hold projects that at least nearly reach the goal. The big difference lies in the way the goal is reached. The purple projects have a rather weak start but perform best in the late campaign. The yellow projects show a nearly linear project development and the blue ones best perform in the early campaign. All high-performance projects are then incorporated in the turquois cluster. In general, all good performing projects have a strong start and collect a big number of pledges in the first moments of the campaign.

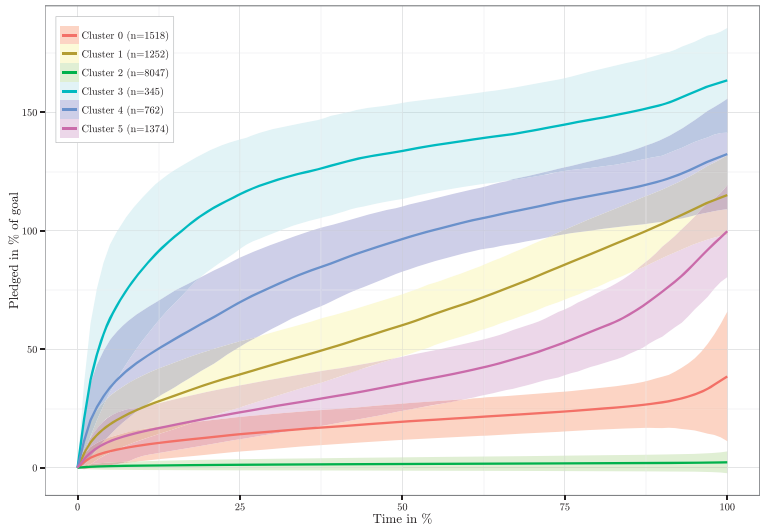



Figure 4.8: Kmeans clusters with a ribbon of one times the standard deviation

The figure 4.9 depicts the evolving cluster homogeneity, when taking all known pledge-values of a project at a specific point of time into account. Intuitively, the plot

shows how safe the predictions are at a specific point of time. Within the first several percent of the project development time, the homogeneity takes a rapid development, until after about ten percent, the homogeneity growth slows down to a linear shape. This means that at first, the identified clusters are rather heterogeneous and cannot be distinguished clearly, which can also be observed in figure 4.8, where within the first several percent the clusters highly overlap.

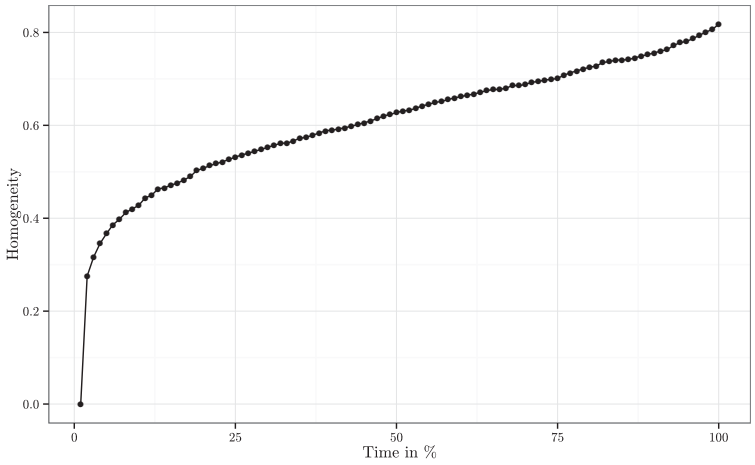



Figure 4.9: Homogeneity over time

With these findings the first part of research question two can be answered, namely the question of "How do projects develop over time?". To sum up, successful Kickstarter campaigns exhibit a steep increase of pledged money in the first moments of the campaign. In contrary, not successful projects have a more gentle increase in promised money. In the mid period of a campaign the project developments diverge a lot. There are projects who live only on the early success, while other projects have a lower increase at first and take up speed right before the end. Using this knowledge should also help to predict the project's success early on, since projects with different final states differ a lot in their development.

## 4.3 Dynamic Success Prediction

Until now, the observed algorithms are able to predict nearly 8 out of 10 project end states correctly. Even with extensive parameter optimization the boundaries of 78.5% are hard to get past. Therefore, new aspects to crowdfunding success prediction shall be examined – the dynamic projects prediction. As mentioned earlier in the related works section 2.3, Etter et al. [Ett+13] used k-nearest-neighbors (kNN) algorithm and Markov chains with learned transition matrices in order to give an early prediction about the project's success. This thesis uses a very similar dataset to Etter et al., but as a

classification algorithm, random forests were chosen.

Markov Chains seem to be a very elegant approach to this matter, but – to slightly anticipate the next section – random forests clearly outperformed all other models for the static approach, which is very handy for the synthesis of the dynamic and static success factor model later on.

No major differences in memory usage should arise between Markov Chains and random forests. Etter et al. needed to keep the transition matrices for each point $i$ out of $n$ points in the memory in order to be able to apply the Markov Chain to a specific input vector[Ett+13]. The decision tree approach on the other hand needs to regularly access $n$ random forests. To simplify the further description, I will refer to this model as the dynamic n-forest approach.

A big advantage of the decision tree application in this matter is that it is very straight forward and intuitive. Assuming, that the project‘s development consists of $n$ observations, where the last observation of $n$ equals the final state. It does not matter if they are interpolated or real observations, only the number of observations for each project needs to be the same. For every time point $i$ with $i = \{1, \dots (n-1)\}$ an own small random forest is learned on the labels “successful” and “failed”, based on all dynamic data, which is present at that specific point of time. To put it more formally, the random forest induction algorithm for time $i$ is performed on the dynamic attributes vector $x_{k,i} = (x_1, x_2, \dots, x_i)$ of the instances $k$. The mapping function for supervised learning at time $i$ is $g_i : X_i \to Y$ over the example set $S = ((x_{1,i}, y_1), \dots, (x_{m,i}, y_m)) \in (X_i \times Y)^m$. The assembled model then simply is: $f(i, X_i) = g_i(X_i)$ where $X_i \subseteq X$.

The dynamic-n-forest incorporates the development of pledges, backers, likes, shares and tweets into one model. While pledges are transformed to a percentage of the goal, all other metrics were included directly. It should be mentioned, that two different attribute representations were used. First, the absolute values for dynamic attributes and secondly, only the relative developments between two observations. Triantaphyllou [Tri+06] states that “If all attributes affect the class attribute independently, the single attribute or first order decision tree has the greatest gain ration”. Consequently, one would think that using the relative attributes would result in better accuracy. In reality, the absolute attribute strategy performs marginally worse in the first time steps but significantly outperforms the relative strategy in the long run. This is the reason why the absolute strategy was selected for further analysis.

Figure 4.10 plots the accuracy that the dynamic-n-forest models produce over time and compares the absolute with the relative model. Both algorithms were trained with random forests. Each contains 50 trees and, depending on the time point, between 2 and 8 randomly selected split features. Training the complete model took about 2m30s. The time scale was, as mentioned in the preprocessing section, interpolated to a number of 100

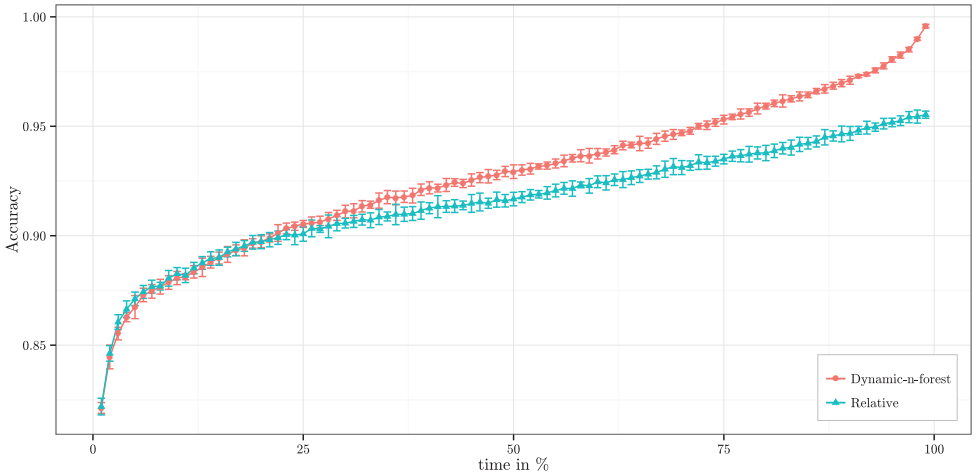



Figure 4.10: Dynamic N-Forest model

observations in order to derive comparable results. The absolute strategy shows that, after 1% of the time an accuracy of about 82.1% could be achieved and after 3% of the live time, which corresponds to 1 day after launch for a 30 day campaign, an accuracy of over 85.5%.

When comparing the project's development from Etter et al. to the dynamic-n-forest model, significant improvements within these very similar methodological approaches can be found (tested for $i = 1, 5, 10, 50$ with $p < 0.01$). Nevertheless, the general character of the curves shown by Etter stays the same. This includes a substantial accuracy gain within the first 10% of the projects live time, then a more or less linear growth in the middle section and a slightly convex curve within the last several time steps. A direct comparison of Etter's findings with the dynamic-n-forest model is in this case much less problematic, since the data sets have only little potential to diverge.

In order to better visualize the results of the dynamic-n-forest model, an even simpler model is introduced, which only uses the pledged amount at time $i$ in order to predict success. This simple approach uses the same random forest settings as before. Figure 4.11 illustrates the evolving success probability at every time $i$ when a certain percentage of the goal was pledged. It shows that the success rate is already very high, if more than 2% of the pledges were acquired within the first moments after the launch. If a project has reached 80% of its goal after 79% of the project time, the success probability is at minimum 90%.

## 4.4 Synthesis Prediction

Right now the dynamic and static aspects of crowdfunding projects were examined separately. In order to further raise the performance, the dynamic and the static approach

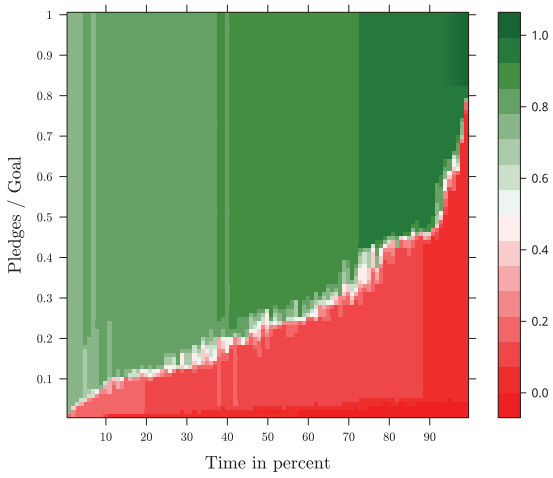


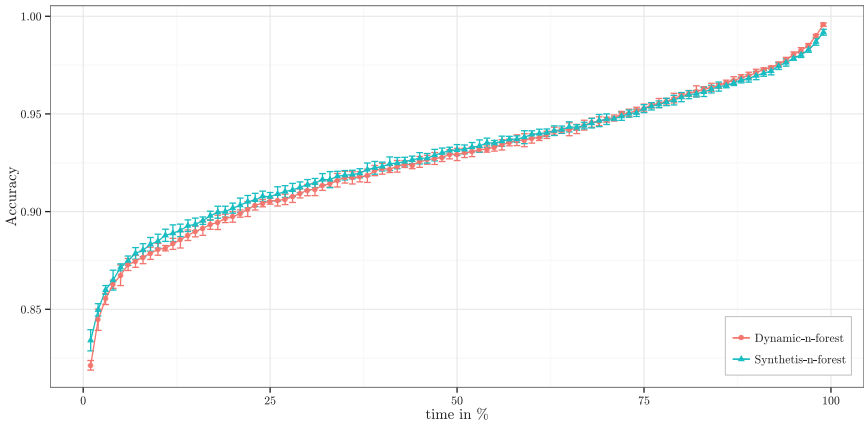

Figure 4.11: Pledges and success probability over time

shall be combined to a synthesis model. As already mentioned this will be done by simply including the static attributes in every random forest model $i$. Quite intuitively, these random forests are trained with the static attributes and the available dynamic attributes at time $i$.

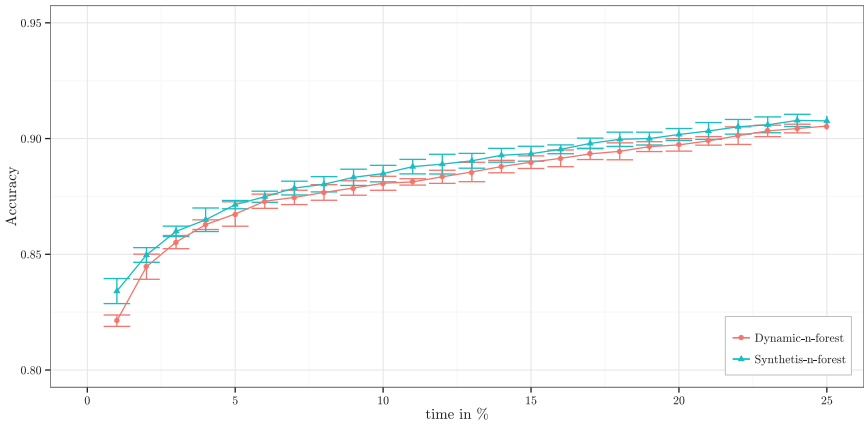
The results of this method, in comparison to the dynamic-n-forest model, are illustrated in figure 4.12. The random forest parameters were kept unchanged. Again, the curves represent the evolving accuracy of each time point with a 95% confidence interval of two times the standard deviation. Until 35% of the time the synthesis model (with 50 random trees, $log_2$(static attributes + dynamic attributes at point$i$) random features, information gain metric) performs significantly ($p < 0.05$) better than the dynamic-only-model. However, at 82% the dynamic model takes overhand and outperforms the synthesis model at each time step ($p < 0.05$). Altogether, synthesis-n-forest model possesses a decent accuracy of 83.4% after one percent of the campaign‘s live-time. That amounts to only 7.2 hours for a 30 days project.

It is worth to mention that in addition to the synthesis model shown above, a stacked model was trained and compared as well. It included the static success factors, predicted by the static model from section 4.1, as a success probability into the dynamic model from above. Unfortunately, this model could not live up to the high expectations and under-performed compared to the previously described synthesis model.

In conclusion, by combining static and dynamic factors into one model a solid project success forecasting model can be found. The synthesis model significantly outperforms the standard dynamic-n-forest model in the early stages of the campaign. The reason




(a) Comparing the results on the complete online time




(b) Early stage comparisons

Figure 4.12: Comparison of accuracy of the dynamic- and the synthesis-n-forest model

for the better performance can be found in the information gain in the early campaign, introduced by the static factors. This could proof very valuable for project founders, because it lets them react early on possible negative developments in the projects evolution. Such increases within the early moments of crowdfunding campaigns are most valuable [Ett+13].

## 4.5 Summary

This chapter starts by presenting methods for predicting the success of crowdfunding projects with supervised learning algorithms. The model is trained on static attributes, which are present before the project has even been launched. Initially, the static analysis compares accuracy and training time for several supervised learning algorithms and

finds random forests with the top accuracy of 78.5% and a reasonable training time as a perfect fit for the data mining problem at hand. The derived random forest model is then compared to the similar model presented by Greenberg et al [Gre+13], finding a significant improvement in accuracy. Secondly a sensitivity analysis is presented, with which made it possible to rank the attributes according to their relative importance. Additionally, the sensitivity analysis provides a way of multi-dimensional attribute analysis and highlights attribute dependencies in vector plots.

The identification of project development patterns in this chapter helps platform providers to find important turning points, so that marketing actions can be planned reasonably. Unsupervised learning methods found a number of six distinctive clusters, each representing a different development pattern, ranging from weak starter projects to early performers.

Last but not least, the chapter covers the dynamic success prediction process. By including dynamically changing attributes like the pledged amount, the number of supporters and social media stats in the model, the so called synthesis-n-forest model evolved. The name emphasizes the combination of static and dynamic attributes with random forests as a classification algorithm. The synthesis-n-forest model could improve the accuracy of the static model to a value of 83.4% after 1% of the project live time. The models created in this chapter serve as a basis for the CrowdData analysis framework, discussed in the next chapter, which will put the created models into a business context.

CHAPTER 5

# CrowdData Framework

The last part of the thesis attempts to sketch an information system, which brings the generated models to life and into action. In the following section, the CrowdData framework is outlined, which lets a project owner or platform provider analyze their projects and produce valuable insights. First, a general structure of such an information system shall be introduced and secondly some real-world Kickstarter projects shall be examined with the framework and should serve as a proof-of-concept.

## 5.1 General Structure

The CrowdData framework should provide a structured way of analyzing projects to the user. Figure 5.1 depicts the idea of the framework within a component diagram. The core component is the global analysis component, which requires a crowdfunding project a data model and some display and analysis settings in order to function properly. The project component contains all information about a project. It wraps static attributes, the dynamic project fact objects, and objects for category, rewards and content. Essentially, it contains the complete star scheme from figure 3.4. Depending on the analysis purpose, the required data mining model is either the static model or the dynamic-n-forest model. The modular build makes it possible to also include future models. The third required interface contains settings for the analysis. These settings‘ major purpose is to let the user actively manage the analysis. For example, the user can order sensitivity analysis for special attributes via this exact interface. Depending on the data mining model, different preprocessing steps need to be taken. Therefore a “Preprocessor” sub-component was introduced within the analysis component, which can be switched out easily. The actual sensitivity analysis and success prediction is performed within the second sub-component “Analysis”. This analysis component runs the input project through the data mining model and the sensitivity analysis. Furthermore it builds an analysis report. This report serves as input for the visualization component at last, which produces human readable

analysis outputs.

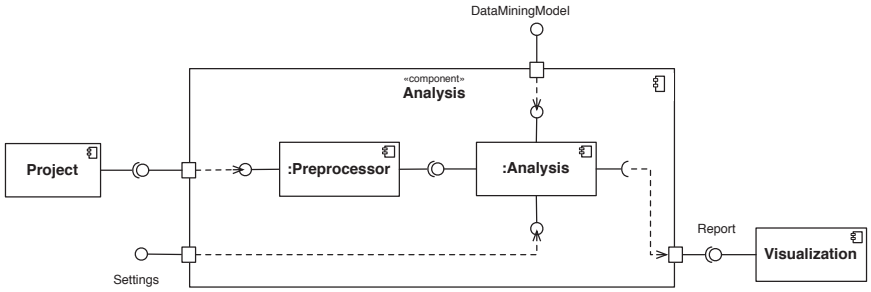



Figure 5.1: CrowdData framework components overview

Two different analysis components are assessed: The static and the dynamic analysis component. First, the static analysis component‘s purpose is to provide the user with a general success probability, information on the relative importance of the different attributes and some user-defined sensitivity analyses. A user-defined sensitivity analysis means that performing a sensitivity analysis on all attributes would be overkill and therefore the user must select the variables he intends to analyze himself. As mentioned before, this could be done via the settings interface at the component. Alternatively, if no input is given, the framework could automatically select the k-top-most important attributes from the relative importance analysis and provide sensitivity analyses for them. Secondly, a dynamic analysis component is introduced in the standard framework, which presents the timely development of the volatile project attributes and the evolving success probability to the user.

## 5.2 Proof-of-Concept

The purpose of this thesis is to provide models for success prediction in the first place. Therefore, the actual implementation of the crowdfunding data analysis framework is omitted, since it would exceed the scope of this work. Instead the idea of such a Crowd-Data framework shall be sketched only and therefore serve as a proof-of-concept. Thus, two projects were chosen, on which the framework is applied. First the outlined static analysis component is tested and subsequently the dynamic analysis component.

### 5.2.1 Static Analysis

As already discussed previously, the static analysis component first reports the relative importance of the projects attributes. It therefore only reads static attributes from the project component. In our example framework the top ten most important attributes are listed. Based on these important attributes a user can conduct a sensitivity analysis.

| Attribute | | Attribute | |
|---|---|---|---|
| goal | 10000 | paragraphCount | 4 |
| currency | USD | wordCount | 124 |
| lang | en | picCount | 4 |
| hasVideo | Yes | sentenceCount | 5 |
| videoDuration | 189 s | fkReadabilityScore | 55.58 |
| duration | 30 d | colemanLiauIndex | 10.74 |
| rewardCount | 11 | gunningFogIndex | 11.6 |
| minRewardLevel | 5 | outOfDict | 7 |
| maxRewardLevel | 5000 | projectsBefore | 0 |
| category | Food | type | PERSON |
| facebookFriends | 50 | gender | MALE |
| sent | pos | state | FAILED |

Table 5.1: An overview of the project, which is used for static analysis

The sensitivity analysis can be done one- or two-dimensionally. In most cases, a one dimensional analysis shall be sufficient, but the two dimensional approach can be especially handy, when the analyst notices manifold success dependencies of a certain attribute.

The selected project is located in the category “Publishing” with a goal of $10,000. A general overview of the project is provided in table 5.1. Project identifiers like names or the URL are intentionally omitted in the description in order to protect the privacy of the project owners. It should be mentioned that the crowdfunding project to examine is unobserved from the learned model.

The algorithm predicts the project as not successful with a success probability of 36%. This is done by processing the project through the static classifier generated in section 4.1. Fortunately, the random forests deliver a class prediction, but also the likelihood of a project belonging to the class. The project owner is then supported by sensitivity analysis, which highlights important attributes in form of a bar plot as in figure 5.2. The one dimensional attribute analysis plots attribute values against the success probability improvements. The results can be observed in figure 5.3. For the scope of the proof-of-concept, it was sufficient to only illustrate some interesting attributes, not the entire top ten. On the other hand, some two-dimensional surface- or level-plots, which let the founder explore the input sensitivities in relation, are depicted in figure 5.4. In fact, the Static CrowdData Framework uses the already discussed sensitivity analysis methods of section 4.1, with the slight difference, that the baseline vector is not generated artificially, but instead the campaign to analyze is taken as such.

According to the CrowdData framework, the project owner derives the following advice and conclusion:

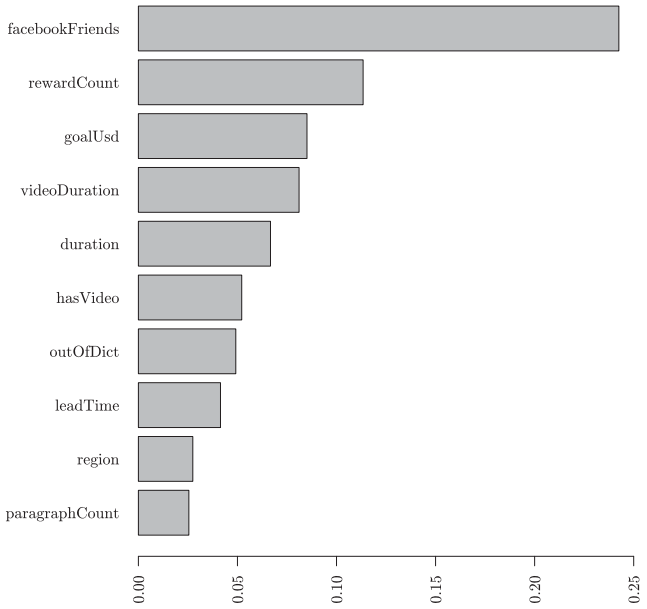



Figure 5.2: Relative importance of the project attributes

- Increase the number of Facebook friends. Two possibilities would be imaginable, to either buy Facebook friends or to connect another team members Facebook account to the project page, who has more Facebook friends. However, it is very likely that buying Facebook friends has no effect, since these "friends" would not participate in the funding process. It could only help the founder by signaling a big social network to possible unknown supporters. Therefore, the latter possibility would be preferable. Depending on the number of Facebook friends, the projects success chance can be increased by over 25%.

- The number of rewards can stay the same. Ten to eleven rewards are considered a good value.

- While with the number of four pictures the campaign has reached a local optimum, the number of sentences shall be increased to a value of 16, in order to achieve a 10% increase in success probability. Also the number of paragraphs is nearly optimal with a value of 4. Therefore we can say, that a higher sentence per paragraph rate is beneficent here.

- The video is too short. If it is possible, the video should be extended to a length of 4 minutes and 10 seconds. This would increase the chance of success by one more percent.

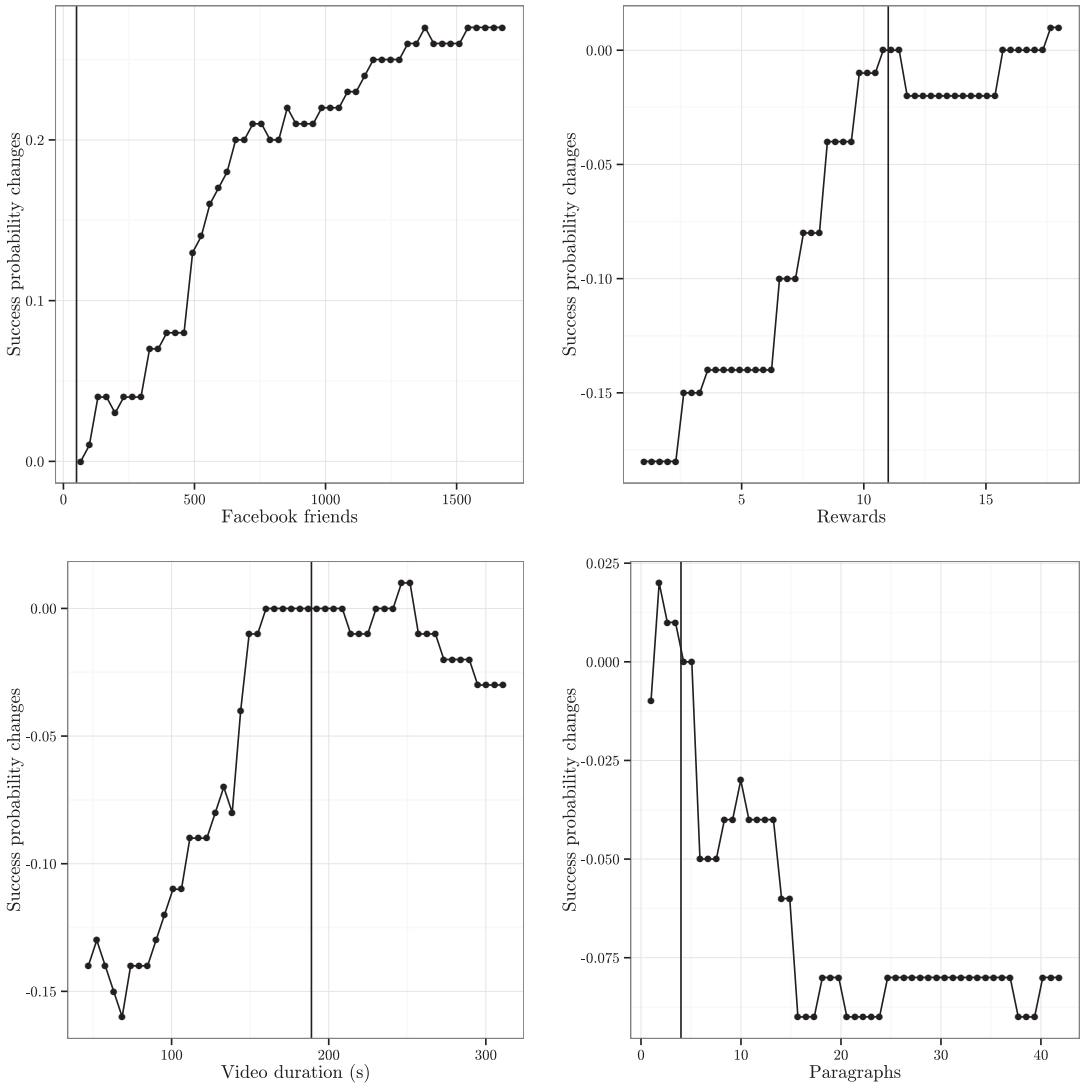



Figure 5.3: One dimensional sensitivity analysis for the attributes Facebook friends, rewards, video duration and paragraphs, where the vertical line mark the respective attribute in the project

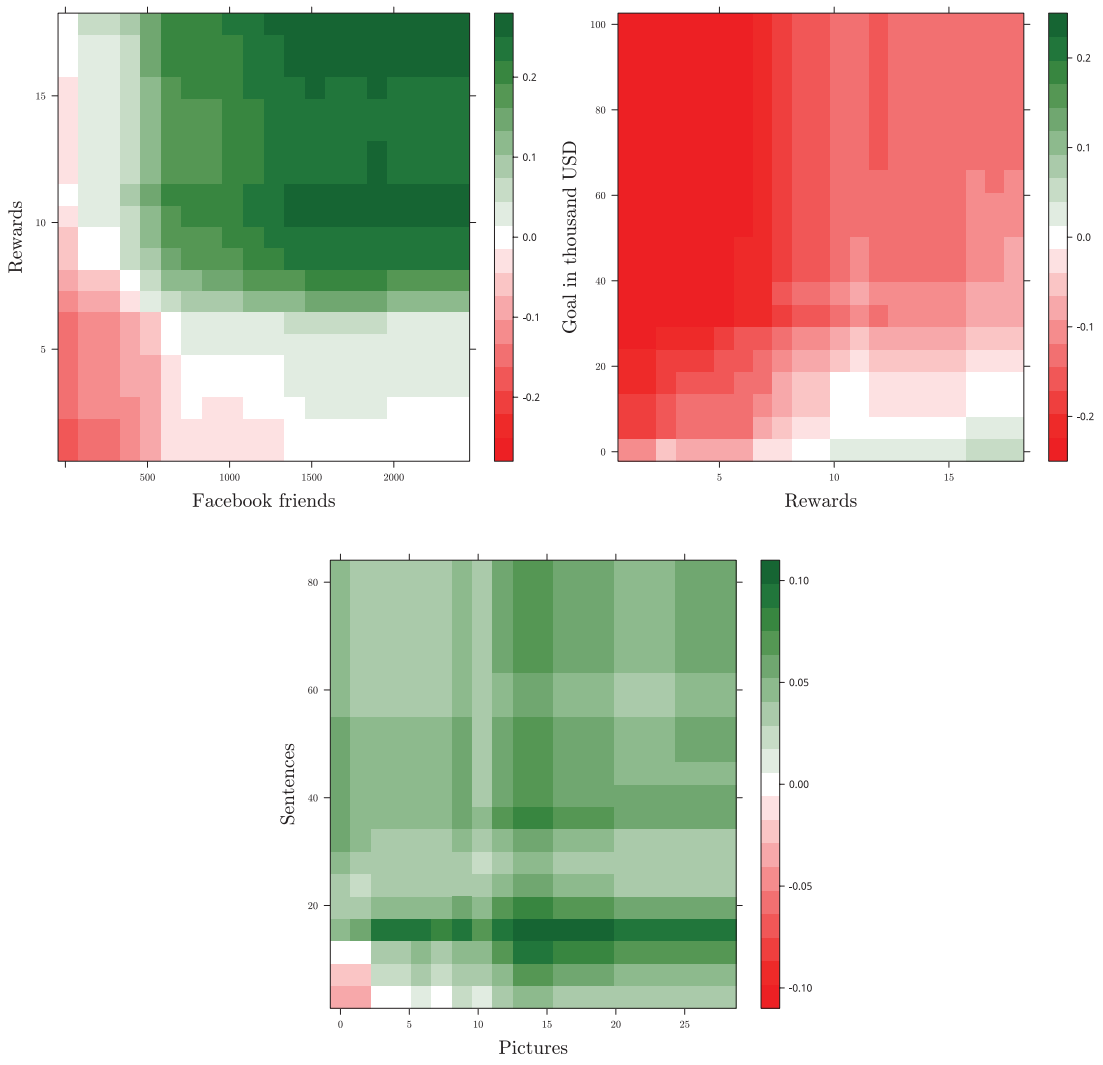



Figure 5.4: Two dimensional sensitivity analysis with the combinations rewards and Facebook friends, rewards and goal, pictures and sentences

| Attribute | | Attribute | |
|---|---|---|---|
| goal | 12000 | paragraphCount | 15 |
| currency | USD | wordCount | 348 |
| lang | en | picCount | 37 |
| hasVideo | Yes | sentenceCount | 20 |
| videoDuration | 142 s | fkReadabilityScore | 62.68 |
| duration | 21 d | colemanLiauIndex | 10.55 |
| rewardCount | 6 | gunningFogIndex | 8.8 |
| minRewardLevel | 1 | outOfDict | 46 |
| maxRewardLevel | 90 | projectsBefore | 1 |
| category | Games | type | PERSON |
| facebookFriends | 744 | gender | MALE |
| sent | pos | state | SUCCESSFUL |

Table 5.2: An overview of the project, which is used for dynamic analysis

### 5.2.2 Dynamic Analysis

The dynamic CrowdData analysis component illustrates the project development at a certain time point. It depicts the development of pledges, backers, social media factors and also the evolving success. This analysis is based on the synthetic-n-forest model. By displaying the changing success probability, the user is able to react early on certain undesirable developments.

For project 2 a dynamic analysis according to the CrowdData Framework is conducted. Again, the project facts are summarized and depicted in table 5.2. This time it is a project from the category "Games" with a goal of $12.000. The static success probability is 60%. The dynamic component produces a report based on the dynamic development and tries to predict the evolving success early on. It is assumed, that this project has just started and the analysis is conducted after 5% and 20% of the time.

### 5.2.3 Success Prediction at 5% of the Time

Five percent of the time equates to one day, 15 hours and 36 minutes from the total 33 days online duration of the project. Provided, that the project development was continuously recorded, the dynamic project outputs the success probability of 85.77% at this time point. Due to the good campaign development in the first moments, the success probability improved over 12% to the initial success probability.

Figure 5.5 depicts the output report diagrams and contains the development of all changing factors: pledges, backers, Facebook likes and shares and Twitter tweets. Additionally, the success probability is plotted over time. In this example, the success probability steadily rises until a break occurs at 3% of the time. At 4% of the time, the project's success chance again rises. The break obviously has something to do with the suddenly reduced growth of the success factors.

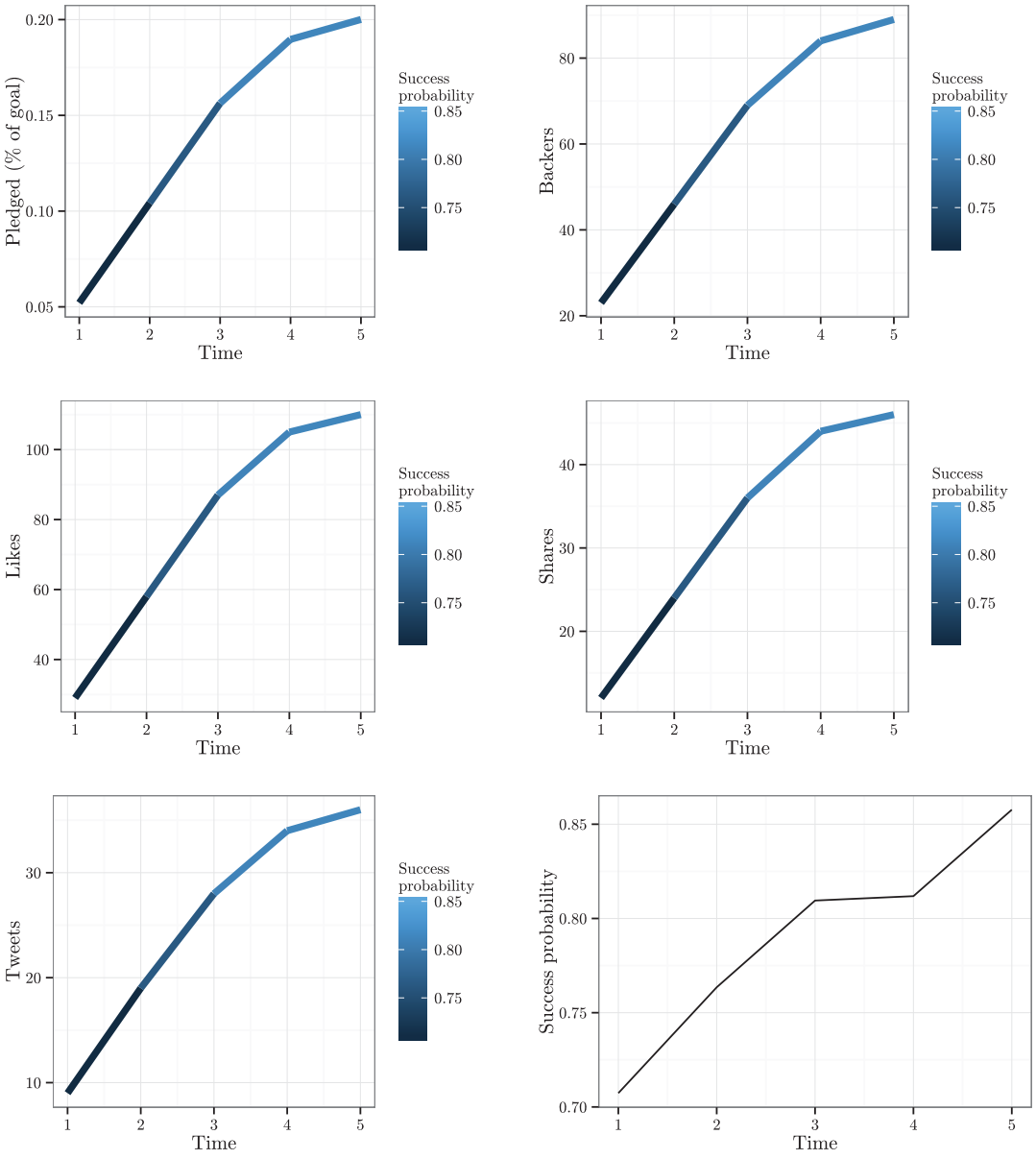



Figure 5.5: The project development at 5% of the project‘s online time

### 5.2.4 Success Prediction at 20% of the Time

After 6 days, 14 hours and 24 minutes the dynamic model predicts the success with a probability of 83.35%. This means a slight decrease of the success probability since the last report. The reasons can be found in figure 5.6. The dynamic model obviously assumed, that, after the first few percent, the project would develop more rapidly and

therefore assumed a higher chance of success at the beginning. In reality, the dynamic project factors took a concave development, where they all converged to a certain value. Therefore the algorithm lowered the expectations. Recently, the Twitter tweets rose again unexpectedly, which once more led the algorithm to raise the success chance.

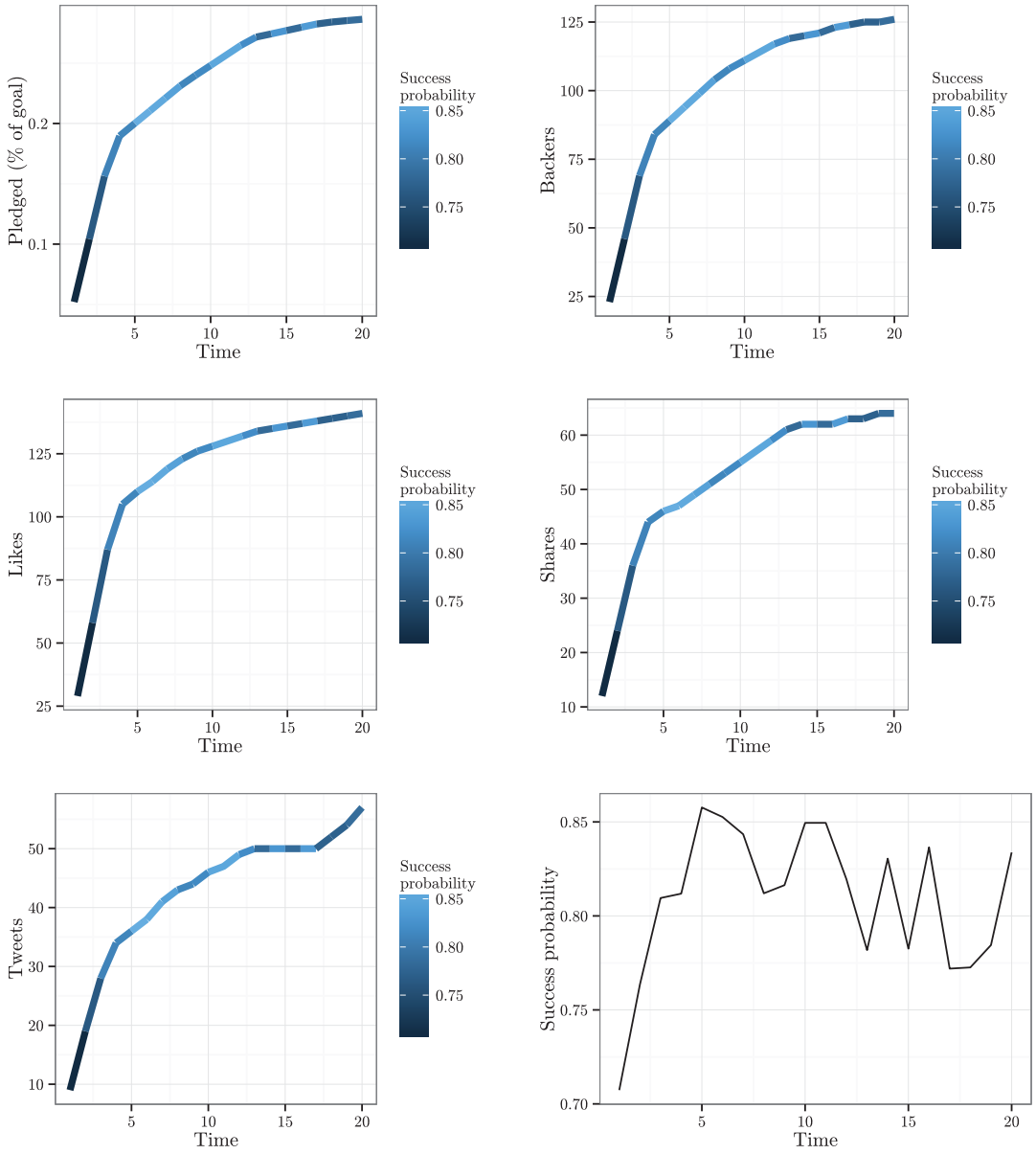



Figure 5.6: The project development at 20% of the project‘s online time

### 5.2.5 Prediction without continuous Recording of the Project

If the user does not have the possibility to continuously record the project state over time, the framework still provides a way for success prediction. For this purposes the previously discussed simple dynamic model can be used, which level-plots the success chance over time and the pledged amount. However, its accuracy is not as good as the synthesis-n-forest model, but it still provides valuable insights and helps the project owner to react on project developments early on. Figure 5.7 again plots the simple model, where the project owner simply sets the parameter‘s time and amount pledged and receives an approximate success probability.

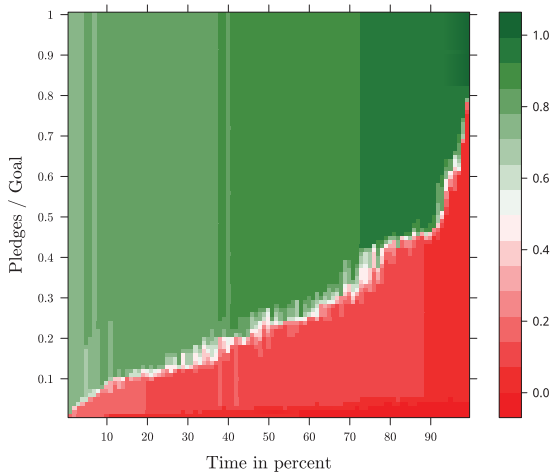



Figure 5.7: Success prediction without continuous project recording

## 5.3 Summary

In summary, this chapter covers the proof-of-concept for a crowdfunding data analysis framework, which, at first, helps a project owner in finding a perfect initial project setup. In order to achieve this, the generated models from the analysis chapter are put into a business context. The framework uses an adapted sensitivity-analysis to provide the user with information about which attributes can be improved in order to optimize the project and consequently maximize the success probability. For the proof-of-concept it was sufficient to apply the static CrowdData analysis framework on a single example project.

In similar fashion, the chapter shows the dynamic part of the CrowdData analysis framework, which enables the user to access a sophisticated, on-going success monitoring. This not only displays the current progress of the dynamic project features, but also

provides a view at the evolving success probability at any given time, so that the user can react immediately upon discovering counterproductive development.

CHAPTER 6

# Conclusion

The overall objective of the thesis at hand was to mainly provide valuable information for project founders, in order to maximize their success chances. Therefore, static and dynamic aspects of crowdfunding campaigns of the platform Kickstarter were analyzed and a crowdfunding data analysis framework was deducted and sketched.

The first crucial part of the thesis was the static analysis. By analyzing state of the art crowdfunding analysis literature and extensive crowdfunding platform data exploration, a range of supposedly important non-volatile project attributes was incorporated into a data mining model. This static model was able to predict the success with an accuracy of 78.5% before the project is even launched, and clearly outperformed existing crowdfunding analysis models. The static model is solely based on random forests, since they outperformed all other tested algorithms by far. Additionally, a conducted sensitivity analysis has highlighted general success drivers for crowdfunding projects. It was shown that rewarding, goal-setting, social network size and content quality signals in particular are crucial for the success of projects.

In similar fashion, a new dynamic forecasting model was proposed – the dynamic-n-forest model. Based on the volatile project attributes it prematurely predicts the success of an already launched project. Such an approach allows the project founder to react early on possibly undesirable project developments. The model already outperformed the static model with an accuracy of 82.1% after one percent of the time. The created dynamic-n-forest model was also combined with the static model to a strong synthesis model. This synthesis model was able to raise the dynamics leading to an accuracy of 83.4% after one percent of the time. The big increase in accuracy after only one percent of the online time is possible because of the rapid development of successful Kickstarter projects. This revealed the development patterns of Kickstarter crowdfunding projects.

With these models a powerful analysis framework could be built, which uses sensitivity analyses in the area of crowdfunding success prediction. The big advantage of this novel approach is, that it directly produces easily understandable diagrams based on real world projects. The CrowdData Framework provides a structured and easy way of crowdfunding project analysis, and helps the user to optimize the project settings and monitors the project success likelihood over time.

Nevertheless, it should be mentioned that at this research state, a crowdfunding project‘s success can never be predicted with a 100% certainty. There will always be some (qualitative) factors, which play a big role and cannot be captured and incorporated in the model. These capturing limitations include the quality of pictures, the video quality and the novelty of the project. Additionally, the thesis has limitations in data accessibility. Obviously due to privacy reasons, it is not possible to collect user-pledge data. With this kind of data the analyst would be able to derive behavioral patterns for the crowdfunding pledging process.

For future works, it would be desirable to dig deeper into some of the attributes. Especially a closer examination of the rewarding strategy would be interesting, since they obviously play a very important role. This could mean the inclusion of attributes like delivery date, a flag indicating early bird rewards and content metrics for the reward. There are also ways to improve the dynamic analysis, for example by adding “staff picks” to the model. Another interesting, but yet completely different data mining task would be, to predict the staff picks according to static factors. This could lead to a more objective and time saving way of highlighting special projects for Kickstarter. Furthermore, the analysis of cannibalism of crowdfunding projects, as a novel area of reasearch, is proposed here. In addition, the examination of differences across crowdfunding platforms would be an appealing research topic as well, in order to answer the question which platform a project owner should choose based on his project characteristics.

Data analysis in crowdfunding obviously has a bright future. The market is developing incredibly fast and data mining is still in its infancy. Also there is a big number of unsolved and interesting problems in this area and a huge amount of tasks, which only wait to be tapped on.

APPENDIX A

# Analysis Additions

## A.1 Attributes used for Analysis

| Attributes | | |
|---|---|---|
| goalUsd | Numeric | The goal of the campaign in USD |
| currency | Nominal | The goal currency |
| country | Nominal | The country of origin |
| subRegion | Nominal | The region of origin |
| region | Nominal | Contains a less fine-granular region |
| hasVideo | Boolean | Whether a video is present in the campaign or not |
| videoDuration | Numeric | The video duration in seconds |
| videoBitrate | Numeric | The average bits per second of the video |
| leadTime | Numeric | The time between project creation and launch in hours |
| duration | Numeric | The time between launch and end date in days |
| rewardCount | Numeric | The number of defined rewards |
| minRewardLevel | Numeric | Minimal level of rewards. The equivalent monetary value of a reward is considered as reward level, because it is a threshold value and the reward can be also chosen by the backer when he pledges more than the rewards monetary value. |
| maxRewardLevel | Numeric | Maximal level of all rewards. |
| meanRewardLevel | Numeric | The average reward level, with all reward levels taken into account. |
| stdDevRewardLevel | Numeric | The standard deviation of reward levels |
| category | Nominal | The project‘s category |
| facebookConnected | Boolean | Whether the project is connected to Facebook or not |
| facebookFriends | Numeric | The number of Facebook friends as indication for the founder‘s network |

| | | |
|---|---|---|
| creatorVerified | Boolean | Whether the founder‘s data was verified by Kickstarter |
| langEn | Boolean | Whether the description contains English text |
| langDe | Boolean | Whether the description contains German text |
| langFr | Boolean | Whether the description contains French text |
| langEs | Boolean | Whether the description contains Spanish text |
| langOther | Boolean | Whether the description contains text in any other language |
| multiLang | Numeric | A combined language flag, indicating whether multiple languages are present |
| colemanLiauIndex | Numeric | The readability grade level after Coleman and Liau |
| gunningFogIndex | Numeric | The fog index of Gunning |
| fkReadabilityScore | Numeric | The readability score of Flesch and Kincaid |
| sent | Nominal | The sentiment of the project description. Either pos., neutral or neg, determined by text-processing.com API. |
| paragraphCount | Numeric | The number of HTML paragraphs in the description |
| picCount | Numeric | The number of pictures in the description |
| wordCount | Numeric | The number of words in the description |
| sentenceCount | Numeric | The number of sentences in the description |
| wordsPerSentence | Numeric | The average number of words in a sentence |
| outOfDict | Numeric | The number of words that cannot be found in a dictionary |
| projectsBefore | Numeric | The number of projects that the user has done before |
| type | Nominal | The user type; either company, person or other |
| gender | Nominal | The user‘s gender |
| state | Nominal | The final state of the project |

Table A.1: The complete list of attributes

## A.2 Experimental Setups

This section contains experimental setups of the data mining algorithms in the different analysis tasks.

### A.2.1 Model Comparison Setup

| Algorithm | Important parameter settings |
|---|---|
| Naive Bayes | - |
| J48 | Pruning confidence Factor: 0.25, Minimum number of instances per leaf: 2 |
| Random Forest | Trees: 100, Metric: “entropy”, Number of features considered in a split: 5 |
| SimpleLogistic | - |
| LibSVM | Kernel: linear, Tolerance of termination: 0.01 |
| Multilayer Perceptron | Learning rate: 0.3, Hidden layers: 20, Training time: 50 |

Table A.2: The used algorithms and their parameter settings in the algorithm comparison of the static dataset

## A.3 Additional Figures

### A.3.1 Dynamic CrowdData Framework

The complete recorded development path of the real world project examined in the dynamic CrowdData framework.

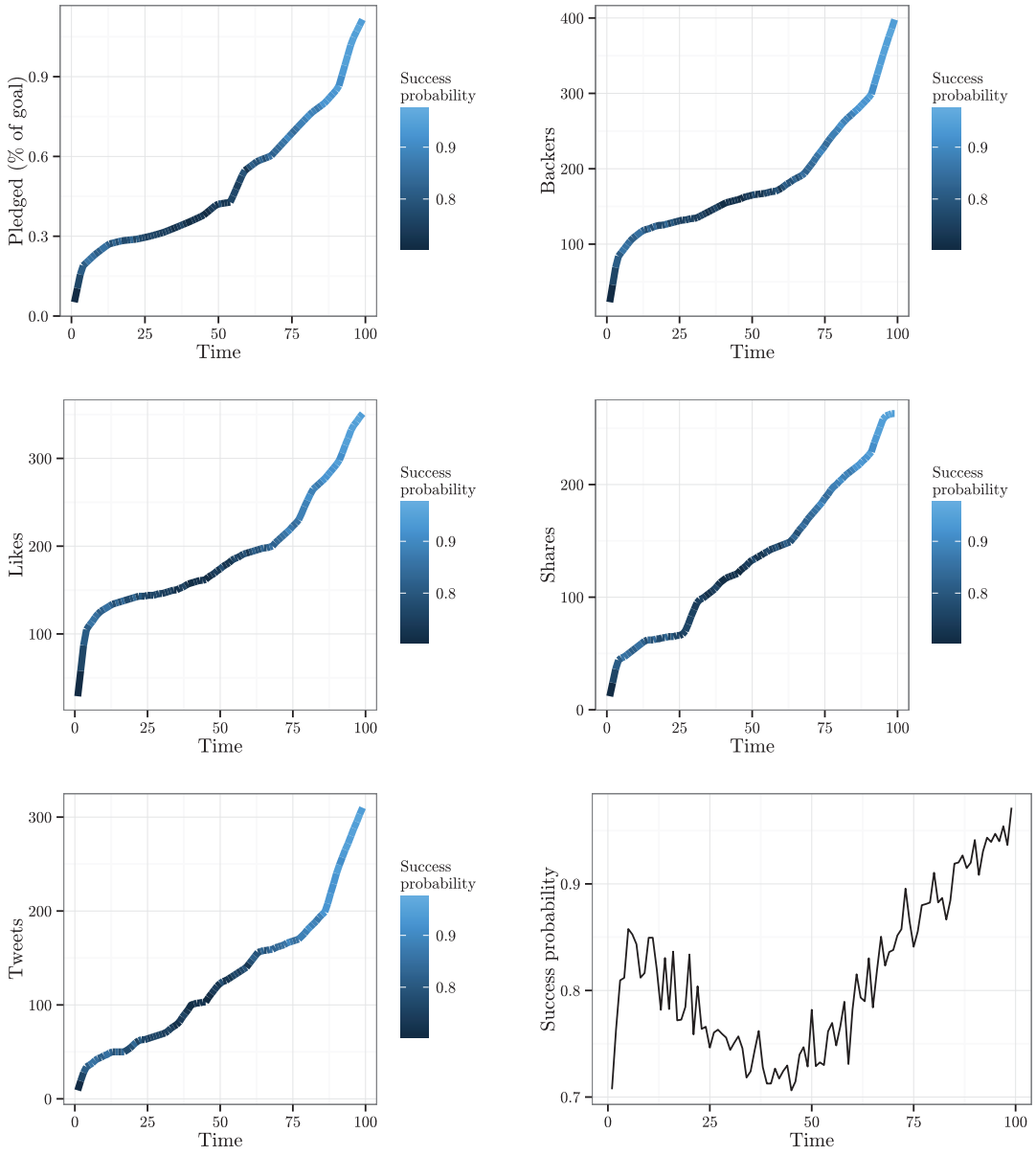



Figure A.1: The project development at 99% of the projects online time

# Bibliography


[Abe10] Shigeo Abe. *Support vector machines for pattern classification*. Of *Advances in Pattern Recognition*. Springer-Verlag London Limited, London, 2010.

[Abe15] Nitsuh Abebe. How to get featured on kickstarter. 2015. URL: `https://www.kickstarter.com/blog/how-to-get-featured-on-kickstarter` (visited on 10/04/2015).

[Acu+04] Edgar Acuña and Caroline Rodriguez. The treatment of missing values and its effect on classifier accuracy. In, *Classification, clustering, and data mining applications*, pages 639–647. Springer Berlin Heidelberg, 2004.

[Agg15] Charu C. Aggarwal. *Data mining; the textbook*. Springer International Publishing, 2015.

[Amo+15] Renato Cordeiro de Amorim and Christian Hennig. Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information sciences*, 324:126–145, 2015.

[Arm+09] Timothy G Armstrong, Alistair Moffat, William Webber, and Justin Zobel. Improvements that don't add up: ad-hoc retrieval results since 1998. In *Proceedings of the 18th acm conference on information and knowledge management*. ACM, 2009, pages 601–610.

[Ash56] W. Ross Ashby. *An introduction to cybernetics*. Chapman & Hall, London, 1956, p. 207.

[Awa+11] WA Awad and SM ELseuofi. Machine learning methods for e-mail classification. *International journal of computer applications*, 16(1), 2011.

[Aze+08] Ana Azevedo and Manuel Filipe Santos. Kdd, semma and crisp-dm: a parallel overview. In *International association for development of the information society european conference on data mining*. IADIS, 2008, pages 182–185.

[Bal+12] S Balaji and SK Srivatsa. Naïve bayes classification approach for mining life insurance databases for effective prediction of customer preferences over life insurance products. *International journal of computer applications*, 51(3):22–26, 2012.

[Bel+10] Paul Belleflamme, Thomas Lambert, and Armin Schwienbacher. Crowdfunding: an industrial organization perspective. In *Workshop digital business models: understanding strategies*, 2010, pages 25–26.

[Bre+00] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Acm sigmod record*. Volume 29. (2). ACM, 2000, pages 93–104.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Buy+12] Kristof de Buysere, Oliver Gajda, Ronald Kleverlaan, and Dan Marom. *A framework for european crowdfunding*. s.n., Germany, 1st ed edition, 2012.

[Cha+00] Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rudiger Wirth. Crisp-dm 1.0 step-by-step data mining guide. Technical report (CRISPMWP-1104). SPSS, 2000.

[Cha+02] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*:321–357, 2002.

[Cha+06] Soumen Chakrabarti, Martin Ester, Usama Fayyad, Johannes Gehrke, Jiawei Han, Shinichi Morishita, Gregory Piatetsky-Shapiro, and Wei Wang. Data mining curriculum: a proposal (version 1.0). *Intensive working group of acm sigkdd curriculum committee*, 2006.

[Cor+11] Paulo Cortez and Mark J Embrechts. Opening black box data mining models using sensitivity analysis. In *Ieee symposium series in computational intelligence 2011 (ssci 2011)*, 2011, pages 341–348.

[Cor+13] Robson Leonardo Ferreira Cordeiro, Christos Faloutsos, and Caetano Traina Jr. *Data mining in large sets of complex data*. Of *Springer Briefs in Computer Science*. Springer, 2013.

[Cri+11] A Criminisi, J Shotton, and E Konukoglu. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. *Microsoft research cambridge, tech. rep. msrtr-2011-114*, 5(6):12, 2011.

[Cum+14] Douglas J Cumming, Gaël Leboeuf, and Armin Schwienbacher. Crowdfunding models: keep-it-all vs. all-or-nothing. In *12th international paris finance meeting eurofidai - affi*, 2014.

[Dav+06] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on machine learning*. ACM, 2006, pages 233–240.

[Den+12] Naiyang Deng, Yingjie Tian, and Chunhua Zhang. *Support vector machines: optimization based theory, algorithms, and extensions*. CRC Press, 2012.

[Eth13] Darrell Etherington. Kickstarter reportedly owns indiegogo with around 6x more total dollars raised, average success rate much higher. 2013. URL: http://techcrunch.com/2013/08/30/kickstarter-owns-indiegogo-with-around-6x-more-total-dollars-raised-average-success-rate-much-higher/ (visited on 06/16/2015).

[Ett+13] Vincent Etter, Matthias Grossglauser, and Patrick Thiran. Launch hard or go home!: predicting the success of kickstarter campaigns. In *Proceedings of the first acm conference on online social networks*. ACM, 2013, pages 177–182.

[Eve98] B. S. Everitt. *The cambridge dictionary of statistics*. Cambridge, UK: Cambridge University Press, 1998.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[Fay+96] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From data mining to knowledge discovery in databases. *Ai magazine*, 17(3):37, 1996.

[Gel+06] Andrew Gelman and Jennifer Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press, 2006.

[Gre+13] Michael D Greenberg, Bryan Pardo, Karthic Hariharan, and Elizabeth Gerber. Crowdfunding support tools: predicting success & failure. In *Proceedings of the 13th acm conference on human factors in computing systems*. ACM, 2013, pages 1815–1820.

[Gre+14] Jason Greenberg and Ethan R Mollick. Leaning in or leaning on? gender, homophily, and activism in crowdfunding. *Gender, homophily, and activism in crowdfunding (july 3, 2014)*, 2014.

[Gru69] Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[Grz+10] Jerzy W Grzymala-Busse and Witold J Grzymala-Busse. Handling missing attribute values. In, *Data mining and knowledge discovery handbook*, pages 33–51. Springer, 2010.

[Guj12] Damodar N Gujarati. *Basic econometrics*. Tata McGraw-Hill Education, 2012.

[Gup06] G.K. Gupta. *Introduction to data mining with case studies*. Prentice-Hall Of India Pvt. Limited, 2006.

[Hai+10] Joseph.F. Hair, William C. Black, Barry J. Barbin, and Rolph E. Anderson. *Multivariate data analysis*. Of *Always learning*. Prentice Hall, 7th edition, 2010.

[Han+01] David J. Hand, Padhraic Smyth, and Heikki Mannila. *Principles of data mining*. MIT Press, Cambridge, MA, USA, 2001.

[Han99] David J Hand. Statistics and data mining: intersecting disciplines. *Acm sigkdd explorations newsletter*, 1(1):16–19, 1999.

[Haw80] Douglas M Hawkins. *Identification of outliers*. Volume 11. Springer, 1980.

[He+09] Haibo He and Edwardo Garcia. Learning from imbalanced data. *Knowledge and data engineering, ieee transactions on*, 21(9):1263–1284, 2009.

[Hea+98] Marti A. Hearst, Susan T Dumais, Edgar Osman, John Platt, and Bernhard Scholkopf. Support vector machines. *Intelligent systems and their applications, ieee*, 13(4):18–28, 1998.

[How06] Jeff Howe. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.

[Hua11] Gao Hua. Customer relationship management based on data mining technique. In *International conference on e -business and e -government (icee)*, 2011, pages 1–4.

[Kha+10] Amir E Khandani, Adlar J Kim, and Andrew W Lo. Consumer credit-risk models via machine-learning algorithms. *Journal of banking & finance*, 34(11):2767–2787, 2010.

[Kin+75] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report. DTIC Document, 1975.

[Kir+14] Eleanor Kirby and Shane Worner. Crowd-funding: an infant industry growing fast. *International organization of securities commissions staff paper*, 2014.

[Kot07] S. B. Kotsiantis. Supervised machine learning: a review of classification techniques. In *Proceedings of the 2007 conference on emerging artificial intelligence applications in computer engineering: real word ai systems with applications in ehealth, hci, information retrieval and pervasive technologies*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 2007, pages 3–24.

[Kov05] A. Kovalick. *Video systems in an it environment: the essentials of professional networked media*. Taylor & Francis, 2005. URL: `https://books.google.at/books?id=UzODX-pRSzYC`.

[Lan+77] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *Biometrics*:159–174, 1977.

[Mai+10] Oded Maimon and Lior Rokach. *Data mining and knowledge discovery handbook*. Springer Science+Business Media, LLC, 2010. edited by Oded Maimon, Lior Rokach.

[Mar+96] J Kent Martin and DS Hirschberg. On the complexity of learning decision trees. In *International symposium on artificial intelligence and mathematics*, 1996, pages 112–115.

[Mar03] Florian Markowetz. Klassifikation mit support vector machines. 2003. URL: `http://lectures.molgen.mpg.de/statistik03/docs/Kapitel_16.pdf` (visited on 06/20/2015).

[Mas15] Massolution. *2015cf. the crowdfunding industry report*, 2015.

[May+13] Viktor Mayer-Schönberger, Cukier Mayer-Schönberger, et al. *Big data: die revolution, die unser leben verändern wird*. Redline Wirtschaft, 2013.

[Mic89] Theodore Micceri. The unicorn, the normal curve, and other improbable creatures. *Psychological bulletin*, 105(1):156, 1989.

[Mit+14] Tanushree Mitra and Eric Gilbert. The language that gets people to give: phrases that predict success on kickstarter. In *Proceedings of the 17th acm conference on computer supported cooperative work & social computing*. ACM, 2014, pages 49–61.

[Moh+12] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. Of *Adaptive computation and machine learning series*. MIT Press, Cambridge (Mass.), London, 2012.

[Mol14] Ethan Mollick. The dynamics of crowdfunding: an exploratory study. *Journal of business venturing*, 29(1):1–16, 2014.

[Osb+04] Jason W Osborne and Amy Overbay. The power of outliers (and why researchers should always check for them). *Practical assessment, research & evaluation*, 9(6):1–12, 2004.

[Pia14] Gregory Piatetsky. Kdnuggets 15th annual analytics, data mining, data science software poll: rapidminer continues to lead. 2014. URL: `http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-software-used.html` (visited on 10/03/2015).

[Ras+07] Zbigniew W. Ras, Shusaku Tsumoto, and Djamel Abdelkader Zighed. Mining complex data. In *Proceedings of the third international workshop on mining complex data mcd*. Springer, 2007.

[Ris+01] Irina Rish, Joseph Hellerstein, and Jayram Thathachar. An analysis of data characteristics that affect naive bayes performance. Technical report. 2001.

[Roj13] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[Ros+07] Andrew Rosenberg and Julia Hirschberg. V-measure: a conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (emnlp-conll)*. Association for Computational Linguistics, June 2007, pages 410–420.

[Saf+91] S. R. Safavian and D. Landgrebe. A survey of decision tree classifier methodology. *Systems, man and cybernetics, ieee transactions on*, 21(3):660–674, 1991.

[Sch+02] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[Sch14] Wil Schroter. Top 10 business crowdfunding campaigns of all time. 2014. URL: `http://www.forbes.com/sites/wilschroter/2014/04/16/top-10-business-crowdfunding-campaigns-of-all-time/` (visited on 10/01/2015).

[Si+01] Luo Si and Jamie Callan. A statistical model for scientific readability. In *Proceedings of the tenth international conference on information and knowledge management*. ACM, 2001, pages 574–576.

[Sto+11] J.H. Stock and M.W. Watson. *Introduction to econometrics*. Of *Addison-Wesley series in economics*. Addison-Wesley, 2011.

[Tan+05] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining*. Pearson Addison Wesley, Boston, 2005.

[Tay13] Kate Taylor. 6 top crowdfunding websites. which one is right for your project? 2013. URL: http://forbes.com/sites/katetaylor/2013/08/06/6-top-crowdfunding-websites-which-one-is-right-for-your-project/ (visited on 12/03/2014).

[Tri+06] Evangelos Triantaphyllou and Giovanni Felici. *Data mining and knowledge discovery approaches based on rule induction techniques*. Volume 6. Springer Science & Business Media, 2006.

[Wit+11] Ian H. Witten, Eibe Frank, and Geoffrey Holmes. *Data mining : practical machine learning tools and techniques*. Of *The Morgan Kaufmann series in data management systems*. Morgan Kaufmann, Amsterdam, Boston, Paris, 2011.