universität wien

MASTERARBEIT / MASTER'S THESIS

"Leveraging User-generated Content on Online Media for Assessing Crowdfunding Projects"

verfasst von / submitted by

Lukas Draschkowitz, BSc.

angestrebter akademischer Grad / in partial fulfillment of the requirements for the degree of

Diplom Ingenieur (Dipl. Ing.)

Wien, 2018 / Vienna, 2018

Studienkennzahl lt. Studienblatt / degree program code as it appears on the student records sheet:	A 066 935
Studienrichtung lt. Studienblatt / degree program as it appears on the student record sheet	Masterstudium Medieninformatik
Betreut von / Supervisor:	Univ. Prof. Dr. Wolfgang Klas

Declaration of Authorship

I, DRASCHKOWITZ LUKAS, declare that this thesis titled, 'Leveraging User-generated Content on Online Media for Assessing Crowd-funding Projects' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ____________________________________

Date: ____________________________________

Statt zu klagen, dass wir nicht alles haben, was wir wollen, sollten wir lieber dankbar sein, dass wir nicht alles bekommen, was wir verdienen.

Unknown

Contents

List of Figures

Listings

Abbreviations

CF	**C**rowd**f**unding
URL	**U**niform **R**esource **L**ocator
URI	**U**niform **R**esource **I**dentifier
REST	**Re**presentational **S**tate **T**ransfer
TF	**T**erm **F**requency
IDF	**I**nverse **D**ocument **F**requency
NLP	**N**atural **L**anguage **P**rocessing
API	**A**pplication **P**rogramming **I**nterface
JSON	**J**ava **S**cript **O**bject **N**otation
HTML	**H**yper**t**ext **M**arkup **L**anguage
JSP	**J**ava **S**erver **P**ages

Abstract

Crowdfunding is a widely used way of funding by groups of investors. Predicting success of newly launched crowdfunding projects is valuable for both, the project creators and project funders. Time plays an important role here as you want your predictions as soon as possible to be effective. The most important prediction features are not known at the time of the project start. Online media data could assist in tracking popularity of projects beforehand. In this work a framework is proposed, that automatically extracts features from crowdfunding projects and at the same time measures projects novelty based on whether similar projects are on the market and projects popularity by leveraging online media data. The prototype implementation also provides an user interface to browse through data. Predictions based on the novelty and popularity score were not found successful the way they were implemented. Nevertheless, the framework is designed to be very flexible and modular and can therefore be extended easily to bring new data into the system. In general the system can be adjusted to any kind of prediction problem, that relates to data available on the Web.

Zusammenfassung

Crowdfunding ist eine weit verbreitete Art der Finanzierung von Projekten durch Gruppen von Investoren . Die Voraussage des Erfolgs von neu gestarteten Projekten kann sowohl wertvoll für die Ersteller dieser Projekte als auch für die Unterstützer sein. Zeit spielt in dieser Hinsicht eine wichtige Rolle, da die Voraussagen so schnell wie nur möglich getroffen werden müssen, um diese effektiv zu nutzen. Die wichtigsten für die Voraussage notwendigen Eigenschaften von Projekten sind zum Zeitpunkt der Erstellung des Projektes allerdings noch nicht bekannt. Daten aus Onlinemedien könnten dabei helfen, die Popularität dieser Projekte schon zu diesem Zeitpunkt zu bestimmen und die Voraussage zu unterstützen. In dieser Arbeit wird ein Framework vorgestellt, welches Eigenschaften von Crowdfunding Projekten vollautomatisch extrahiert und gleichzeitig Einzigartigkeit anhand der Tatsache, ob ähnliche Projekte am Markt sind und die Popularität anhand von Onlinemedien misst. The Implementierung eines Prototypen verfügt darüber hinaus über eine Benutzeroberfläche für die Durchsicht der Daten. Auf den beschriebenen Eigenschaften Einzigartigkeit und Popularität basierende Voraussagen, so wie sie implementiert wurden, führten zu keinen korrekten Ergebnissen. Nichtsdestotrotz ist das vorgestellte Framework modular und flexibel konzipiert und kann daher einfach erweitert werden, um neue Datenquellen zu erschliessen und das Voraussagemodell zu verbessern. Ganz allgemein kann das System auf jedwedes Problem angepasst werden, welches Bezug zu Onlinedaten aus dem Internet hat.

Chapter 1

Introduction

1.1 Motivation

The collective power of the crowd in online communities have been leveraged since the rise of the internet in order to serve business goals, improve public participation in governance, design products, and solve problems. A recent phenomenon in this context is crowdfunding, where online platforms such as *Kickstarter* assist to raise funds for projects. Users on such platforms require support in their decision to fund or submit a project and to identify success drivers. A system, capable of predicting the success of projects and computing performance indicators, can be valuable for project creators in the process of the definition and creation of their crowdfunding project as well as for funders - the so called backers - that naturally aim to invest as soon as possible in projects, that ultimately succeed. [1], [2]

Online media data as the source of information in the process of prediction assures the supply of classical news data on one hand and user generated data on the other hand. News information used gained from newspapers but as everything turns digital and has digital representations can now be found online for the most part The growing fraction of user generated data and associated social network statistics and metrics are potential key factors in predicting novelty, popularity and successfulness of crowdfunding projects.

With the ever increasing number of crowdfunding projects - 3790 active projects on 19.07.2018 on Kickstarter [3] - there is a whole lot of information to process. So investors can benefit by saving time and possibly money.

One typical way to assess crowdfunding projects for potential funders is to gather information from relevant news pages or social media networks about trending topics and interesting projects. The main attention lies on rather big projects that already gathered some kind of momentum and possess a promising degree of popularity and a good number of backers from their utilized network at that time. Projects, in their early stages, as well as smaller sized crowdfunding projects, will therefore be easy to miss. That is against the idea of crowdfunding in the first place, that utilizes the crowd to contribute small amounts and therefore also back projects of all sizes rather than only the big ones.

Speaking of small amounts of money, investors will probably spend less time on assessing investments in this environment and consequently may be missing a deeper understanding of the matter.

By the time a project provides sufficient amount of data for the use in a machine learning prediction, a successful project will likely be already widely known. Therefore time is also a factor as some of the rewards, that are provided for backers are limited. If one is willing to wait for a considerable fraction of the project time, it is not hard to predict the successfulness for that project anyway.

Being able to automatically create datasets to work on is a profitable capability by itself. Statistics show that areas, where there are open datasets available, are boosting in contrast to areas where large open datasets of proper quality are not available 1.1 [4].

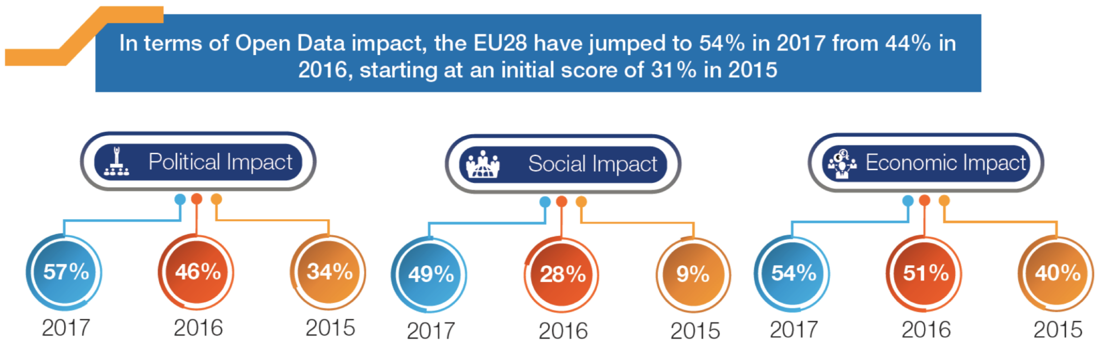


FIGURE 1.1 | **The Impact of Open Data.** The figure shows current statistics from data acquisition about the european data portal that provides open data. [4] The term *Open data* refers to "publicly available data that can be universally and readily accessed, used and redistributed free of charge. It is structured for usability and computability"[5] here.

The proposed framework is set to be modular and universal enough to adapt to similar use cases that do not involve crowdfunding projects but other kind of data, that is available online and can be assessed through social media data.
The key features of the system namely continuous automatic data extraction and popularity, novelty and success scoring remain intact. Data could as well come from other social media platforms and accordingly the system could be scoring other kind of entities like movies. The framework is able to gather information and build data sets on it's own.

1.2 Definition and Methodology

This master thesis deals with the development of a framework, that is able to identify novelty of a newly launched crowdfunding project by searching for similar projects on other platforms, to identify popularity of a newly launched project by searching online media sources such as social media and news platforms and to ultimately predict and present the success for newly launched crowdfunding projects.

Further implementing additional computed features enables the system to classify projects at a time where important features of that project do not exist or are not useable just yet. An example would be the number of backers - a feature that almost certainly will be ignorable for projects that launched only minutes ago. Besides that, these features could potentially have positive impact on the success prediction process.

The work aims at realizing a system that automatically determines the novelty, popularity and successfulness of new crowdfunding projects. This is achieved by crawling and using data from crowdfunding websites, social media and online news platforms periodically from the web.
Relevant data is turned into structured information to be processed by machine learning, natural language processing, text mining and other techniques to compute future success of projects.

The assumption is, that in order for a project to have high novelty, it must be the case, that there are very little similar crowdfunding projects around. Therefore a measure called novelty is defined for this application.
As we will see in detail in 2.1, a crowdfunding project is for the main part defined

by its name (and description). Other features, like the projects deadline or its rewards and videos, are also important to the project - especially for the projects success of course - but they don't reflect the projects concept or uniqueness. In crowdfunding projects names seem to be very meaningful and descriptive in most cases. Acronyms are rarely used as creators want to give an idea of what they are offering to the user without having them to click on the project description.

> "The headline title of your campaign is what your prospective backers will see first. It's what they call you. It's the first thing they begin to understand about your crowdfunding campaign." [6]

Furthermore, as described in [7] "Campaigns with longer descriptions and names are more likely to be successful". Therefore it is assumed that the description doesn't even have to be included in similarity computation in most cases.
So projects where name and/ or description are closely related to each other are potentially very similar and by searching for similar projects we can further define its novelty by measuring the string similarity between projects.
This is the reason why integrating various crowdfunding platforms into the proposed system is possible in the first place. As for users comparing projects of different platforms is hardly possible and definitely inaccurate and time intensive.

The approach to measuring popularity is a very similar one. Again, text similarity between crowdfunding project and online news articles will be measured, this time refined with data from Twitter. Similar to what Asur et. al [8] do in their work where Twitter is queried for movie keywords to predict popularity of movies. The proposed approach though goes beyond that by adding online news data and data analysis to measure popularity.
As stated before the definition of an ideal crowdfunding project name is very similar to titles on a newspaper article, an advertisement, or blog posts. All of them seem to somehow describe the concept of their contents.
So the idea is to extract a project's concept and search for that concept in online media, as being a factor in online media is assumed to be equivalent to supplying a trending topic.
Consequently, text similarity between project and online media might be an indicator for project popularity. These assumptions will be integrated into a model, that is used to compute popularity and novelty features.

The realization of a prototype as an implementation of the proposed framework enables the validation of all of the described model assumptions about crowdfunding project novelty, popularity, and success.

1.3 Outline

The rest of this master thesis describes scientific background, related work, the application, and results of the work. The thesis is structured as follows:

- The thesis starts with an Abstract, giving a short overview of the work
- Chapter 1 gives an Introduction, that describes the motivation and intention of the experiment
- Chapter 2 is about the Scientific Background about related scientific fundamentals
- Chapter 3 covers Related Work, that addresses social media mining and particularly crowdfunding prediction
- Chapter 4 describes the Conceptual Description of the Framework including Requirements, conceptual design of the approach taken, software design, and the developed architecture.
- Chapter 5 is about the Implementation and Technologies and describes the work towards the implementation of the prototype.
  The prototype system consists of a restful API, that provides a web user interface to present its prediction and analysis results. It is able to automatically find similar projects, retrieve popularity and overall opinion about projects and to predict success for projects. Hereby the use of data from two crowdfunding websites, Twitter and online news platforms is implemented.
- Chapter 6 presents Results and Evaluation of the prototype implementation, the measuring of popularity, novelty, and prediction of successfulness.
- Chapter 7 delivers a Conclusion including a summary, limitations of the approach taken and future work

- Chapter A provides an Appendix, that lists the URLs crawled by the prototype in order to assess news platforms.

Chapter 2

Scientific Background

2.1 Crowdfunding

"Crowdfunding is an initiative undertaken to raise money for a new project proposed by someone, by collecting small to medium-size investments from several other people (i.e. a crowd)[9]"

Crowdfunding had a massive rise in the recent past and particularly effected smaller projects and startups, that are now able to generate money over the internet without having to convince someone to risk a lot of money. Instead many people spend little money to realize the same. There are different roles involved in this process. The *creators* trying to raise money and getting access to the market by presenting their ideas or projects. They determine the kind of reward that the so-called *backers* or *funders* receive for funding. This can be the finished product or something completely different. Secondly there is the crowd that financially support a project and lastly, there are the crowdfunding organizations, that bring together these people. There are many different organizations at the market, each providing slightly different conditions and rules but the general idea remains the same. The two most popular platforms are *Kickstarter* and *Indiegogo* [10], [11], [9], [12].

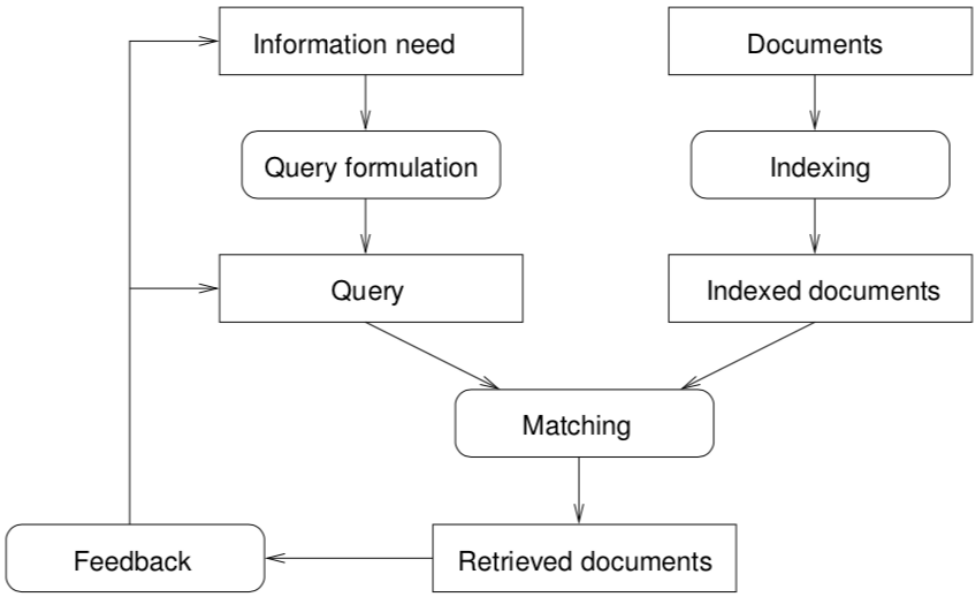


FIGURE 2.1 | **The Process of Information Retrieval.** Description of the general workflow for an information retrieval system [14]

2.2 Information Retrieval System

"Information Retrieval is the discipline that deals with retrieval of unstructured data ... in response to a query or topic statement" [13]. In many cases the information, that is retrieved, is a collection of textual documents. The demand for automated information retrieval methods is rising with the growing amount of unstructured data especially but not solely related to the web.
An information retrieval system then is the program, that stores and manages this kind of information to assist in finding the needed information.

The necessary steps in such a system are described in 2.1. Firstly documents need to be represented in the system. This is referred to as indexing, where most of the time only important information, that can also be derived from the document, is stored while the original document doesn't have to be maintained in the index. Users of the system then formulate a query from their information need, that has to be matched to indexed documents to gather a subset of relevant documents to be returned to the user. This matching process is usually realized through scoring models that assign high scores to documents that are close matches to the query and consequently enable the creation of a ranked document list of which the most relevant are returned in the computed order [13], [14], [15].

To obtain opinion from data the story is slightly different. The major difference is that in a traditional search one would only need the top ranked pages because they already contain all information - "one fact equals to any number of the same fact" [16]. In opinion mining it is important "to reflect the natural distribution of positive and negative opinions". For instance, opinion related to high information content is probably more important for searches about opinion. Further, sentiment analysis can be used to additionally determine whether or not and what kind of opinion is expressed in a text [16].

2.2.1 Apache Lucene

Apache Lucene is an open source project and "is considered the go-to text search framework by many" [17]. It provides mechanisms to store information in an index and to search for it. Thereby it tokenizes phrases and stores keywords with reference to the respective documents they came from in an inverted index. It also applies text analyzation techniques like stop word filtering, stemming, text normalization, and synonym expansion.
The Lucene standard scoring implementation leverages the Boolean Model and the Vector Space Model.

In this project Apache Lucene is used as the information retrieval system. Data is stored in a Lucene index to be able to search for similarity between records and newly incoming crowdfunding projects rapidly. Over Java APIs Lucene provides direct access to low level functions like *MorelikeThis* and other scoring functions that implement TFIDF-based scoring models amongst others [17].

2.2.2 Natural Language Processing

Natural Language Processing (NLP) "is the ability to train computers to understand both written text and human speech"[18]. It combines methods from linguistics and information technology to process natural language.
It is a major part of information retrieval, that retrieves data from natural language, and artificial intelligence (AI), where intelligent behavior among other things means understanding language [19].

In order to understand natural language knowing the usual meaning of a word is not enough. "It is necessary to convey the meaning of the entire message [20]". To work out the meaning of text operations like tokenization, splitting text into entities like words or sentences, and syntactic and semantic analysis of the text are required [20]. Two common application areas are language translation and sentiment analysis. Sentiment analysis determines whether or not and what kind of opinion is expressed in a text.

2.2.3 Text Similarity

Measuring the similarity between words, sentences, paragraphs, and documents is an important component of information retrieval systems and the most important part of Apache Lucene's similarity scoring models.
Finding similarity in words is the first step in this process. Words can be similar lexically, where the word's characters are similar and semantically, where those words have similar meaning, are used in the same context or are of the same type. The former type of similarity is measured by string based metrics that take string sequences and character composition into account and measure similarity or rather distance that amounts to dissimilarity between two strings. To measure the second type of similarity, corpus or knowledge based-algorithms are used that determine the similarity between words according to information gained from large corpora or respectively semantic networks. Of course solutions that combine both approaches exist. One of them is the standard similarity implementation offered by Lucene that uses Boolean and Vector Space Model combined with synonym expansion [21], [17].

2.2.4 Boolean Model

The boolean model matches query terms to indexed documents where the matching is based on the three logical boolean expressions AND, OR and NOT. These expressions have to be derived from a users query and are further used to return a set of matching documents. This works fine in Apache Lucene for instance as it maintains a term index storing all documents a term appears in. This method finds relevant documents but has the disadvantage of not being able to rank relevant documents [17] [22].

2.2.5 Vector Space Model

Ranking can be accomplished by implementing Vector Space Model. To search for a document this model generates a document from the query to further quantify similarity between query document and each document in the collection. Considering the representation of documents as vectors embedded in a high dimensional Euclidean space, where each term is assigned a separate dimension, the similarity measure is the cosine of the angle that separates the two vectors. What this model does, is treating all terms of a document equally, which doesn't hold true for reality as there will be certain words that are more valuable matches than others [22].

TFIDF Similarity

TFIDF is an implementation of the vector space model that adds weighting to the terms represented in the vector. Weights are based on term frequency (TF) quantifying the frequency of the term in the document and the inverse of the document frequency (DF), which is the number of documents containing a certain word. The weighted score for a certain word (W) is then defined as the product of TF and IDF. Different schemes of TFIDF scoring exist, that combine slightly different variations of TF and IDF formulas. 2.1 shows those, that are used as part of the similarity computation in *Lucene*

$$TFIDF(W) = TF * (1 + log(\frac{N}{(DF(W) + 1})) \tag{2.1}$$

, where N represents the total number of documents, TF equals the number of times the word appears in the document to be scored and DF is the number of documents containing the word [23], [24], [14], [22], [25].

2.2.6 Crawling the Web

A crawler is essentially a program that downloads and stores web pages from the Internet.
To be able to find web pages the crawler would have to start with some seed URLs. From there, web pages are downloaded and content will be parsed. In this process data to be stored will be found but at the same time unseen links will appear on

this page. So in following the links to the next web page the crawler carries on and repeats the process.

There is "a lot that goes into creating a full-fledged Web crawling application[26]" like adhering the *Robot Protocol* or handling scenarios like Web server errors, redirects, duplicate links, uninformative pages, page-flipping links or entry URL discovery. The *Robot Protocol* is specified in a file called *robots.txt*, that basically contains restrictions for crawlers. Websites can define restricted areas or restrict the number of requests in a certain time window. Other problems arise from *Cloaked Content* and the *Deep Web*. Cloaking refers to the practice of serving different content to web crawlers than to human viewers of a site. Many websites rely heavily on JavaScript. The *Deep Web* means "web sites whose content is not reachable via hyperlinks and instead can only be retrieved by submitting HTML forms [27]" [26], [28], [27], [29].

2.3 Data Mining

Data mining is about "uncovering hidden trends and developing new data and information from data sources" [30]. These sources include structured data we find in relational databases, but can also be multimedia sources like videos or written texts and other information from the web.
Data mining originates from three distinct disciplines, namely statistics, artificial intelligence, and machine learning and uses tools and methods from these fields in order to gather knowledge from data [30].

Social Media Mining

Social media mining is an emerging discipline in data mining, that covers "the process of representing, analyzing, and extracting actionable patterns from social media data [31]".
In addition to traditional data mining it integrates social theories and collects information about individuals, entities and their interactions, and consequently reveals new challenges. For instance, while social media data is undoubtedly big, often there is only very little data for individuals available. This is referred to as the *Big Data Paradox* [31].

Twitter

Twitter is a microblogging platform where users receive short messages called *Tweets* from other users by following them. There are other possibilities to interact. "Individuals on Twitter have the opportunity to forward tweets to a broader audience via the retweet capability [31]". There is also a favoriting function that is used quite similar to retweeting but more often used as a way of bookmarking according to [32] and is not presented to other users apart from the one, who posted the original Tweet.

Twitter launched in back 2006 and meanwhile generated a user base of approximately 336 million active users per month according to [33]. Due to its high outreach Twitter is used by news organizations, to spread news updates in the community and is also used by businesses "to advertise products and disseminate information to stakeholders [8]" [31], [8], [32].

2.4 Machine Learning

A scientific model is a set of assumptions about the data used to make predictions from unseen data. Machine learning is required to adjust the model parameters to patterns in data. It enables computer systems to learn from data.
There is a distinction between supervised and unsupervised machine learning techniques where the later cover methods that work on unlabeled data. Supervised learning can be applied when class labels are present in the data. These class labels are used to train a machine learning model, that finally predicts class labels for unlabeled data. Predicted class labels are usually evaluated on a test set after the classification process, that fits the model to the training set. If the model performs better on the training set than on the test set, the model is over-fitted. At this point the model adjusts to aspects of the data that are not relevant to the general prediction problem. On the other side of the spectrum the model can be under-fitted, hence performing poorly due to over-generalization [30], [19], [34], [35].

2.4.1 Neural Network

Neural networks "simulate human synaptic activity by modeling the brain's mesh-like network of neurons and axons as perceptrons and weighted links [30]". Based on training data the weights can be adjusted to the problem and complex data patterns can be found in the data. They consist of an input layer, where data is ingested, one or many hidden layers, and an output layer of perceptrons [30], [34], [35].

2.4.2 Tree Boosting

Decision tree classifiers are among the so-called white box machine learning classifiers. One can draw conclusions from the way the model is built. The so-called black box classifiers on the other hand provide no information on the internal decision process modeled.
Decision trees consist of nodes and branches where nodes represent features of the dataset and branches represent values features can assume. To classify an instance of the dataset one would have to follow the tree upwards from the root node. At every node the decision, which branch to follow has to be made according to the feature value of the instance to be classified. Leafs of the tree correspond to class labels.

[36], [35].
Boosting is a method, that combines weak classifiers to create a new classifier, performing significantly better. Varying algorithms implementing this idea exist and one of the them is called *AdaBoost*. It stands for adaptive boosting and operates by running classifiers on a dataset one after the other. Along the way weights are assigned to data points according to how difficult previous classifiers have found to get them correct [34].
To summarize this, tree boosting is a boosting method, that combines tree classifiers [36].

2.4.3 Sequential Learning

"In many modern applications data is available only in a streaming fashion, and one needs to predict labels on the fly [37]". So opposed to offline learning, where data is static and ready before model training, in online and batch learning new data is added frequently. In batch learning new data comes into the system in batches and the machine learning model will be updated based on the new data, whereas in online learning the model is updated with each new observation [30]. [38] experiences slightly better results for offline and batch learning while significant performance improvement was measured for the online learning algorithm, which makes it superior especially in big data applications.

Chapter 3

Related Work

3.1 Crowdfunding Prediction

Some work has been done on the topic of predicting the success of crowdfunding projects and very different approaches and conclusion have been taken. Nevertheless most of the work is based on analyzing basic crowdfunding project features like the number of backers. Most of the time projects, that got no funding were removed from the data set.

Online media as the source of information in prediction problems has been leveraged quite a lot in recent years. But as we will see no work has been done on leveraging the full spectrum of Twitter or online media data to predict crowdfunding project success.

In the following related work regarding crowdfunding project success prediction and prediction based on online media data will be described shortly.

[39] built a model to predict Kickstarter project success and reached an accuracy of about 72 percent using features from a *Kaggle*[1] dataset containing standard features from Kickstarter projects plus the number of the project owners Facebook friends. They further demand, that projects, that merely miss funding should be added to the Kickstarter's *Staff Pick* list, that represents recommendations from the Kickstarter team. They also state that "staff picks are nearly 10 times as likely to succeed as campaigns without the flair [39]". The importance of staff picks is also described in [40].

[1]www.kaggle.com

[41] achieved 68 percent accuracy in their predictions of crowdfunding projects whith simple classifiers working better for their data. They used scraped data including data generated from sentiment analysis, grade analysis and social media connection, but only assessed the number of friends or respectively subscribers regarding the analyzed project. They also included some calculated features like the number of sentences in the project description or the number of followers on Twitter.
Finally they observed, that the "accuracy seems to hit an upper bound of 67percent irrespective of how we break down the dataset. This suggests that there is a possibility of the existence of a hidden variable that would help us classify better [41] ".

In [40] Downs et. al. experimented with different algorithms with the standard Kickstarter feature set and ended up with an accuracy of 67 percent.

[42] use a combination of predictors that work on time series data of money pledges on Kickstarter and predictors working on tweets about that specific project and Kickstarter's project data. Relations to the backers and their respective data are represented in a graph, reaching impressive 76 percent accuracy for projects already 4 hours after the project launch.

[43] realized that 3 percent of rewards on Kickstarter where not shipped to investors on time and therefore the authors propose a model that predicts if creators will be able to deliver rewards on time.

[7] removed weak unsuccessful projects right away: "We remove projects that were canceled or suspended, with less than one backer, or with less than 100 dollar of pledged amount" and "show that the on-going status (or popularity) of the projects plays an important role in improving the recommendation performance [7]".
They predicted the time for reaching the projects goal through survival analysis by using data from Kickstarter, Twitter and Facebook stating that Twitter posts from the first 3 days after project launch are the most relevant.

[12] predicted the number of backers and the success of projects to demonstrate "that the features arising from social media can help improve the accuracy of predicting eventual outcomes, specifically how popular a project will be, and which projects are likely to successfully reach the fundraising goals [12]".

3.2 Online Media Mining Applications

Asur et. al [8] leveraged Twitter data by searching for movie keywords on Twitter to predict popularity and finally success of movies. They utilized keywords from the movie title as search arguments when calling the *Twitter Search API*.
They also generalized their prediction model for social media to work on any product by measuring the rate of chatter over time for a product and then fitting a linear regression model. They compute a ratio between positive and negative tweets concerning a movie to measure polarity.

Bo et. al [44] predicted popularity of news on Twitter and found among other things that negative sentiment correlates with retweet popularity.

[45] "predicted the popularity of news items on Twitter using features extracted from the content of news articles [45]" and reached an overall accuracy of 84 percent by taking into account information derived by natural language processing methods like category, subjectivity in the language, and named entities.

[46] measured user influence on Twitter on a large scale by the measures indegree, meaning the number of followers, retweets, and mentions. found that indegree measures a user's popularity but not necessarily the user's influence in contrast to retweets.

[47] used a system called Lydia to automatically analyze 1000 online newspapers and predict movie grosses.

[48] described a potential architecture for opinion mining on social media. In contrast to other approaches like reading emoticons they "use domain-specific training data to build a generic classification model from social media data to help improve the performance".

[49] predicted collective behavior of people by training a nature-inspired model on available online blogs using swarm intelligence by observing their interactions.

Chapter 4

Conceptual Description of the Framework

The proposed framework is designed to generate an autonomous system, that is able to gather online data to score and update a list of the most promising new crowdfunding projects on the fly.

The application is designed to create a running system, that is able to gather online data to score and update a list of the most promising new crowdfunding projects on the fly.

The context of the work is the extraction and analysis of project-relevant data from the Web for the process of assessing novelty, popularity, and successfulness respectively. To meet those demands the following tasks are implicitly required.

- Discover and extract data from social media, online news platforms, crowdfunding platforms and other sources of relevant information. This process is referred to as crawling.
- Derive information from the extracted data by parsing text, using methods from data mining and natural language processing and doing semantic analysis in order to enable the subsequent workflow.
- Answer a query by employing the extracted information
- Analyze the similarity between projects
- Compute the projects current popularity.

- Analyze and predict the possible success of a project.
- Rank and present projects accordingly.

4.1 Specification of the proposed System

The initial points of the necessary information are crowdfunding projects and all the corresponding data on the Web, or more precisely data from social media, online news platforms and crowdfunding platform Web sites.

Since the Web is very large, distributed and rapidly growing and changing, one great challenge is to generate and maintain an index of the Web, that is sufficiently exhaustive - ideally covering the entire Web - and sufficiently up-to-date and accurate. Google [1] maintains such an index but there is currently no way to access this index free of charge.
For this project the set of used websites is limited, so on one hand the number of crawled crowdfunding sites is limited and on the other hand also the set of social media sources and news platforms is preliminary defined.
Nevertheless the indexed pages will be changing and growing and therefore the constructed index needs to be updated over time. This process will be separated from the rest of the project, so part of the crawling will not be done for every new query but in a fixed interval. The index construction will still be one of the addressed problems although building the subsystem that feeds raw text to the indexing process will be not as complex compared to full Web crawler projects.
Suitable crawling methodology is highly depending on the web pages, that are being crawled, so the crawler will be pluggable as a separate program part, that can be written specifically tailored to certain application areas and can be plugged into the proposed framework.

The project focuses on giving the optimal results for a query.

The majority of the work is done on the backend respectively in the prediction process. The focus lies on predicting the success of the project through machine learning algorithms and data analysis.
The user interface design and user satisfaction of the prototype system are not

[1]www.google.com

part of the work. Furthermore fast result generation through parallel processing or other methods are not in the focus of the work either.

4.1.1 Potential Users of the proposed System

Potential users of the proposed system can be funders as well as project creators on crowdfunding platforms. Investors need to know how successful projects potentially are. Time is also a factor as some of the rewards that are provided for investors are limited.

The proposed framework is therefore designed to identify potentially successful projects as soon as possible.
Creators of crowdfunding projects are in the need of a system, that can either tell them if the project they are planing is likely to succeed in the current composition or even more help them in finding a good crowdfunding project configuration.

4.1.2 Requirements and Graphical User Interface Design

4.1.2.1 Functional Requirements

The prototype implementation of the proposed system should provide at least the following functionality, that is necessary to realize the use case shown in Figure 4.1.
Firstly the top 50, recently launched crowdfunding projects in the system are presented on a Web page accessible for users of the application. Top 50, here refers to the projects holding the highest scores according to the machine learning model, used to assess crowdfunding projects in the system. The presentation of the crowdfunding projects should include novelty and popularity scores of the crowdfunding project as well as the scoring and other features, that are stored in the system.
Secondly a user should also be able to create a crowdfunding project over a web interface provided by a web service to receive a score for that project.

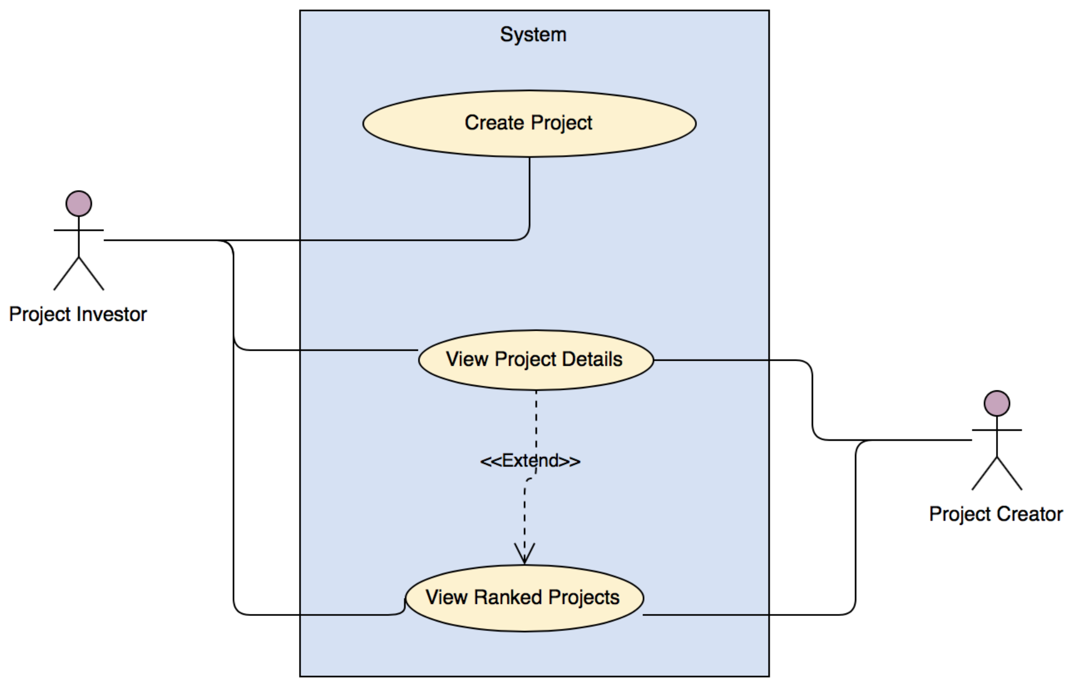


FIGURE 4.1 | **Use Case Diagram.** Use Case Diagram showing the interaction between the users of the prototype system and the system. The three presented Use cases describe the functionality of the system provided to the users.

4.1.2.2 Non-functional Requirements

Data stored in the system should be as much up to date as possible, as projects that are already on crowdfunding platforms for a longer time will forfeit their need for assessment for one of two reasons:

- In case the project looks promising it is already promoted or displayed accordingly on the platforms Web pages or have already done good progress towards their goal.
- Or in contrary, the project does not look very promising as it has little to no backers after that considerable time that already past.

Consequently it is crucial to the application to automatically extract, process, update project data in the system and present the newest projects to the users.

In the light of the fact that the number of new platforms is increasing and online websites are changing frequently, it is important to keep the framework modular to make it possible to adapt to new sources of information and new interfaces to platforms and online media and still be able to work autonomously.

This facilitates reuse of system components and enables the system to be solve other problems apart from crowdfunding.

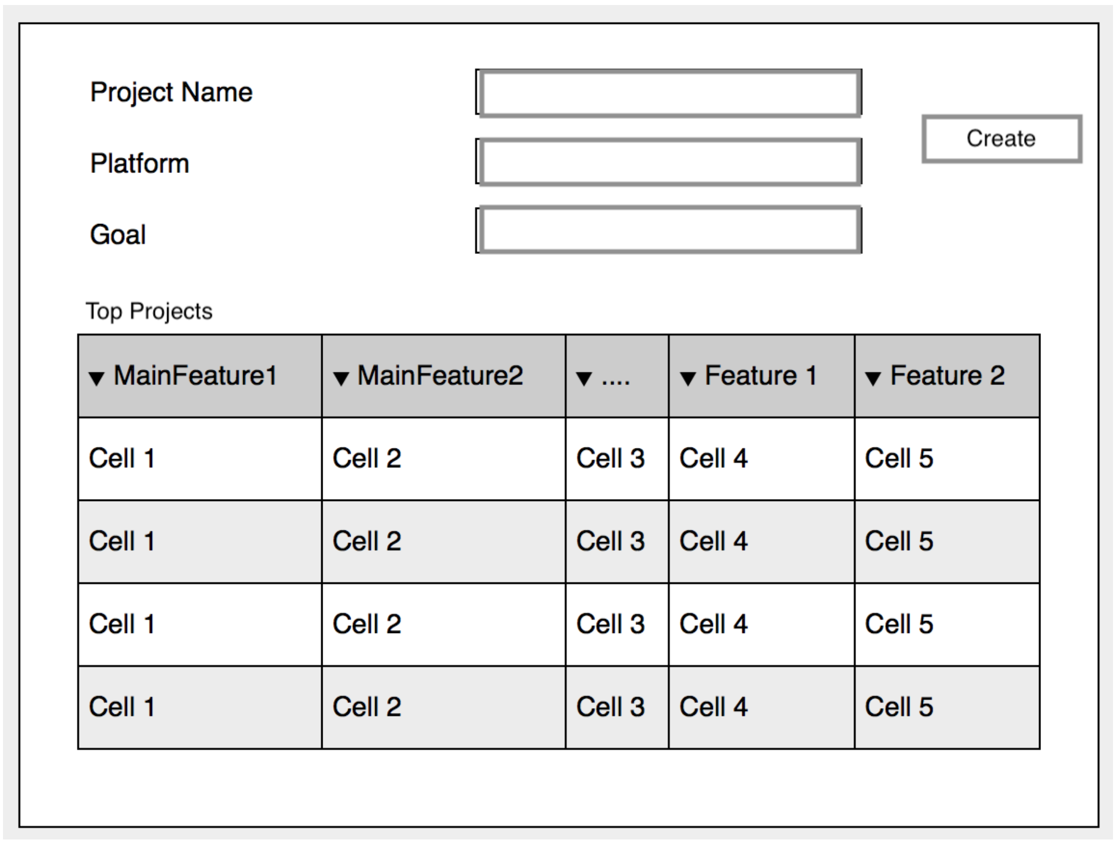
The application shall be scalable to provide the capability to work on big data as the number of crowdfunding platforms and consequently the number of projects is rising. Additionally online media is also rising and it can be assumed, that more information from online media and crowdfunding projects enable better accuracy of the machine learning model.

4.1.2.3 Graphical User Interface

The user interface as shown in Figure 4.2 presents the top 50 highest rated crowdfunding projects currently known by the prototype system. The list is sorted by score automatically and the user can see essential project features right away. These include name, score, similarity score, popularity score and the platform it is launched on. Additional features can be shown by clicking the details button. These features are essentially those that are platform specific or not as relevant to the project as the before mentioned primary features. The user interface is implemented in such a way, that it is capable of presenting an arbitrary number of features. As mentioned earlier on, the system should be modular and extensible and therefore the number of features a crowdfunding platform offers can't be known in advance. Features must be presentable in string format.
The user interface also provides mechanisms to allow the users to create projects.

Required Tasks

To meet the requirements the system must be able to accomplish the tasks shown in Figure 4.6.
Specifically extracting online media data and crowdfunding data simultaneously, performing data analysis on extracted data, computing similarity between projects and similarity between project and online media, update crowdfunding project data frequently, measuring novelty, popularity, and successfulness, taking user input to create projects and presenting data to the user.



Project Name	
Platform	
Goal	

Top Projects

▾ MainFeature1	▾ MainFeature2	▾	▾ Feature 1	▾ Feature 2
Cell 1	Cell 2	Cell 3	Cell 4	Cell 5
Cell 1	Cell 2	Cell 3	Cell 4	Cell 5
Cell 1	Cell 2	Cell 3	Cell 4	Cell 5
Cell 1	Cell 2	Cell 3	Cell 4	Cell 5

FIGURE 4.2 | **Graphical User Interface Mockup for Client Side Presentation and User Interaction.** Graphical user interface accomplishes the presentation of project data to the user and provides the mechanisms to allow for the user to create projects. The user interface presents the top 50 project sorted by the computed successfulness score plus the projects, that were added by the current user over the user interface.

4.2 Architecture and Design of the proposed System

4.2.1 Conceptual Architecture and Logical Design

This section describes the conceptual architecture and logical design of the approach taken, that is modeled in Figure 4.4.

To keep the system flexible and modular the system is further designed to follow the Model-View-Controller pattern as shown in Figure 4.3, where the system is divided into the three components *Model*, *View*, and *Controller*. The *View* is responsible for the representation of the information, while the *Model* stores and manages the data and implements logic. The *Controller* takes input from the user and delegates *Model* and *View*.

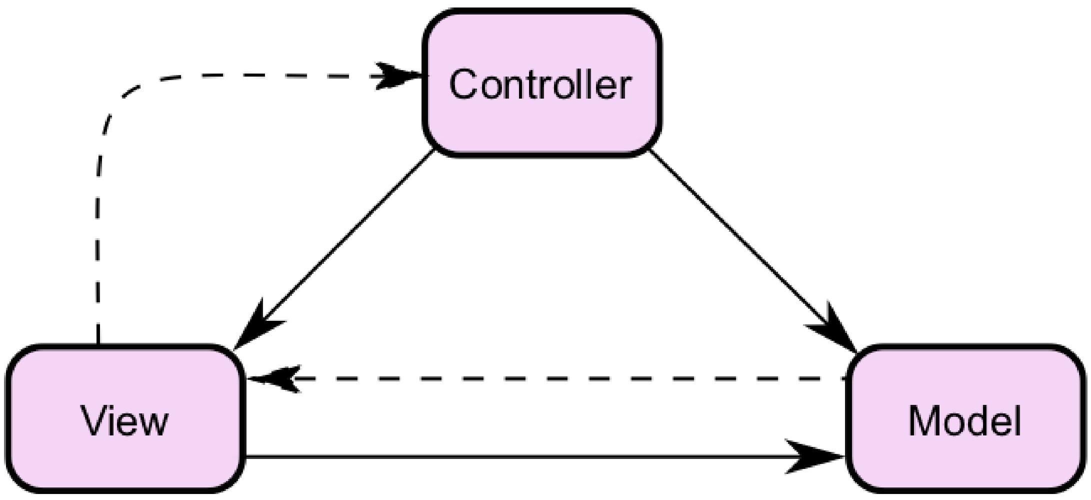


FIGURE 4.3 | **Description of MVC - Pattern deployed in the Prototype.** Figure describes interaction between the components in model - view - controller architecture, that is deployed in the prototype [50]

The center part of the application lies in the *Coordinator* that coordinates data processing steps from crawling over storing and indexing to data analysis steps and the prediction process. It shall coordinate the workflow between the respective components namely *Crawler*, *Indexer*, *Aggregator*, *User Interaction Component*, and *Machine Learning* component and is therefore in charge of the application logic. The workflow will be described later on in this section. The *Coordinator* component also realizes the *Controller* component we saw in Figure 4.3.

The *User Interaction* component, that is equivalent to *View* component described earlier is handling logic for user interaction and presenting projects and scoring results by offering a graphical user interface to web clients.

The *Crawler* provides the methods, that enable the system to retrieve information from external sources. First and foremost these external sources are the crowd-funding hosting platforms for project information, but also social media platforms like *Twitter* and news websites [2]. Individual implementations of *Crawler* classes can be integrated modularly to be able to crawle sources, that vary in it's structure and require different methods to be crawled. Twitter provides an API for instance and shouldn't be crawled the way news platforms or crowdfunding websites need to be crawled.

[2]The list of news websites crawled by the prototype system is provided in the Appendix A.0.1

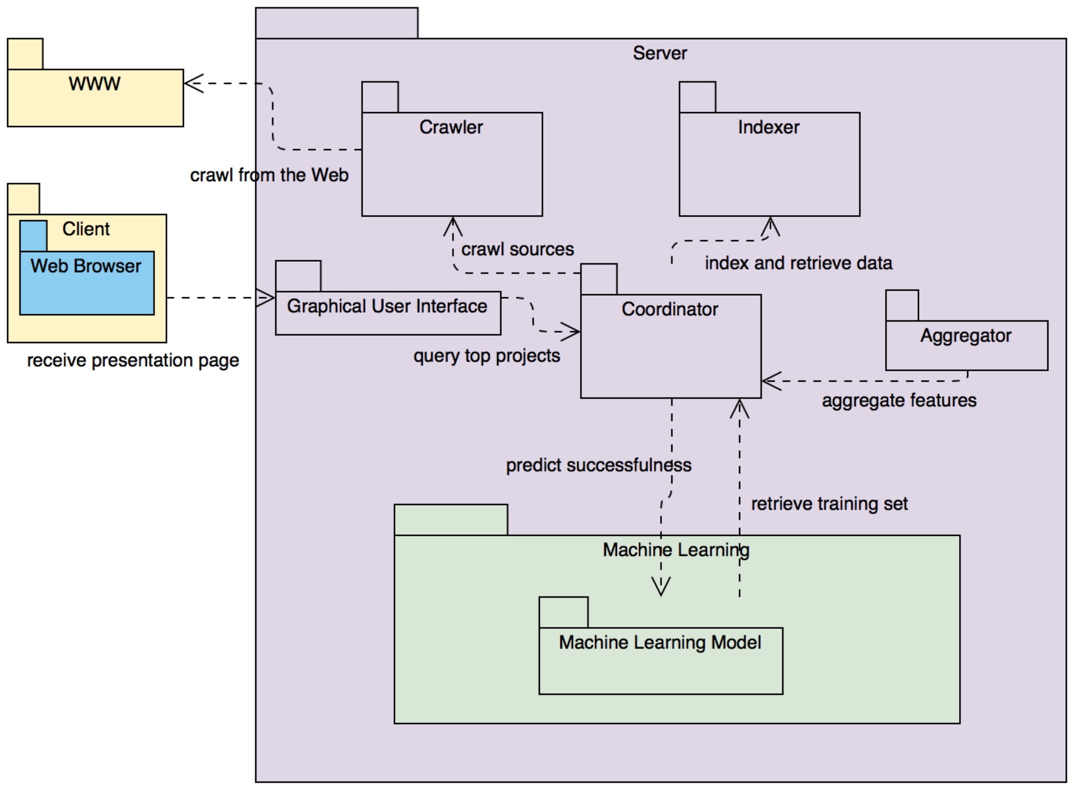


FIGURE 4.4 | **Package Model describing System Architecture of the Prototype.** Figure describes the conceptual system architecture of the proposed Prototype and dependencies between the components.

The *Indexer* is responsible for storing and indexing information in a way it can be accessed in a efficient way. It is realized by an information retrieval system to provide sufficient search possibilities.

The *Aggregator* provides data analysis capabilities as well as the computation of novelty and popularity.

Finally the *Machine Learning* component gathers data to learn or relearn the model that is used to predict project success and labels new data. The system is deployed on a Web server.

4.2.2 System Design and Component Description

Figure 4.5 shows a more detailed architecture of the proposed system.

The *Information Extraction* component is essentially the *Crawler* component we saw in 4.4 but now split up into two parts. One part that is scraping data from

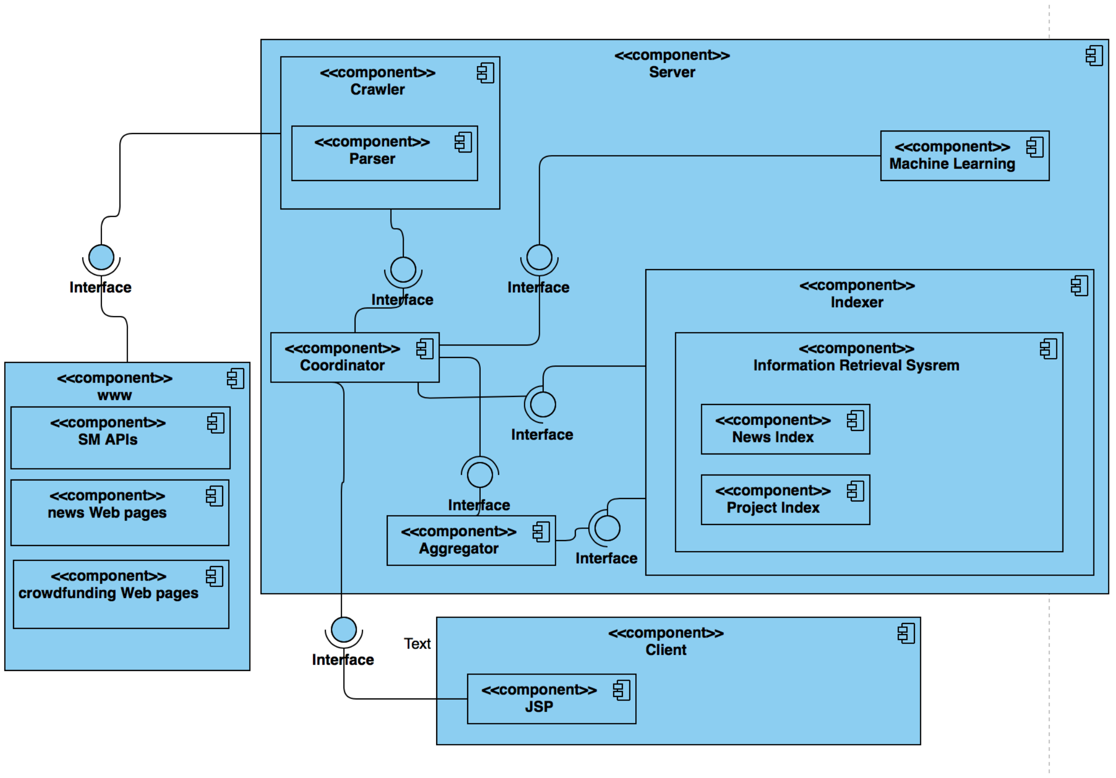


FIGURE 4.5 | **Component Diagram describing more detailed Prototype Design.** Figure describes the Project Design greater detail

online websites or other online sources and a second part that extracts content from the data.
This includes parsing textual data as well as extracting metadata from documents, aggregating data and data analysis like sentiment definition for certain texts. In addition new search URLs and documents to be fetched have to be identified and sent back to the Crawler.
The crawling process itself is independent from the rest of the workflow and is pluggable to adjust to different websites and services. The sources of information are social media, news articles from certain domains and crowdfunding websites. Extracted data is not used to answer user queries directly but to be processed by machine learning algorithms and analysis methods that result in the computation of novelty, popularity, and in the prediction of successfulness.

The challenges in modeling successfulness are (1) finding a similarity model, that indicates the degree of novelty on project index, (2) analyzing news and social media data to compute popularity using statistics, sentiment analysis and prediction methods, and (3) extracting and using project information like country etc... To solve these challenges, different solutions have to be considered.

To measure similarity text similarity measures are being used whereas a model integrating additional extracted feature data from online news and social media is constructed to measure the projects popularity. Machine learning algorithms then run on all that information to realize project success prediction.
Two indices are being created in order to run text similarity functions on the data. One storing crowdfunding projects and another one storing online news data. In the prototype implementation these indices store all available data.

Both indices are queried with crowdfunding project names. The computation of scores will be described in detail in Section 4.2.5. Computed data is stored into the index from where the prediction component takes hold of the data to generate scores. Combined data is then utilized by the User Interface component to be presented to users.

4.2.3 Process Flow in the System

The process flow in and between the components is further described in Figure 4.6.
Crowdfunding project data and online news data needs to be crawled, stored, and indexed simultaneously to create a base index. This has to be done beforehand, in a pre-run before the system can start to work on scoring projects. Crowdfunding projects from the past are also valid at this stage. So as long as the system is in the pre-run state, it keeps looping trough the described process flow.
After some the system finishes the pre-run. The system keeps crawling the same kind of data but then for each crawled project, respective data from social media is retrieved. Additionally data analysis will be performed and popularity and novelty values are computed for the project. Prediction of successfulness will also be established at this stage.
This includes training the machine learning model that is used for prediction with projects that are already labeled. All available data at this point will be stored again and is presented to the users. The same process applies to projects, that were not crawled but were added to the system by users via the provided graphical user interface.

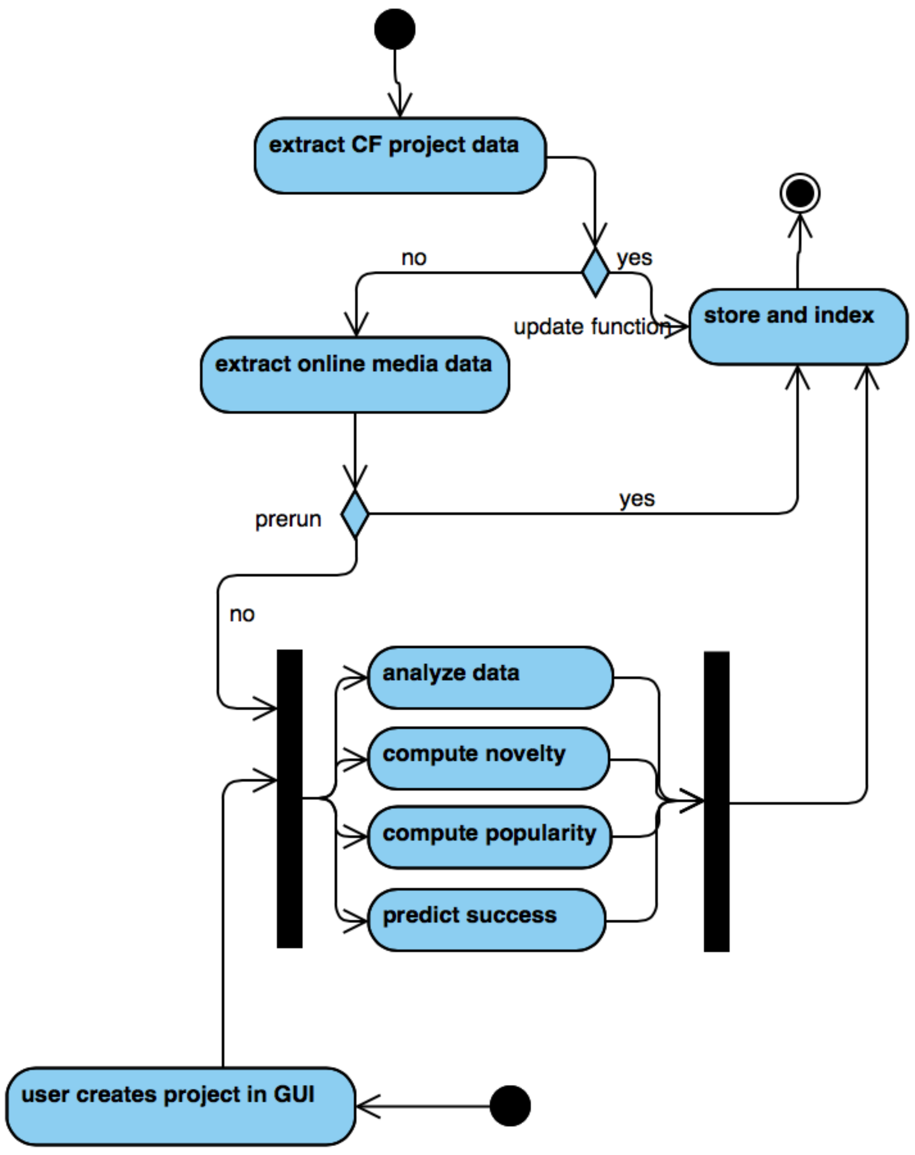


FIGURE 4.6 | **Background Tasks in the System.** Figure describes the background tasks in the prototype system modeled in an activity diagram. There is a distinction between pre-run, update function and the standard workflow that is reflected in the decision nodes. In the pre-run the data base has to be set up. The update function solely updates project data.

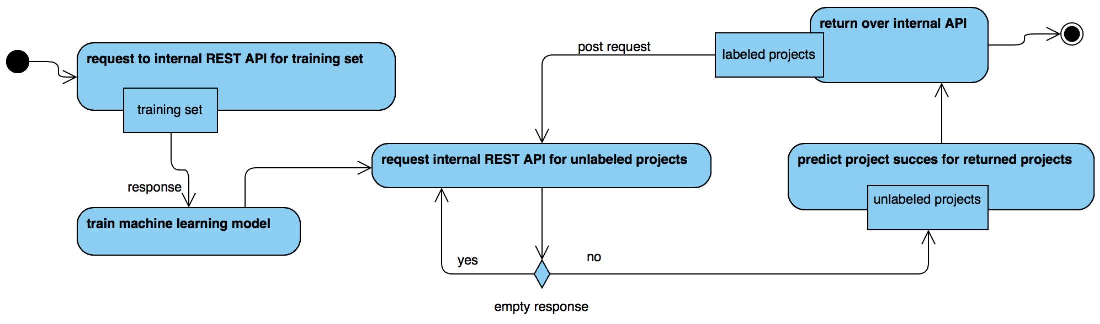

request to internal REST API for training set
training set
response
train machine learning model
request internal REST API for unlabeled projects
yes
no
empty response
post request
labeled projects
return over internal API
predict project succes for returned projects
unlabeled projects

FIGURE 4.7 | **Activity Diagram of the Machine Learning Component Workflow.** Figure describes the workflow in the machine learning application that interacts with the Coordinator component via REST interface.

4.2.4 Machine Learning Component Description and Process Flow

The *Machine Learning* component is modular as it communicates over a RESTful interface with the rest of the system. The process inside the machine learning component is further described in Figure 4.7

On start up it calls the URL for receiving the current training set. This set includes all projects, that are already labeled through an update function. This function checks all indexed projects periodically for changes in the pledged amount and additionally sets the project's label for successfulness in case the deadline is over to true or false.
As new projects come into the system the machine learning component has to update the training set and retrain the machine learning model from time to time. So after having trained a model for the first time the application waits for notifications by frequently calling an URL within the application. The RESTful service will return HTTP status 204 code, signaling no content as long as there is no data to be labeled in the application. Whenever the response contains content, the machine learning component predicts labels for the projects. More precisely project success for all projects in the list that has been received over that URL is predicted.
Finally the projects in conjunction with the newly added labels are returned to the RESTful service.

4.2.5 The developed Novelty and Popularity Models

This section deals with the process of generating scores describing novelty and popularity of a project to the user.
Supervised machine learning is not suitable at this stage of the application, as this would require a lot of historical data and more importantly class labels for novelty and popularity are not available, because there is no universal definition of novelty and popularity for crowdfunding projects as described in the respective background section 3.1.
These terms were only loosely defined in the introduction section as from the view point of the user it is somewhat clear what they mean. But for our model the terms and some implications will be further defined now. Keep in mind, that the scores for these terms are intended to be presented to the user, to get a sense.
Nevertheless correlation between novelty and project success is described in [11] and respectively in [12] regarding popularity and project success. Consequently these scores might as well improve the project success prediction.
The model for prediction of novelty and popularity therefore is theory-based in contrast to the data-based model used in the machine learning component. This leads to a more lightweight application that is less data hungry and most importantly allows faster decisions on smaller amount of data. In the Cambridge Dictionary [51], novelty is described as "the quality of being new and unusual [51]". As already described in Section 1.2 our novelty model is therefore based on the assumption that unusual projects are the ones that are unique in that there are very little similar projects around. To measure similarity between two projects, they can be treated as documents in an information retrieval system. Similarity is then computed over the implemented similarity model.

The model describing popularity is more complex and some situations have to be neglected, which is fine for simple models. Project popularity as "the fact that something or someone is liked, enjoyed, or supported by many people [52]" shall be measured by its support and frequent occurrence of topic or idea of the project in online media.
As such occurrences may impact project success differently, they have to be weighted. Documents from online media sources are weighted equally as all sources are manually selected and should ideally represent high quality media sources. Reputable media platforms are important for their value as information filter and

information quality assurer and can therefore be regarded as containing qualitative as well as well-balanced information as compared to the average social media information. Again text similarity from the crowdfunding project to online news articles will be measured and stored as the *newsScore* for a document.

The prototype system uses Lucene indexes along with standard Lucene scoring functions described earlier in 2.2.1 to find similar projects or respectively related online news articles in the corresponding index. The prototype searches for the top 5 most similar entries and sums up the respective ranking scores. This applies to *NoveltyScore* and *NewsScore*. Lucene utilizes some useful analyzer techniques like synonym expansion, where words are expanded with additional similar words, for matching. This can help in searching for general ideas. The standard similarity algorithm used in Lucene is essentially TFIDF similarity which we get our scores from. Twitter also uses Lucene to retrieve tweets [17].

NewsScore will be refined with data from Twitter like Asur et all [8] do by searching for movie keywords on Twitter to predict popularity of movies. The proposed approach though goes beyond that, by adding online news data as already stated and data analysis on top to measure popularity. Relevance scoring for matching Twitter documents is calculated by taking the number of retweets (Cr), the number of times the tweet got favorited (Cf), the length of the tweet (Tl), and the analyzed sentiment of the tweet into account.

The above-mentioned values are weighted differently as they are expected to affect popularity unequally. Shorter texts are penalize because we assume that they are easier to match in the search process as they contain more potentially matching phrases. So the multiplicative inverse of the text length is used in our model. [46] states that "Indegree represents a user's popularity, but is not related to other important notions of influence such as engaging audience". Indegree denominates the number of followers here and is considered not to be important to be influential or to be representing popular views. Therefore we don't include the number of followers in our model. More importantly they found that "Retweets are driven by the content value of a tweet [46]", which indicates the importance of the number of retweets for our model. [32] investigated "favoriting" on Twitter and found that it is being used similar to retweeting but often also used as a bookmarking function, so in our model the impact of favorited tweets should not be modeled as big as for retweets. Consequently favorites shall be given higher impact than standard tweets and retweets shall be even more influential, so the value of each retweet is modeled to be 15 times higher than the tweet itself, favorites are valued 10 times the value of a single tweet in our model. Each extracted tweet is enriched with a sentiment score between 0, signaling very negative tweets, and 4, signaling very positive tweets. Therefore a value of 2 describes neutral tweets and -1 is assigned to unclassifiable tweets [53]. [44] found out that negative sentiment in tweets seem to have impact on the popularity of that tweet. It can be deduced that such tweets will have a higher number of retweets because of their negative sentiment rather than because of the topic or product they relate to. So in order to counterbalance this issue negative sentiment tweets have to be penalized. Further, as popularity describes something that is being liked and enjoyed, the more positive a tweet is classified the more impact it can have on popularity. This is modeled in function 4.1

$$f(x) = 1/4 * x * (x + 1) \tag{4.1}$$

where the function is defined for integer values in the interval between [-1,4], which correspond to the Twitter sentiment values. In 4.8 we can see that this function applies zero values to undefined (-1) and very negative tweets (0) and is then increasing according to the mentioned assumptions.

$$x \in Z[-1, 4] \tag{4.2}$$

By adding that transformation function to the model, we arrive at the following formula:

$$TwitterScore = (log(Cr) * 15 + log(Cf) * 10 + log(\frac{1}{Tl})) * f(S) \tag{4.3}$$

The logarithm to base 10 is applied here to guarantee the contribution of each score as equal to the final score as we have different standard deviations in different variables. We want to give equal weight to all features before we start adding our own weights according to the mentioned assumptions, hence transform absolute change in variables into relative [54].

Finally *NewsScore* and *TwitterScore* are added up in Equation 4.4

$$PopularityScore = log(NewsScore) + log(TwitterScore) \tag{4.4}$$

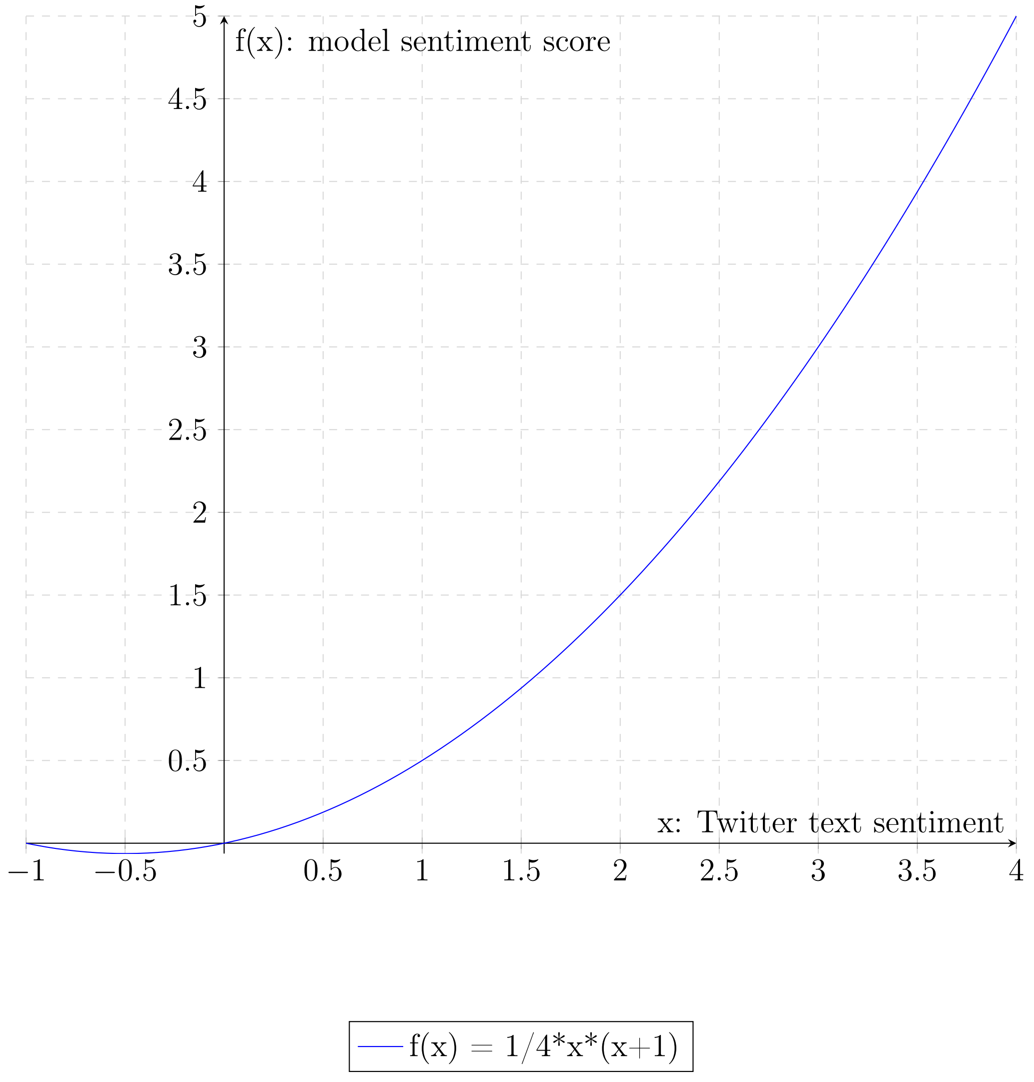


FIGURE 4.8 | **Sentiment Transformation Function.** Plot of the function that transforms Twitter sentiment data into model sentiment values, that are essential to Twitter score and respectively Popularity score calculation.

and further simplified to Equation 4.5

$$PopularityScore = log(NewsScore * TwitterScore) \tag{4.5}$$

that gives us the *PopularityScore*. [55]

Chapter 5

Implementation and Technologies

5.1 Java Class Design of the System

The prototype system consists of a *Model* package and a *Controller* package, that contains *Indexing*, *Crawl*, and *Aggregator* packages.

Classes in the *Model* package are representations of data in the system, specifically crowdfunding projects, and online media objects such as Tweets or news articles and can be seen in Figure 5.1.

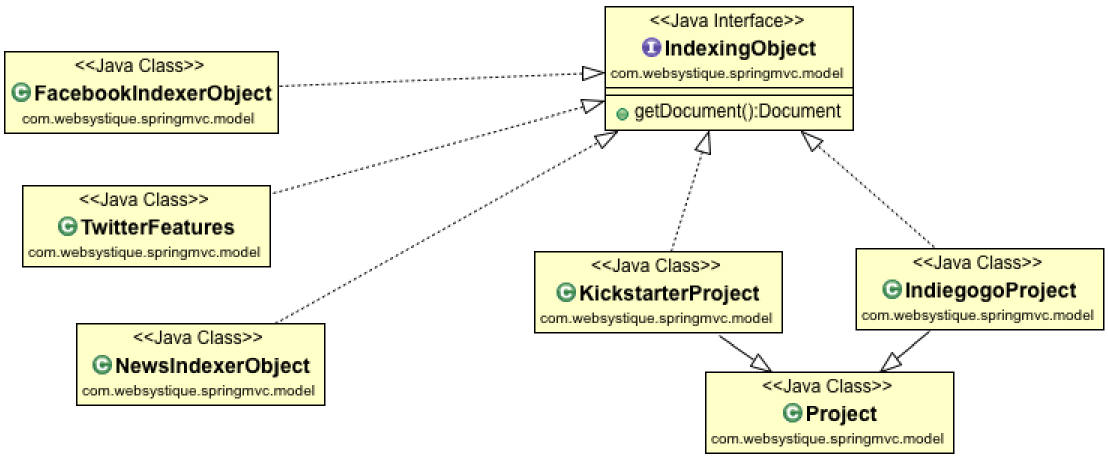


FIGURE 5.1 | **Class Diagram illustrating the *Model Package*.** Figure illustrating the *Model package, that contains classes to represent data in the system. Besides that it provides a superclass and an interface to ensure interaction with Indexer package for current and future implementations of model classes.*

```java
import org.apache.lucene.document.Document;
public interface IndexingObject {
%   public abstract KickstarterProject(Document doc)
    public abstract Document getDocument();
}
```

Listing 5.1: Superclass that needs to be implemented by all model classes in order to ensure interaction with Apache Lucene

They are partly created by an online application called *jsonschema2pojo* that, as the name suggests, turns JSON objects into Java classes.
Model classes have to implement the *IndexingObject* interface shown in Listing 5.1 to make sure data can be stored in the information retrieval system.
Further all classes holding crowdfunding data - in the prototype these are *KickstarterProject* and *IndiegogoProject* - need to extend superclass *Project*, that stores platform independent features common to all projects like the score for instance. This demands them to implement the Java interface *java.lang.Comparable* to be able to realize a sorting functionality, that is required to present the top 50 projects at a time to users.

The *Indexer* package is responsible for access to the Lucene index and therefore contains all retrieval and storing methods.

The *Crawl* package contains all classes responsible for collecting relevant data from sources. In the prototype these sources are news websites, Twitter feeds, and online crowdfunding platforms. Each source had to be implemented in a different way further described in the data extraction section 5.6.1.
The implementation for crawling online media sources was based on Apache Stormcrawler. This meant to implement *ParserBolt* and *IndexerBolt* classes. These classes are responsible for parsing crawled web pages in order to extract and prepare data for the storage in the Lucene index. *ParserBolt* basically extracts contents and metadata with the help of Apache Tika while *IndexerBolt* communicates with the index by sending *NewsIndexerObjects* to the *Indexer*.

Interaction between Java Classes of the Prototype in the Process of Extracting and Assessing Crowdfunding Projects The interaction between Java classes of the prototype in the process of extracting and assessing projects is illustrated in Figure 5.2.

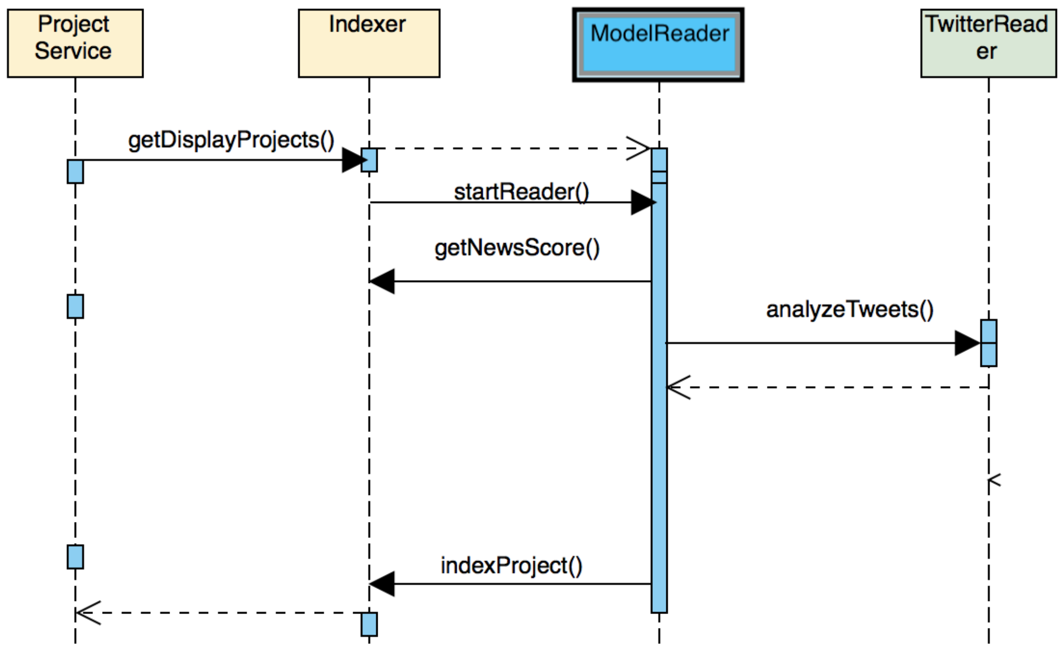


FIGURE 5.2 | **Sequence Diagram.** Figure describing part of the message flow between Java classes in the process of extracting and assessing projects.

The *ProjectService* class is part of the *View* component in the Model-View-Controller Architecture and therefore calls the *Indexer* class for data to display.

The *Indexer* class enables access to all stored projects and their respective data. The *ProjectReader* is started to crawl new projects and calls *TwitterReader* and the *Indexer* to add computed features to the project. *TwitterReader* returns Twitter features and *TwitterScore* while *NewsScore* comes from the index, that calls the respective similarity functions from the information retrieval system as described before.
Finally *NewsScore* and *TwitterScore* are aggregated to a *PopularityScore*. *NoveltyScore* and *PopularityScore* will then be further aggregated as well and stored in the index together with the project data. From here the project is ready to be scored by the machine learning component and to finally be presented to the user.

```java
//abstract crawler to inherit for crowdfunding project pages
public abstract class ModelReader {

    //run through all projects to update pledged
    public abstract void updateProjects();

    //read Projects and add to project list
    public abstract void retrieveProjects(final List<Project> l);
}
```

Listing 5.2: All classes that read from crowdfunding platforms to crawl projects must subclass the abstract *ModelReader* class

5.2 Extending the Framework

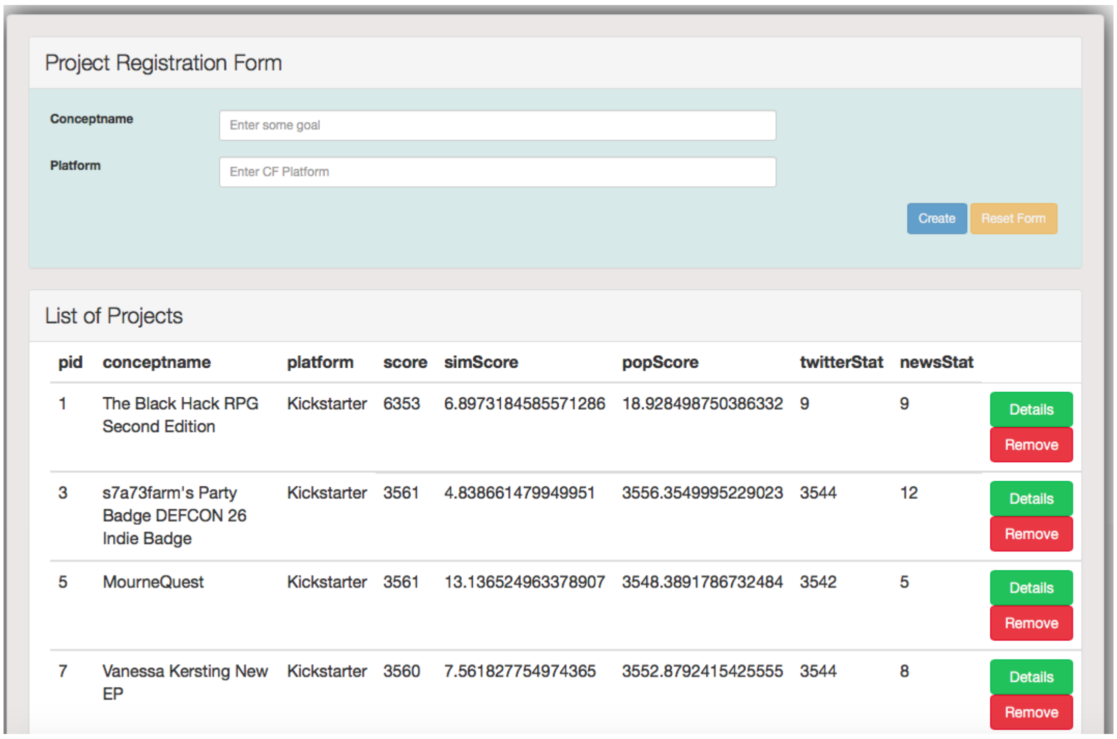
Model classes can be added to the framework as long as they implement the *IndexingObject* and respectively extend parent class *Project*. This satisfies a condition arising from the fact, that data is stored persistently in the index. Namely the condition, that an object can be created from the index through an appropriate constructor. This constructor should be overwritten of course when subclassing the *Project* class.
To collect data one has to write a reader class by extending the abstract class *ModelReader* provided in Listing 5.2.

Lastly the machine learning model has to be enhanced accordingly, in case new features should be integrated into the prediction model.
To utilize newly integrated model features it is necessary to adapt the machine learning model accordingly.

5.3 User Interface Implementation

The user interface as it is implemented in the current prototype application is shown in Figures 5.4 and 5.3. It presents a list of the top 50 ranked projects in the system, sorted by successfulness score. Automatically project features are presented for all projects, whereas detailed information presentation is activated by pressing a *Detail* button.

Detailed information can vary of course depending on the kind of source the project came from, but the GUI will adapt automatically by generating column names and respective data values on the fly, reacting to the type of project that is represented 5.3.



Project Registration Form

Conceptname	Enter some goal
Platform	Enter CF Platform

List of Projects

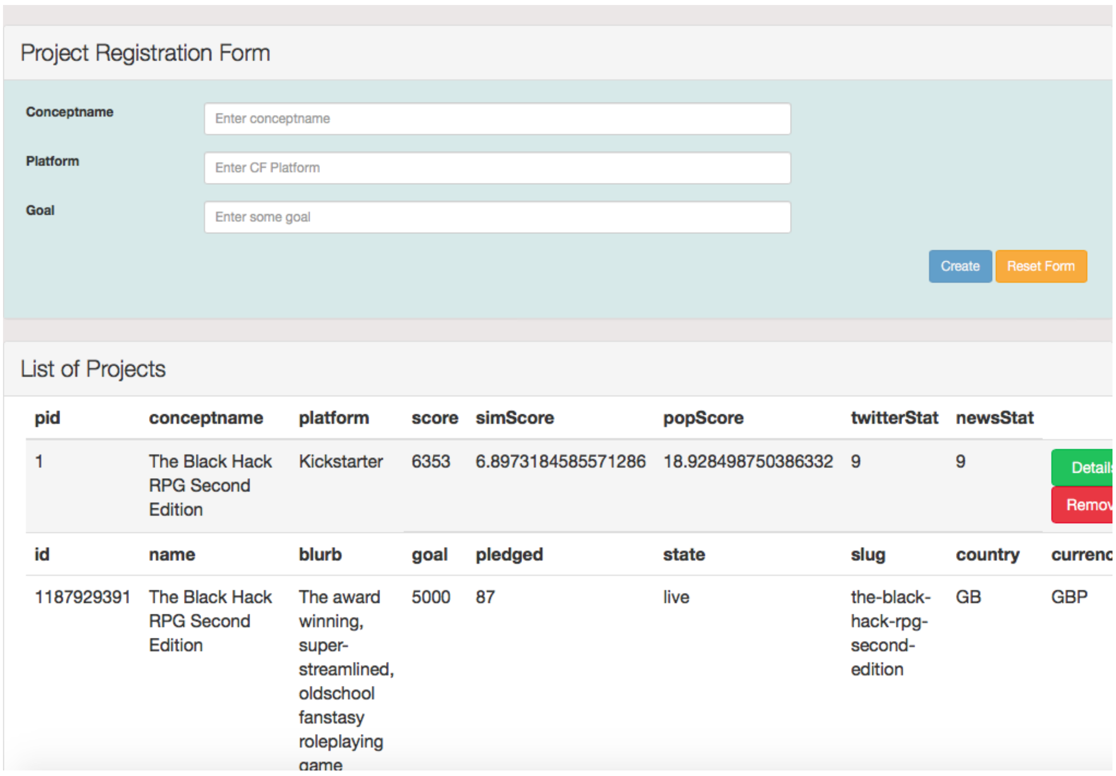
pid	conceptname	platform	score	simScore	popScore	twitterStat	newsStat	
1	The Black Hack RPG Second Edition	Kickstarter	6353	6.8973184585571286	18.928498750386332	9	9	Details Remove
3	s7a73farm's Party Badge DEFCON 26 Indie Badge	Kickstarter	3561	4.838661479949951	3556.3549995229023	3544	12	Details Remove
5	MourneQuest	Kickstarter	3561	13.136524963378907	3548.3891786732484	3542	5	Details Remove
7	Vanessa Kersting New EP	Kickstarter	3560	7.561827754974365	3552.8792415425555	3544	8	Details Remove

FIGURE 5.3 | **Graphical User Interface - Presentation and Registration.** Figure shows User interface with the list of indexed projects and their main features. Further a registration form is provided at the top to provide the possibility for users to add their projects to the system and receive scoring and presentation in the ranked list.

On top of the page there are three text fields to enable users to inject their own projects into the system. Project name, platform and funding goal are requested in order to create a project over the GUI. From these features a project will be created internally and all scores will be computed. Finally the new project is added to the list of projects in the GUI, which is realized through JSPs and the AngularJs framework. Client - server communication happens over the internal REST interface.

5.4 General Implementation Details

The prototype is published on an Apache Tomcat 9 web server and all data is stored and indexed in an Apache Lucene index. The superior solution would be to store data in a relational database and make an index over it, but this step is excluded to simplify and speed up the implementation.



Project Registration Form

Conceptname	Enter conceptname
Platform	Enter CF Platform
Goal	Enter some goal

List of Projects

pid	conceptname	platform	score	simScore	popScore	twitterStat	newsStat	
1	The Black Hack RPG Second Edition	Kickstarter	6353	6.8973184585571286	18.928498750386332	9	9	Detail Remov

id	name	blurb	goal	pledged	state	slug	country	currenc
1187929391	The Black Hack RPG Second Edition	The award winning, super-streamlined, oldschool fanstasy roleplaying game	5000	87	live	the-black-hack-rpg-second-edition	GB	GBP

FIGURE 5.4 | **Graphical User Interface - Project Details.** Figure presents user interface showing full set of project features after detail button is pressed for project *The Black Hack RPG Second Edition"*

The web server provides a REST interface, that processes requests sent by a Client Application realized trough JSPs and is also used for inter process communication inside this application.
As the machine learning component is implemented in Python there must be an interface to communicate with the rest of the program that is written in Java. The REST interface could as well be used to provide an API to other applications not running on the same server. All requests are stateless and return JSON data.

Data representation in the JSP is realized with the help of AngularJS which is a Google framework based on jQuery and implemented in JavaScript. It is intended to build Web pages and applications implementing the MVC design pattern [56]. There are newer versions of this framework available but at the time of implementation AngularJS seemed to be better documented due to being available for a longer time.

Java is the chosen language for the whole system implementation with the exception of the machine learning component. The programming language bears the advantage of speed, scalability, and robustness. Java is also easily enhanced with

```
<div class="panel panel-default">
<!-- Default panel contents -->
<div class="panel-heading"><span class="lead">List of Projects </span></div>
<div class="tablecontainer">
<table class="table table-hover">
<thead>
<tr >
<!-- on detail button click show additional features of the project - first
    create column names then values -->
<th ng-repeat="(key, value) in ctrl.projects[0]" ng-init="pindex =
        ctrl.getIndexofProject(ctrl.projects[0])" ng-show = "pindex <= $index">
<span ng-if = "pindex <= $index" ng-bind="key">{{key}}</span> </th>
</tr>
</thead>
<tbody ng-repeat="u in ctrl.projects" ng-init="pindex = ctrl.getIndexofProject(u)">
<tr>
<td ng-repeat="(key, value) in u" ng-show = "pindex <= $index">
<span ng-if = "pindex <= $index" ng-bind="value">{{value}}</span>
</td>
<td>

<button type="button" ng-click="hideShow=(hideShow ? false : true)"
    class="btn btn-success custom-width">Details</button>
<button type="button" ng-click="ctrl.remove(u.id)" class="btn btn-danger
   custom-width">Remove</button>
</td>
</tr>
<tr ng-if="hideShow" ><th ng-repeat="(key, value) in u"  ng-show = "pindex > $index">
<span ng-if = "pindex >$index" ng-bind="key">{{key}}</span>
</th></tr>
<tr ng-if="hideShow"><td ng-repeat="(key, value) in u"  ng-show = "pindex > $index">
<span ng-if = "pindex > $index" ng-bind="value">{{value}}</span>
</td></tr>
</tbody>
</table>
```

LISTING 5.3: Generic Buildup of the Project List in the User Interface implemented in *ProjectManagement.jsp*

a broad variety of libraries over Apache Maven. Also Apache Lucene, Apache Tika - used for metadata extraction in the prototype - and many other Apache applications are written in Java and provide a native API. It is ideal for complex robust applications. It is designed to develop secure, high performance, and highly robust applications according to [57] and seems to be a good choice for complex applications.

5.5 Discussion of Technical Implementation and Technologies

5.5.1 Information Retrieval Technologies and Technical Implementation

This section describes the decision about how to realize the information retrieval system for extracting and scoring relevant information about a crowdfunding projects. Final candidates for implementation were Apache Solr [1] vs Apache Lucene [2] vs Elasticsearch [3]. Solr, similar to Lucene and ElasticSearch is an open source search platform while Elasticsearch additionally provides an analytics engine.

The standard way to use Solr is via Solr's standard HTTP interfaces. Embedding Solr is less flexible, harder to support, not as well tested, and should be reserved for special circumstances.

> "If you need to embed search functionality into a desktop application for example, Lucene is the more appropriate choice. For situations where you have very customized requirements requiring low-level access to the Lucene API classes, Solr may be more a hindrance than a help, since it is an extra layer of indirection [58]".

If Solr crashes, there are chances of getting downtime for the application. So its always better to run Solr independently. Embedding Solr is also not recommended according to [59].

Elasticsearch can be accessed over a REST API and like Solr provides a convenience layer above Lucene using Lucene itself, but is therefore not as flexible and customizable compared to using Lucene directly. It is designed to work in a distributed environment and provides many features to realize its functionality in such an environment, but a distributed system is not required in our use case. Lucene on the other hand offers Maven support and is faster as it provides a direct Java API using function calls and can consequently be easily integrated into the

[1]`http://lucene.apache.org/solr/`
[2]`https://lucene.apache.org`
[3]`http://elastic.co`

prototype system. Low level API access enables direct access to internal functions using similarity scoring models, that will be used in the prototype. Therefore Lucene is the best pick here as we have a use case, that is not well suited for Solr or Elasticsearch.

5.5.2 Crawling Technologies and Implementation

Crawling is required in the proposed system, as services like Google's News API [4], Microsoft's Bing API [5] and other similar API's are either deprecated or liable to pay costs.

The decision about the used crawling technology finally included three contenders, namely Apache Nutch [6], Apache Stormcrawler [7] and the Java library Jsoup [8].
Apache Nutch is a fully implemented online crawler that is intended to provide a REST service to access automatically crawled data. Unfortunately it is designed to work with Apache Solr as opposed to Apache Lucene that was chosen as the information retrieval component.
It comes without a Java API or any straight forward way to use low level functions or possibilities customize the system apart from standard built-in customization methods. It needs to be set up separately and the access over a Web service again is very slow and limited compared to direct access inside of the application.

Jsoup is a very popular library for web scraping, but hardly provides very little sophisticated functionality compared to systems like Nutch, that offer automatic data storage, crawl databases and obeys robot.txt files just to name a few. Some functionality can be outsourced to other libraries like crawler-commons [9], that facilitate robot.txt parsing and rule-set creation and other tasks related to crawling. So it might be a good choice for scraping of single web sites, that are not JavaScript based, but it is not as sophisticated for online crawling.

Stormcrawler is more similar to Nutch in its functionality, while being intended to be integrated over Java classes and methods and is therefore highly customizable in the way it crawls websites or stores data.

[4]https://developers.google.com/news-search/
[5]https://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api/
[6]`http://nutch.apache.org`
[7]`http://stormcrawler.net`
[8]`https://jsoup.org`
[9]`https://github.com/crawler-commons/crawler-commons`

Selenium, a headless browser that is capable of scraping JavaScript websites, is implemented in Stormcrawler. Further it has to be considered that, using Selenium to process Ajax calls is really slow performance-wise and demanding resource-wise, because it uses a browser to navigate trough web sites.
Unfortunately Stormcrawler's standard implementation doesn't work for the Ajax driven Kickstarter and Indiegogo sites even though having Selenium support, so in this case a tailored solution is needed.
The module to crawl from Kickstarter finally was written with Jsoup after reverse engineering relevant URLs from the webpage by catching XHR requests with the help of Firefox Developer Edition.

The prototype uses Jsoup on the one hand for implementing scrapers specifically designed to read data from Ajax based crowdfunding websites like *Kickstarter*. On the other hand Stormcrawler is used for crawling online news media platforms.
Stormcrawler has to be customized to extract the metadata needed and to store data to the index. It is configured to ignore links to Web pages outside of the currently crawled domain. StormCrawler already implemented methods to follow the guidelines from robot.txt files that contain information for crawlers like crawl delays and valid URLs on this domain, but for crowdfunding project extraction with Jsoup this has to be implemented additionally.

5.5.3 Inter Process Communication Technologies and Technical Implementation

As seen in 4.2.1, in this application there is a need for communication between certain components of the application, namely *User Interface*, *Machine Learning* and the rest of the application.

The decision on how to handle inter process communication on the prototype server fell on RESTful webservices. Other options like using pipes or a storage layer in between lack the universality in implementation compared to REST services that rely on well established HTTP protocols. They can be implemented in any language on any system obeying these protocols and therefore also secure future implementation of additional system parts or changes of system parts.

5.5.4 Machine Learning and Data Analysis Technologies and Technical Implementation

Python was used as the programming language for the *Machine Learning* component as it is widely used in data science and therefore most of the newest libraries and technologies are designed to be elaborated in Python. Other upsides are fast implementation possibilities and the fact that it is widely usable as a general programming language, also supporting data structures commonly used in R like data frames.

Keras, Tensorflow and XGBoost are powerful machine learning libraries that natively support Python. They represent current state of the art in machine learning, are well documented and have big communities behind them. Keras on top of Tensorflow enables to work on a different abstraction layer and provides a simplification for building models with Tensorflow. Tensorflow operates beneath it as a machine learning library specifically designed for neural networks. XGBoost is a scalable tree boosting system using minimal resources [36].
Besides Tensorflow and Keras and XGBoost, the component uses some functions from *skicit-learn* for data preprocessing, which is a machine learning library for Python. The *Machine Learning* component is implemented in Python using Jupyter [10] notebooks.

To decide on which classifier to use essentially three perspectives have to be taken into account.
The complexity, respectively the dimension of the problem, the size of the dataset, and the processing effort for learning the model and for the classification process. Additionally, there are so called black box algorithms as opposed to white box implementations. The difference lies in the way the trained model is represented. For some classifiers one can draw conclusions from the way the model is built. In a decision tree for instance, one can observe the decisions made on each node. Other classifier like the neural network give no hint on how the internal decision process is modeled and no assumption can be made upon.
In this application a neural network was chosen as one candidate to use as the classifier, as this is straight forward to do with Keras and Tensorflow, leverages the power of those libraries, and adjust to many different learning problems. Secondly

[10]http://jupyter.org

the XGBClassifier from the XGBoost library was a candidate, as it is a white box algorithm.

5.6 Data Processing and Machine Learning Model Design

5.6.1 Data Extraction

As described earlier in 4.2.1, the architecture for data extraction is structured modularly, so new sources of data can be added without major adjustments to the system interfaces. But most of the time it will be necessary to adjust the extraction process to the source of data, which is unavoidable.

The prototype fetches data from different online sources namely Kickstarter, Indiegogo, Twitter, and various online news platforms listed in A.0.1. Facebook data was also included at the beginning, but had to be left out in the end for being to restricted in terms of access.

Data is crawled continuously from online news and crowdfunding platforms. Additionally crowdfunding project data can be added to the system by a user over the *User Interface*. Based on the crowdfunding project, that is subject to analysis, relating data will be gathered from Twitter.
Extracted data will be utilized for computing novelty, popularity and successfulness of crowdfunding projects and to be presented in the *User Interface* of the prototype.

5.6.1.1 Data Extraction from Online Media

Gathering relevant data on the fly was not possible as the relevant APIs are not receptive for these kind of calls. In the best case, thinking of Twitter, relevant data can be accessed by commercial users. In the case of Facebook it can not be accessed at all. Crawling data from the whole web on the fly would definitely take to long and pre-indexed systems like Google are currently not providing free APIs.

Online Media Data used in the Prototype and how it is brought into the System

- Facebook data was still part of the framework in the early stages of project implementation. Facebook provided an API to make public data available, whereas private information was not fetchable for obvious privacy reasons. Furthermore it was not possible to search for certain posts.
  Consequently public posts were taken from public groups on Facebook. The application took data from the most common facebook groups that deal with crowdfunding.
  Another source of information was taken from public profile information like the number of friends a facebook user is connected to. Specifically the owner of the crowdfunding project and the number of his friends were were observed, as this would potentially correlate with the speed a crowdfunding project goes viral after the owner of that project shares it on facebook.

  So data from Facebook was very limited as it was restricted to publicly available data in the first place.
  After the latest events regarding Cambridge Analytica and the following data policy refinement [60], it is nearly impossible to retain useful data from Facebook without publishing a full Facebook application that gathers data for you and even then it has to be worked out how this would be possible in a legal way.
  User related data whatsoever is now no longer accessible without respective user credentials and is therefore left out of the project. Not even public data or posts from public groups are accessible.

- Twitter provides two kinds of APIs: A streaming API, that provides a stream of all current data, and a search API, that approximately goes back seven days according to [61], but can't handle the same amount of data.
  The key difference between the two lies in the query terms and data limits. The streaming API has no limitations on data, but one cannot change the query string frequently. For the search API in contrary the amount of data is the limiting factors.
  So for the streaming APIs it would not be possible to query for different projects within the query limit. A solution would be to use a datastore and index the full Twitter data but this would blow up the whole application in size by several magnitudes. Solely keeping data related to crowdfunding

would minimize the size the data to store in the system, but would leave out all information not related to crowdfunding on the other hand. Therefore the prototype calls Twitter's search API and modifies queries to with respect to the crowdfunding project to score.

Usable data on Twitter includes the tweets text, the number of retweets, number of likes, number of followers, number of comments and also aggregated data like the number of tweets for a certain query.
The used Twitter search API allows for 450 queries per window, which is a period of 15 minutes. Each query can return up to 100 tweets, which adds up to a maximum of 45000 tweets per 15 minutes. It is possible to paginate through the result set of a query to retrieve more than the 100 tweets one call returns.

- Online media platforms in general are very useful information filters. Gathered information will be ensured to accord to certain quality guidelines and will reach some sort of minimal information value. This is grounded in the fact, that journalists and media organizations shall serve in public interest.

  > "The media is expected to fulfill a variety of social roles, from creating a market of opinions and ideas, to enhancing pluralism and serving as the fourth estate [62] ".

  Consequently online media platforms are a great source for qualitative data and have the potential of improving the system's accuracy.

  The implemented crawler for news websites is configured with 50 well established platform URLs, following links without leaving the domain. The mentioned list of links to online news websites that were being crawled came from *abyznewslinks* [11], a web page that lists the most important online news platforms per country. In the prototype the links for international media were chosen. These are listed in the Appendix A.0.1 .

  News article content together with some meta data elements is extracted with the help of Apache Tika. The prototype stores title and content of the news article as well as the crawl date.

[11]http://www.abyznewslinks.com

5.6.1.2 Data Extraction of Crowdfunding Project Data

Project data is gathered from Kickstarter by frequently scrapping the newest projects from the official website. The implemented crawler gathers data by requesting new project frequently from `www.kickstarter.com`.
It is not guaranteed though that the full list of the projects is complete, application. It can also be the case, that duplicate projects are generated in this workflow, but this problem is attacked in the information retrieval system in the indexing step, where duplicate projects will be updated instead of being added again.

The data set contains all different kinds of values. For instance date values like deadline, boolean values like the information, if a video exists, class values like the category of the crowdfunding project, numeric values like goal and string values like name of the project. Other projects related to crowdfunding prediction like [7] discard unsuccessful projects right away by removing projects without backers, but this is not possible for this application as projects should be assessed instantly for new projects.

In the course of the pre-training phase crowdfunding project data and online news data is stored in the index to be able to compute similarity in later phases for each project as otherwise it wouldn't be possible to compute similarity to other projects or similarity to online news articles for the first project that gets into the index.
Data sets are at this stage do not have to necessarily contain any of our computed features like novelty or popularity scores. Consequently project data from a website [12] that scrapes and provides free data sets is being used.

After the pre-training phase project data is collected by calling the Kickstarter website. Specifically sites, that present certain classes of projects and project presentation sites are processed.
The prototype calls a certain URL to receiving the webpage containing the newest projects frequently. In the returned HTML document name and description for each project are extracted. Also a link to the data API that can be called to receive project data directly in JSON format is parsed from the HTML document. This API is used internally to fill up project sites with relevant data. Now to be able to call that API the URL to the project site has to be called first. This is

[12] `www.webrobots.io`

necessary to be allowed to receive data from the internal API, as it is not intended for external calls.

So the full project URL has to be stored to enable receiving project data in the future as this URL has to be called first in order to request pure project data from the website.

Now all Kickstarter features can be extracted as JSON data, be converted to Java objects right away and be further processed in the application.

In the prototype Indiegogo data was used in the pre-run only and Kickstarter projects would be scraped frequently as well.

5.6.2 Data Storage

The presented data should be stored persistently in order to retain data over application lifetime. Relational databases are ideal for storing and manipulating structured data, eliminating redundancy and quickly and securely updating individual records.
Lucene on the other hand is a file system, that provides full text search, natural language processing functions, text similarity computation and retrieval functions. It handles high volume of free form text data, high volume of interactive text-based queries, has fewer demands for different record types, and at the same time is able to do non-text data manipulation [17].
Lucene is specifically designed to handle text data and computations, while relational databases are designed to handle all kind of structured data and transactions and handle updates more effectively.

For the prototype Lucene is used for data storage which bears potential for future enhancements. Storing data in a relational database and making an index over it would be better for productive systems but this is excluded to simplify and speed up the implementation of the prototype.

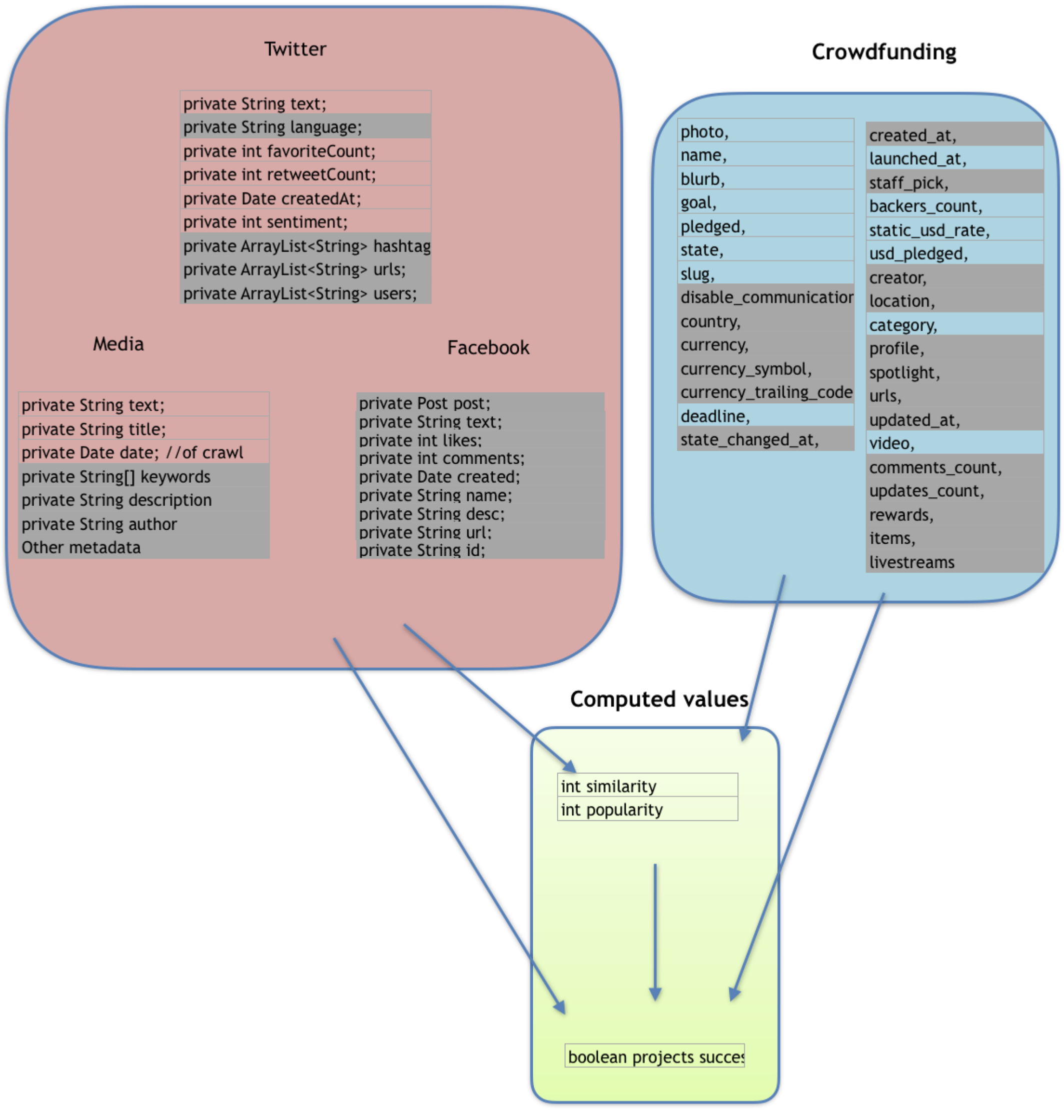


FIGURE 5.5 | **Dataset Description.** Figure describing the generated set of data from the sources used in the prototype implementation.The features not being used in the prototype are greyed out. The composition of computed values in the green box is also demonstrated.

5.6.3 Data Set used in Successfulness Predictions

All data extracted in the prototype can be seen in Figure 5.5.
Data is further analyzed and certain metrics are computed or aggregated from extracted data before being fed into the machine learning application. NoveltyScore as described in 4.2.5 is computed from crowdfunding project data. Various metadata information is extracted from news pages, but only creation date is used in the prototype. The sentiment of each tweet is computed to draw the conclusion if the tweet talks positive, negative, or neutral about the contained topic. Statistics

for all tweets about a project will be aggregated to a Twitter score according to the created model while NewsScore is computed from news data. Then these metrics are combined to the final PopularityScore. The machine learning component uses the provided data to compute the score for successfulness of the projects.

5.6.4 Data Preparation and Prediction Process

Data preparation starts immediately after the crawling process. Certain metrics are computed according to the models used to describe novelty and popularity including sentiment analysis and aggregation of values.
By using natural language processing (NLP) techniques additional metadata can be extracted from texts. Especially full online news articles contain a variety of information. In the prototype system NLP tools are solely used for sentiment analysis on tweets and for metadata extraction. It is not sufficient to identify certain keywords in sentiment detection. There are a number of necessities like part-of-speech-tagging, to be able to extract sentiment from text as it is important to know about the relation between words. A simple example would be the negation of a statement. The prototype uses the Java StanfordNLP library to realize these tasks. Derived sentiment scores are used to compute scores according to the model.
All information that is contained in the index can be used in the *Machine Learning* component, where data is further transformed to fit into a neural network. The *Machine Learning* component receives data as a response from our REST API in JSON format. Data is then processed by the Python libraries scikit-learn, Tensorflow, Keras, XGBoost, and Pandas.

Before we arrive at the final machine learning prediction model, first we create a reference model solely using features from Kickstarter, which is very similar to other works described in Chapter 3. Then we will try to improve the model in terms of preconditions and overall accuracy. A Detailed discussion of the prediction results are presented in Chapter 6.

In the first cycle we went on to solely test our Kickstarter project features in the dataset to see how far we come from there.

The following steps have to be taken in the process of prediction. Some of the features have to be encoded in order to do computation on them.

```
#Encoding the categorical features(the independent variables)

from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler

labelencoder_X_1 = LabelEncoder()
X[:,0] = labelencoder_X_1.fit_transform(X[:,0])
labelencoder_X_2 = LabelEncoder()
X[:,1] = labelencoder_X_2.fit_transform(X[:,1])

onehotencoder = OneHotEncoder(categorical_features = [0,1])
X = onehotencoder.fit_transform(X).toarray()

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

Listing 5.4: Encoding and Scaling in the Machine Learning Application

```
             precision    recall  f1-score   support

          0       0.90      0.96      0.93         0
          1       0.83      0.68      0.75         0

avg / total       0.89      0.89      0.89         0
%shorter version dann noch mal in result section
```

Listing 5.5: Results of the Crowdfiunding Project Success Prediction Using All Features. Class Label 1 specifies successful projects.

String labels have to be encoded to numerical values and further we have to apply *one-hot encoding* [63] to encode categorical features as binary vectors. Each integer value will be represented as a binary vector that is zero for all values except for the index of the integer in the categorial vector. This ensures, that the model doesn't hypothesize a distance relationship in these features.

Scikit-Learn's StandardScaler standardizes features by removing the mean, scaling to unit variance, and approximating to standard normally distributed data. The process of Encoding and standardization can be seen in Figure 5.4.

The dataset is then divided into a training set holding 90 percent of the dataset and a testing set holding the remaining ten percent. Then a very simple neural network with four layers of perceptrons is trained on the data. The results are shown in Listing 5.5

Figure **??** shows a correlation matrix visualization, that demonstrates the relation between *backers count* and *pledged amount*.
Dark, green spots indicate low correlation between the values of x - y axis. Bright, yellow spots in contrary signal higher correlation. The feature *comments count*

```

              precision    recall  f1-score   support

          0       0.75      1.00      0.86         9
          1       0.00      0.00      0.00         3

avg / total       0.56      0.75      0.64        12
```

Listing 5.6: The feature set is limited to those features, available at the creation time of a crowdfunding project. Class Label 1 specifies successful projects.

shows a similar correlation. The diagonal shows correlation with itself and hence contains no meaning in this plot.

We can follow that the number of backers as well as the number of comments are the key features in predicting success with precision and recall close to 90 percent for our dataset, which is what we saw in 5.5. This is very similar to what was already found in [41] where boosting the model with parameters describing the number of backers and the pledged amount would result in similar prediction accuracy. The problem with these kind of features is, that they are time sensitive in the sense that they cannot be used before a considerable time of project funding has past.
In contrary there is data available on online media that can potentially assist in the predicting process much faster. In [7] it is stated that tweets from the first three days are the most relevant for crowdfunding projects.

So we discard our first model and start a second cycle training a model on data that (1) is already available at, or a very short time after project creation, and (2) is not exclusive to certain crowdfunding projects or online media platforms So we add the following features to the set and excluded those that are not meaningful in the beginning phase of a project.
So effectively the number of backers, the number of comments, and the pledged amount we used in the first model were replaced by our computed features in the model of the second cycle. Newly included features are the average text length of relevant tweets, the average sentiment for those tweets, the total number of retweets, the total number of favorites, twitter score, popularity score according to our model, novelty score and news score computed over similarity measures. The same preprocessing was applied. Results can be seen in Listing 6.1

We can see, that the prediction for successful projects performs poor. All of the projects, that are successful in reality, are labeled incorrectly.

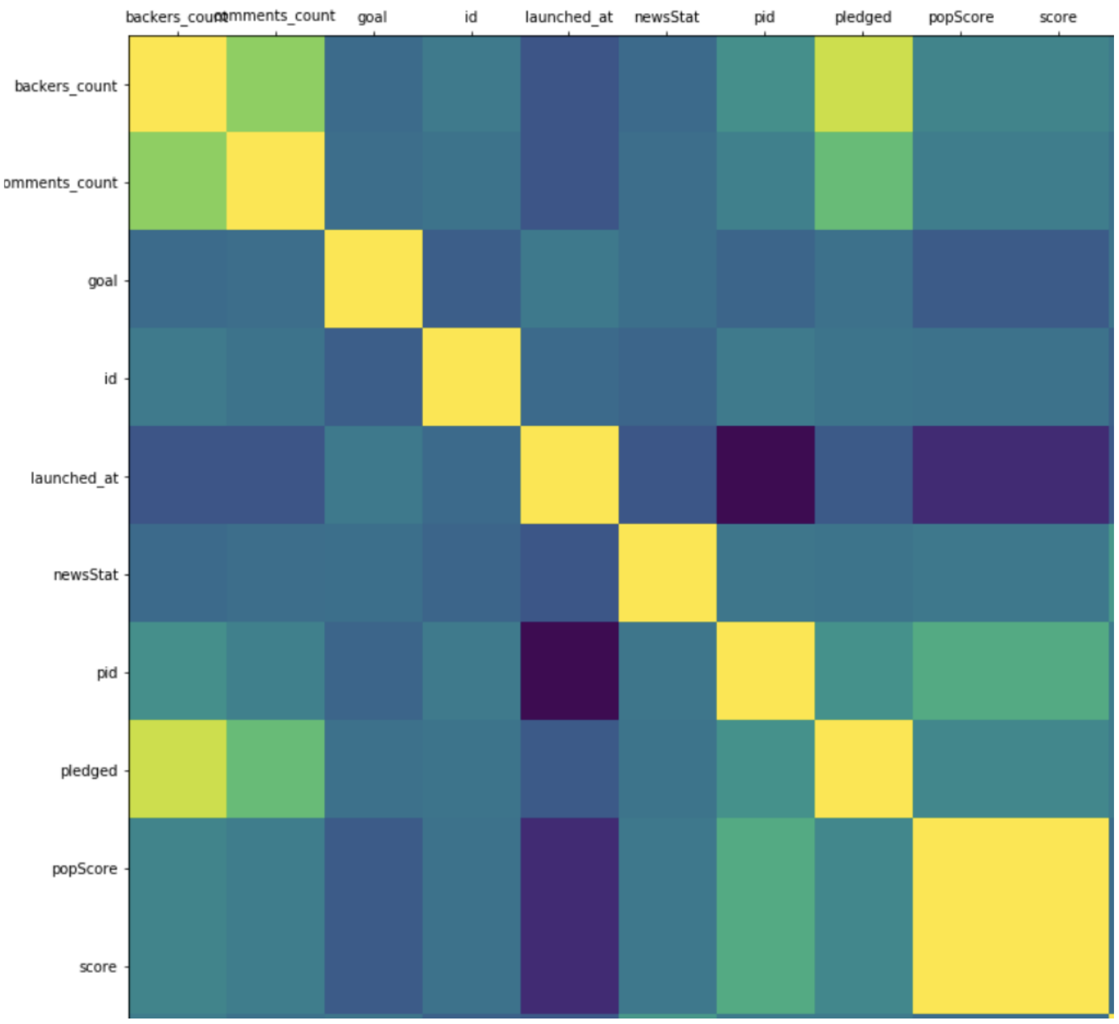


Figure 5.6 | **Correlation Matrix Visualization.** The figure shows a correlation matrix visualization, that demonstrates relation between *backers count* and *pledged amount*. Dark, green spots indicate low correlation between the values of x - y axis. Bright, yellow spots in contrary signal higher correlation. Also *comments count* shows similar correlation. The diagonal shows correlation with itself and hence contains no meaning in this plot.

This could be induced by the unbalanced data set. Most of the projects in the data set are unsuccessful. In order to improve the model one could respectively either include a higher number of successful projects into the training set, or modify the loss function used in the model training phase to penalize false negatives.

Another option is the *XGBClassifier* from the XGBoost library. Results can be seen in Listing 5.7.

The overall accuracy in the model is acceptable, while only features available at the start of the crowdfunding project were utilized in the model. Further result discussion takes place in Chapter 6.

```

             precision    recall  f1-score   support

          0       0.89      0.89      0.89         9
          1       0.67      0.67      0.67         3

avg / total       0.83      0.83      0.83        12
```

LISTING 5.7: The feature set is limited to those features, available at the creation time of a crowdfunding project. Class Label 1 specifies successful projects.

5.6.5 Batch Learning of the Machine Learning Model

In online machine learning new data is added in sequential order and is used to update the machine learning model as opposed to offline learning where the full data set is available from the beginning. To make use of the fact that the application is continuously crawling new data, we must update the training set frequently and retrain the machine learning model from time to time.
This is a method that lies somewhere between offline and online learning.

Accordingly an update function, shown in Listing 5.8 is implemented, that revisits crowdfunding project sites after some time to update the pledged amount of projects. Hence it checks whether the project was successful or not.
Now the function that returns the training set to the machine learning function will have access to labeled data and the *Machine Learning* component consequently operates on a larger data set the longer the application runs. At the same time this supports the prediction model in adopting to changes in the crowdfunding environment.

```java
    //run through all projects to update pledged
    public void updateProjects() {
          Indexer in = new Indexer();
          in.setIndex("project");
        List<org.apache.lucene.document.Document> pro = in.searchAllDocs();
        for (org.apache.lucene.document.Document d : pro) {
            String link = (d.get("articleLink"));
            String page = d.get("articlePage");
            if (page != null) {
            try { Jsoup.connect(page).ignoreContentType(true).execute();
            } catch (IOException e1) {e1.printStackTrace();}

            Double p = 0.0; //initial p
            if (d.get("pledgedInitial") == null) {p = Double.valueOf(d.get("pledged")); } else {
            if (Double.parseDouble(d.get("pledgedInitial")) == -1.0) {
                p = Double.valueOf(d.get("pledged"));
            } else {p = Double.parseDouble(d.get("pledgedInitial")); } }
//          get project and only change score and pledged
            KickstarterProject k = null;
            ObjectMapper mapper = new ObjectMapper();
            try {
            k = mapper.readValue(new URL(link), KickstarterProject.class);
            } catch (IOException e) {System.err.println(e.getMessage());}
            d.removeFields("pledgedInitial");
            d.removeFields("pledged");
            d.add(new StringField("pledgedInitial", String.valueOf(p), Field.Store.YES));
            d.add(new StringField("pledged", String.valueOf(k.getPledged()), Field.Store.YES));
            in.index(d);
            }
        }
    }
```

Listing 5.8: The function to update project's *pledged amount* in the *KickstarterReader* class. The Implementation is necessary in order to generate class labels for crowdfunding projects

Chapter 6

Results and Evaluation

6.1 Results and Evaluation of the Thesis

The goal of this thesis was to leverage online media data to automatically predict successfulness of crowdfunding projects and further compute novelty and popularity measures. Therefore data was gathered from different online sources and fed into an information retrieval system. Retrieval methods were used to retrieve relevant data according to our models describing successfulness, novelty and popularity.

To assess the computed scores a prototype was implemented.
The prototype crawls new data fully automatically from websites and updates the user interface and machine learning components frequently.
Online media data from Twitter and online news platforms is extracted and leveraged to compute novelty and popularity of crowdfunding projects. The system computes text similarity between projects and as well as between projects and online media data to generate the additional features novelty and popularity. The features are intended for the presentation to the users and also used to train a machine learning model. This model predicts successfulness for projects. A graphical user interface is implemented listing the top 50 ranked projects in the system, sorted by their given successfulness score.

The prototype indexed 195614 projects from Kickstarter and 20970 Indiegogo from February 2018 in an pre-training phase.

```

             precision    recall  f1-score   support

          0       0.89      0.89      0.89         9
          1       0.67      0.67      0.67         3

avg / total       0.83      0.83      0.83        12
```

LISTING 6.1: The feature set is limited to those features, available at the creation time of a crowdfunding project

For the first run of the prototype, the indexer held 61870 news websites and 217022 crowdfunding projects in total. During the run data was analyzed and scores were predicted for 105 crowdfunding projects of which the top 50 were presented via user interface. The system ran for a relatively short time and consequently only 105 projects were analyzed. This implicates 105 different queries to Twitter. The total number of retrieved Tweets is never stored in the system, because Twitter data is aggregated instantly.

Results of the project success predictions can be seen in Listing 6.1

The machine learning model predicting successfulness of projects was tested by using a test set holding 10 percent of the data, that is used to train the model. The model reaches an average precision of 83 percent, a recall of 83 percent and a respective f1-score of 83 percent.

6.2 Discussion

The implemented prototype demonstrates the ability of the proposed framework to facilitate and automatize the generation of data sets from online resources. Besides that, modularity for future extensions or adjustments is ensured by the described architecture. This is an important aspect, as online content is subject to frequent changes.

The system quantifies novelty and popularity of projects. Although these are no objective measures, and therefore can not be validated. Nevertheless the computed features might be helpful to project creators, that desire additional information to improve their projects conceptually. Thereby novelty and popularity shall be indicators. The focus in the work however lies on ultimately predicting project success.

In the success prediction phase important project features were dismissed in the machine learning model to enable instant assessment of newly launched projects, that don't yet hold values for features like pledged amount or the number of backers.

Nevertheless, the prediction process in the implemented prototype use case was comparable to other solutions that previously worked on this topic using all features, as we saw in Section 3.

The overall accuracy of our crowdfunding project success prediction was acceptable. Above all the model performance might be improved by the presence of additional data. Particularly a higher number of successful projects in the training set might result in improved prediction accuracy. Especially our trained neural networks could perform significantly better on larger datasets.

In the end investments are never going to be 100 percent safe and same risk will remain. So as there is always some kind of uncertainty in many cases it is more important to have an answer right away as opposed to a more accurate answers later on. One situation where this point certainly holds true is the time when project creators are launching their campaign and need to know their success predictions beforehand. This enables them to make adaptions to the campaign, that are not possible at later stages.

Chapter 7

Conclusion

7.1 Summary

As more and more users are connected online, the phenomena of making use of the crowd is on the rise. On crowdfunding platforms project creators are hoping for the crowd to invest in their project, but it is hard to tell what makes a project appealing to the investors and contrary it is not easy to find the right projects to invest in.

A framework to facilitate and automatize the generation of crowdfunding project data sets from online resources is proposed. This framework also handles the presentation and interaction with users over a website and generates scores for novelty, popularity, and successfulness. The framework could as well be used on other online resources apart from crowdfunding.
Further, the implementation of a prototype tackling the before-mentioned problem and demonstrating the ability of the framework is described in detail.

The system integrates many strong technologies and is designed in a way that continuously ingests data into the system. Consequently scores and predictions are being kept up to date while the system is adjusting and improving on the new data that is integrated.

Being able to automatically create datasets to work on is a profitable capability by itself. The demonstrated approach could basically be used to attack any kind of prediction problem, that is influenced in some way by novelty and popularity.

The accuracy for the prediction in the implemented prototype reached a value of 83.33 percent. This result is particularly promising, as the predictions are solely based on static project features, available at the start of the crowdfunding campaign.

The proposed framework has the potential of assisting project funders in the process of project creation as well as backers in discovering promising projects, but in the end these are only predictions. Ultimately investments are never going to be 100 percent safe and predictable. There will always be some kind of risk and uncertainty in funding crowdfunding projects.
The popular adage from Heinlein, Robert A. in his book *"The Moon Is a Harsh Mistress"* from 1966 still holds true:

> "There ain't no such thing as a free lunch" [64]

Some risks have to be taken to profit but in the end funders take relatively low financial risks.

7.2 Limitations and Future Work

One of the limitations of the prototype is the selection of online news media platforms to crawl from. In a first step an improvement could be to limit media platforms to what is called *Western Media* [1] [65] as opposed to the international selection of media used in the prototype system. Kickstarter and many projects are not intended to be launched internationally.
This could be enhanced by adjusting the news selection to the region of the projects launch because it can be expected that most of the funding will be received from there. Besides that, specialized media about crowdfunding is potentially providing even more relevant data.

The process of novelty computation involves computing similarities between projects. At the moment all similarities are equally weighted ignoring the properties of the projects being compared. Similarity to successful projects could be rated higher than similarity to projects, that lack backers or have minimum funding goals.
So including project statistics in the process of novelty prediction could possibly

[1]Media in North America and Western Europe

improve the novelty score and its contribution to success prediction.
Besides that, measuring similarity could include other project features apart from their name to ensure that cloned projects with differing names are regarded as similar.

Another important aspect that could easily be improved in the application, for sure, was the short period time the system was running and consequently the relatively small data set, that was being used to train the machine learning model. However the architecture of the application is very much scalable and by design, the longer the application is running, the more labeled data will be available to the system. Particularly in machine learning and first and foremost in neural networks large data sets are key in creating sufficiently good models for prediction. Above all FENG et al. in [66] pointed out, that using the logarithm in normalization and data variability reduction in many cases may not perform well and should be avoided. So it might be a good idea to think of superior solutions in the proposed popularity model.

There are other aspects of crowdfunding that are still not covered in the prediction model or rather in the crawled data. What about projects that get funded successfully but cannot deliver afterwards ? This surely is an important aspect and would require to include fraud detection and prediction of failure in realization after successful funding for reasons like overestimation of one's own capabilities or other reasons [67]. To assess these project success indicators additional features would possibly be needed like experience and expertise of project creators, an assessment of the development stage of the project, and the projects promised features and functionalities.

To improve accuracy of the prediction the prototype implementation can be extended as it is meant to be done, by writing additional modules to extract data from other crowdfunding websites continuously. Other extensions of the framework could address the enhancement of the user interface. For instance providing possibilities to further explore project data or providing some kind of sorting capabilities to the user.

Lastly the machine learning component, that uses a white box classifier enables the system to give more detailed feedback to the user by highlighting the decision process in the machine learning model.

Appendix A

Appendix

A.0.1 Online News Websites crawled to assess Popularity Score

List of 50 links to online news websites that were being crawled in order to assess a popularity score

http://www.trust.org/humanitarian/

http://www.nytimes.com/

https://www.armedpolitics.com/

http://www.einnews.com/

https://www.fronteranews.com/

http://www.globalpost.com/

http://www.globaltrading.com/

http://www.greatreporter.com/

http://www.ihavenet.com/

http://www.indepthnews.net/

http://www.insnews.org/

http://www.insideworld.com/

https://iwpr.net/global/afghanistan

http://www.irinnews.org/

http://www.klezio.com/

http://www.laspecula.com/

http://www.lepetitjournal.com/

http://www.mambolook.com/

http://www.mapreport.com/

http://newsmixx.com/

http://www.newsnow.co.uk/

http://www.oneworld.org/

http://www.peacelink.it/

http://reliefweb.int/

http://www.topix.com/

http://www.veryquiet.com/news.php

http://wn.com/

http://www.theworldfolio.com/

http://www.worldphotos.com/

http://www.worldpoliticsreview.com/

http://www.worldpronews.com/

http://www.worldtribune.com/

http://www.theworldweekly.com/

http://news.yahoo.com/

http://www.economist.com/

http://hosted.ap.org/dynamic/fronts/WORLD?SITE=AP

http://www.ipsnews.de/

http://www.ipsnews.net/

http://www.ipsnoticias.net/

http://www.interfax.com/

http://ara.reuters.com/news/world

http://www.reuters.com/news/world

http://www.redherring.com/

http://www.scientificamerican.com/

http://www.wired.com/

http://news.cnet.com/

http://recode.net/

http://www.talkingnewmedia.com/

http://www.thetechherald.com/

http://www.zdnet.com/

Abstract

Crowdfunding is a widely used way of funding by groups of investors. Predicting success of newly launched crowdfunding projects is valuable for both, the project creators and project funders. Time plays an important role here as you want your predictions as soon as possible to be effective. The most important prediction features are not known at the time of the project start. Online media data could assist in tracking popularity of projects beforehand. In this work a framework is proposed, that automatically extracts features from crowdfunding projects and at the same time measures projects novelty based on whether similar projects are on the market and projects popularity by leveraging online media data. The prototype implementation also provides an user interface to browse through data. Predictions based on the novelty and popularity score were not found successful the way they were implemented. Nevertheless, the framework is designed to be very flexible and modular and can therefore be extended easily to bring new data into the system. In general the system can be adjusted to any kind of prediction problem, that relates to data available on the Web.

Zusammenfassung

Crowdfunding ist eine weit verbreitete Art der Finanzierung von Projekten durch Gruppen von Investoren . Die Voraussage des Erfolgs von neu gestarteten Projekten kann sowohl wertvoll für die Ersteller dieser Projekte als auch für die Unterstützer sein. Zeit spielt in dieser Hinsicht eine wichtige Rolle, da die Voraussagen so schnell wie nur möglich getroffen werden müssen, um diese effektiv zu nutzen. Die wichtigsten für die Voraussage notwendigen Eigenschaften von Projekten sind zum Zeitpunkt der Erstellung des Projektes allerdings noch nicht bekannt. Daten aus Onlinemedien könnten dabei helfen, die Popularität dieser Projekte schon zu diesem Zeitpunkt zu bestimmen und die Voraussage zu unterstützen. In dieser Arbeit wird ein Framework vorgestellt, welches Eigenschaften von Crowdfunding Projekten vollautomatisch extrahiert und gleichzeitig Einzigartigkeit anhand der

Tatsache, ob ähnliche Projekte am Markt sind und die Popularität anhand von Onlinemedien misst. The Implementierung eines Prototypen verfügt darüber hinaus über eine Benutzeroberfläche für die Durchsicht der Daten. Auf den beschriebenen Eigenschaften Einzigartigkeit und Popularität basierende Voraussagen, so wie sie implementiert wurden, führten zu keinen korrekten Ergebnissen. Nichtsdestotrotz ist das vorgestellte Framework modular und flexibel konzipiert und kann daher einfach erweitert werden, um neue Datenquellen zu erschliessen und das Voraussagemodell zu verbessern. Ganz allgemein kann das System auf jedwedes Problem angepasst werden, welches Bezug zu Onlinedaten aus dem Internet hat.

Bibliography

[1] Lester Allan Lasrado and Artur Lugmayr. Crowdfunding in finland: A new alternative disruptive funding instrument for businesses. In *MindTrek*, 2013.

[2] Daren C. Brabham. *Crowdsourcing*. The MIT Press, 2013. ISBN 0262518473, 9780262518475.

[3] Kickstarter statistics, July 2018. URL `https://www.kickstarter.com/help/stats`.

[4] Cosmina Radu Wendy Carrara and Heleen Vollers. Open data maturity7 in europe 2017. November 2017.

[5] Stefaan Verhulst and Andrew Young. Open data in developing economies - toward building an evidence base on what works and how. JULY 2017.

[6] Picking a killer title for your crowdfunding campaign, 07 2018. URL `https://www.thebalancesmb.com/picking-a-killer-title-for-your-crowdfunding-campaign-985186`.

[7] Yan Li, Vineeth Rakesh, and Chandan K. Reddy. Project success prediction in crowdfunding environments. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 247–256, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3716-8. doi: 10.1145/2835776.2835791. URL `http://doi.acm.org/10.1145/2835776.2835791`.

[8] Sitaram Asur and Bernardo A. Huberman. Predicting the future with social media. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT

'10, pages 492–499, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4191-4. doi: 10.1109/WI-IAT.2010.63. URL `http://dx.doi.org/10.1109/WI-IAT.2010.63`.

[9] A Ordanini. Crowd funding: customers as investors. *The Wall Street Journal, 23 March, p. r3.*, 2009.

[10] Andrea Ordanini, Lucia Miceli, Marta Pizzetti, and A. Parasuraman. Crowdfunding: Transforming customers into investors through innovative service platforms. *Journal of Service Management*, 22(4):443–470, 8 2011. ISSN 1757-5818. doi: 10.1108/09564231111155079.

[11] Anirban Mukherjee, Cathy L. Yang, Ping Xiao, and Amitava Chattopadhyay. Does the crowd support innovation? innovation claims and success on kickstarter. *HEC Paris Research Paper No. MKG-2017-1220*, 2017.

[12] Chun-Ta Lu, Sihong Xie, Xiangnan Kong, and Philip S. Yu. Inferring the impacts of social media on crowdfunding. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 573–582, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2351-2. doi: 10.1145/2556195.2556251. URL `http://doi.acm.org/10.1145/2556195.2556251`.

[13] Ed Greengrass. Information retrieval: A survey. Technical report, University of Maryland, Baltimore County, 2000.

[14] Ayse Goker and John Davies. *Information Retrieval: Searching in the 21st Century.* October 2009.

[15] Richard Chbeir, Youakim Badr, Ajith Abraham, and Aboul-Ella Hassanien. *Emergent Web Intelligence: Advanced Information Retrieval.* Springer Publishing Company, Incorporated, 1st edition, 2010. ISBN 1849960739, 9781849960731.

[16] Bing Liu. *Sentiment Analysis and Opinion Mining.* Morgan & Claypool Publishers, 2012. ISBN 1608458849, 9781608458844.

[17] Vineeth Mohan Edwood Ng. *Lucene 4 Cookbook.* Packt Publishing, June 2015.

[18] Judith Hurwitz Daniel Kirsch. *Machine Learning For Dummies, IBM Limited Edition.* John Wiley and Sons, Inc., 2018.

[19] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python.* O'Reilly Media, Inc., 1st edition, 2009. ISBN 0596516495, 9780596516499.

[20] Nitin Indurkhya and Fred J. Damerau. *Handbook of Natural Language Processing.* Chapman & Hall/CRC, 2nd edition, 2010. ISBN 1420085921, 9781420085921.

[21] Wael H. Gomaa and Aly A. Fahmy. Article: A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18, April 2013. Full text available.

[22] Djoerd Hiemstra. Information retrieval models. In *Information Retrieval: Searching in the 21st Century*, November 2009.

[23] Lucene similarity class, August 2018. URL `https://lucene.apache.org/core/3_5_0/api/core/org/apache/lucene/search/Similarity.html`.

[24] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval.* Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715.

[25] Liu H. Hu X. Text analytics in social media. *Aggarwal C., Zhai C. (eds) Mining Text Data*, 2012.

[26] H. Schildt and J. Holmes. *The Art of Java.* In-Depth Programming and Web Development Series. McGraw-Hill/Osborne, 2003. ISBN 9780072229714. URL `https://books.google.at/books?id=aBQARPLJTgoC`.

[27] Christopher Olston and Marc Najork. Web crawling. *Found. Trends Inf. Retr.*, 4(3):175–246, March 2010. ISSN 1554-0669. doi: 10.1561/1500000017. URL `http://dx.doi.org/10.1561/1500000017`.

[28] K.Thirumoorthy T. Mahara Jothi. A survey on web forum crawling techniques. *International Journal of Innovative Research in Science, Engineering and Technology*, Volume 3, Special Issue 3, March 2014.

[29] Namrata H. S. Bamrah, B. S Satpute, and Pramod Patil. Web forum crawling techniques. *International Journal of Computer Applications*, 85(17):36–41, January 2014.

[30] Robert P. Schumaker, Osama K. Solieman, and Hsinchun Chen. *Sports Data Mining*. Springer Publishing Company, Incorporated, 1st edition, 2010. ISBN 144196729X, 9781441967299.

[31] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social Media Mining: An Introduction*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107018854, 9781107018853.

[32] Florian Meier, David Elsweiler, and Max L. Wilson. More than liking and bookmarking - towards understanding twitter favouriting behaviour. In *Eighth International Conference on Weblogs and Social Media, ICWSM 2014*, 2014. URL `https://epub.uni-regensburg.de/33987/`.

[33] Number of monthly active users on twitter, July 2018. URL `https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/`.

[34] Stephen Marsland. *Machine Learning: An Algorithmic Perspective*. Chapman & Hall/CRC, 1st edition, 2009. ISBN 1420067184, 9781420067187.

[35] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. informatica 31:249–268, 2007.

[36] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL `http://doi.acm.org/10.1145/2939672.2939785`.

[37] Ethem Alpaydin. *Introduction to Machine Learning.* The MIT Press, 2nd edition, 2010. ISBN 026201243X, 9780262012430.

[38] Nikolay Burlutskiy, Miltos Petridis, Andrew Fish, Alexey Chernov, and Nour Ali. An investigation on online versus batch learning in predicting user behaviour. In *SGAI Conf.*, pages 135–149. Springer, 2016.

[39] Peter (Haochen) Zhou. Predicting the success of kickstarter campaigns, December 2017. URL `https://www.stat.berkeley.edu/~aldous/157/Old_Projects/Haochen_Zhou.pdf`. STAT 157 Final Project.

[40] Muhammad Ghauri Rachel Downs. Predicting kickstarter campaign success, January 2018. URL `http://rachellaurenwoods.com/wp-content/uploads/2018/01/Term-Project-Report.pdf`. Term-Project-Report.

[41] Michael D. Greenberg, Bryan Pardo, Karthic Hariharan, and Elizabeth Gerber. Crowdfunding support tools: Predicting success and failure. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 1815–1820, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1952-2. doi: 10.1145/2468356.2468682. URL `http://doi.acm.org/10.1145/2468356.2468682`.

[42] Vincent Etter, Matthias Grossglauser, and Patrick Thiran. Launch hard or go home: Predicting the success of kickstarter campaigns. In S. Muthu Muthukrishnan, Amr El Abbadi, and Balachander Krishnamurthy, editors, *COSN*, pages 177–182. ACM, 2013. ISBN 978-1-4503-2084-9. URL `http://dblp.uni-trier.de/db/conf/cosn/cosn2013.html#EtterGT13`.

[43] Thanh Tran, Kyumin Lee, Nguyen Vo, and Hongkyu Choi. Identifying on-time reward delivery projects with estimating delivery duration on kickstarter. *CoRR*, abs/1710.04743, 2017. URL `http://arxiv.org/abs/1710.04743`.

[44] Bo Wu and Haiying Shen. Analyzing and predicting news popularity on twitter. *Int J. Information Management*, 35(6):702–711, 2015. URL `http://dblp.uni-trier.de/db/journals/ijinfoman/ijinfoman35.html#WuS15`.

[45] Roja Bandari, Sitaram Asur, and Bernardo A. Huberman. The pulse of news in social media: Forecasting popularity. *CoRR*, abs/1202.0332, 2012. URL http://arxiv.org/abs/1202.0332.

[46] M. Cha, H. Haddadi, F. Benevenuto, and K.P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *4th International AAAI Conference on Weblogs and Social Media (ICWSM)*, 2010. URL http://scholar.google.de/scholar.bib?q=info:rqhbqWEH79kJ:scholar.google.com/&output=citation&hl=de&as_sdt=0&ct=citation&cd=10.

[47] Wenbin Zhang and Steven Skiena. Improving movie gross prediction through news analysis. *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, 1:301–304, 2009.

[48] Po-Wei Liang and Bi-Ru Dai. Opinion mining on social media data. In *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management - Volume 02*, MDM '13, pages 91–96, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-0-7695-4973-6. doi: 10.1109/MDM.2013.73. URL http://dx.doi.org/10.1109/MDM.2013.73.

[49] Soumya Banerjee and Nitin Agarwal. Analyzing collective behavior from blogs using swarm intelligence. *Knowledge and Information Systems*, 33(3): 523–547, 2012. ISSN 0219-3116. doi: 10.1007/s10115-012-0512-y. URL http://dx.doi.org/10.1007/s10115-012-0512-y.

[50] Model view controller on wikipedia, July 2018. URL https://de.wikipedia.org/wiki/Model_View_Controller.

[51] Cambridge dictionary - novelty, July 2018. URL https://dictionary.cambridge.org/de/worterbuch/englisch/novelty.

[52] Cambridge dictionary - popularity, July 2018. URL https://dictionary.cambridge.org/de/worterbuch/englisch/popularity.

[53] R.M. Reese. *Natural Language Processing with Java*. Community Experience Distilled. Packt Publishing, 2015. ISBN 9781784398941. URL `https://books.google.at/books?id=q7y4BwAAQBAJ`.

[54] John H. McDonald and University of Delaware. *Handbook of Biological Statistics*. Sparky House Publishing, 2009.

[55] Rami Belkaroui, Rim Faiz, and Pascale Kuntz. User-tweet interaction model and social users interactions for tweet contextualization. In Manuel Núñez, Ngoc Thanh Nguyen, David Camacho, and Bogdan Trawiński, editors, *Computational Collective Intelligence*, pages 144–157, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24069-5.

[56] Ken Williamson. *Learning AngularJS: A Guide to AngularJS Development*. O'Reilly Media, Inc., 1st edition, 2015. ISBN 1491916753, 9781491916759.

[57] J. Gosling and H. McGilton. *The Java Language Environment: A White Paper*. Sun Microsystems Computer Company, 1995. URL `https://books.google.at/books?id=pUJ_GwAACAAJ`.

[58] Lucene vs solr, 07 2018. URL `http://www.lucenetutorial.com/lucene-vs-solr.html`.

[59] Embeddedsolr, 07 2018. URL `http://wiki.apache.org/solr/EmbeddedSolr`.

[60] The cambridge analytica files, July 2018. URL `https://www.theguardian.com/news/2018/mar/26/the-cambridge-analytica-files-the-story-so-far`.

[61] Twitter blog, 07 2018. URL `https://blog.twitter.com/`.

[62] Itai Himelboim and Yehiel Limor. Media institutions, news organizations, and the journalistic social role worldwide: A cross-national and cross-organizational study of codes of ethics. 14:71–92, 01 2011.

[63] One-hot-encoding scikit-learn, 08 2018. URL `http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html`.

[64] Robert A. Heinlein. *The Moon Is a Harsh Mistress*. 1966.

[65] J. Hardy. *Western Media Systems*. Communication and society. Routledge, 2008. ISBN 9780415396912. URL `https://books.google.at/books?id=3RGM7wsZthEC`.

[66] LU N. FENG C., WANG H. Log-transformation and its implications for data analysis. *Shanghai Archives of Psychiatry*, pages 105–109, 2014.

[67] Vom netz finanziert und nie geliefert crowdfunding als millionengrab, 07 2018. URL `http://derstandard.at/2000061926301/Vom-Netz-finanziert-und-nie-geliefert-Crowdfunding-als-Millionengrab`.