Creating Business Value with Multi-Domain Data Analysis on Web-Scraped Data

from the Largest German Recipe Platforms

M.Sc. Industrial Engineering
Master's Thesis

Tolga Buz
Matr. No.: 346836
Technical University of Berlin
06.04.2018

Supervision:
Prof. Dr. R. Zarnekow, Thorsten Pröhl
Chair of Information and Communication Management
Institute of Technology and Management

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

I hereby declare that I have produced this thesis independently and genuinely without unauthorized help of a third party and solely by using the specified references and tools.

Contents

Abstract

This thesis presents an exemplary approach to an end-to-end *Data Science* project, which serves the goal of performing in-depth data analysis to create business value by using external data sources from online communities. The project is set from a business perspective, more specifically in the context of a start-up for providing food boxes, i.e. deliveries that contain fresh ingredients for preparing specific recipes. For this purpose, a business case and the respective goals are defined, followed by the implementation, which includes collection, cleaning and analysis of data, as well as the evaluation of all results. This project employs best practices and popular frameworks that are being used frequently in this field, e.g. the CRISP-DM methodology for data mining projects and the programming language Python. The analysis includes techniques from exploratory analysis for pattern discovery, trend analysis for finding local and global trends, and data mining for clustering the data. The analysis is performed on data scraped from the three largest recipe websites regarding the size of their recipe databases, which contain 200,000 to 400,000 recipes each and communities of thousands of active users.
The project is able to achieve the proposed goals by successfully applying different analysis techniques to extract useful insights from the data sets and propose new product features. Even though there are limitations due to dirty data and missing values, sufficiently clean and valuable datasets have been crafted and used for various analysis techniques, which all serve the purpose of providing useful insights on market trends and user preferences as well as creating a competitive advantage for the business. The complete source code for this project has been open-sourced and is available online[1].

[1] `https://github.com/tbuz/RecipeScraper`

1 Introduction

This section aims to explain briefly why data is being generated and how it is being used for gaining insights, changing user experience and generating additional revenue. The following subsections explain why the setting of recipe websites has been chosen for this project, as well as the goals that the project aims to achieve.

1.1 Motivation

In recent years, IT services have become an important part of our economy. As server capacities have grown and become more stable, powerful and cheap, many new businesses have started building their value proposition mainly on digital services. With a growing number of such businesses, the amount of generated or collected data grows as well. In addition, traditional businesses have been undergoing a digital transformation, digitising at least parts of their businesses to keep up with the competition.

With a steadily increasing amount of services generating and collecting data, all actors in the internet's digital ecosystem have been creating a very large pool of different types of data. Usually, every service provider manages their data for a main purpose: to fulfill their value proposition. However, such data sources can be used to gain much more information and knowledge, if viewed from a different angle or in a different context. For instance, a website like facebook.com has been created as a social network, which enables users to create their own profile, add friends and exchange messages or online content. Thus, the main purpose for storing data on facebook's servers is to enable users to access their own and others' profiles, as well as see the history of posts and messages. Afterwards, more features have been added for convenience, like small games and pages that users can like and follow.

This is continuously generating large amounts of data, which can be used to conduct further (meta-) analysis, for example how strongly the users are linked to each other and how their relationships change over time [59], or how these social networks are able to influence the users' political activity and their personal network [5]. Facebook, being a company that needs to generate revenue and profits to perform well on the stock market, can use the data to understand each user's preferences and interests and thus offer personalised advertising. This enables the company to access the rapidly growing sector of targeted online marketing, which has become a significant factor in today's economy [37]. However, this also has implications to users' privacy and user experience [55]: For some people (and the government [6]), it may be disturbing that a user receives an advertisement on facebook about sports clothing, after they have liked a sports picture on Instagram, a social network for sharing pictures[2].

A less intrusive way of presenting users personalised content to improve their experience are *recommender systems* [46], which are employed, for example, by online shopping websites like Amazon [32]. These recommendations are based on the actions a user takes and suggest new products openly, based on a user's search history or wishlist. Being more transparent and possibly more relevant, such recommender systems are used less as a tool for third party marketing, but rather for increasing the platform's own revenues by motivating the users to buy and consume more.

In the recent past, Netflix and the KDD Cup (the so-called "Netflix Prize" [3])

[2] as described on Instagram's help page [25]

organised a quite sensational competition for motivating researchers to create an improved recommendation algorithm for the movie streaming service. This has inspired many other companies to publish parts of their databases and challenge the community to develop new services and features, for example Yelp, a website for reviewing businesses [62].
Applications like these are the main reason why the field of *Data Science* and data mining techniques have become remarkably popular in the past years, as they combine statistics, mathematics, computer science, and more to develop solutions that can tackle the growing needs of digital businesses. The process of creating value with data analysis consists of multiple steps, which require knowledge of different disciplines. The purpose of this thesis is to present an end-to-end process: data extraction from the web, its analysis in order to produce results and the evaluation in order to propose new business strategies. Thus, this work will present an exemplary *Data Science* project from a business viewpoint, explaining step-by-step which tasks need to be performed to create valuable results, which can be used to enable new services, features, or strategic decisions.

1.2 German Recipe Websites

With the emergence of continuous connectivity and social networks, the internet has proven to be a fountain of user generated content, mainly on account of discussion forums and digital communities. These websites usually specialise on a certain topic like video games, sports or astronomy, to offer their users a platform to talk about their hobbies, share their experiences and discuss the news and their opinions.
This project will work with German websites for cooking communities, which are mainly pages for sharing recipes. For this purpose, the three largest websites with regards to the number of recipes available (on their websites) have been chosen: *chefkoch.de*, *kochbar.de* and *eatsmarter.de*. (In the following, these websites will be referred to as chefkoch, kochbar and eatsmarter for simplicity.) There are multiple reasons to justify the choice of recipe communities:

Large amounts of data: The three selected websites offer large databases of recipes that add up to more than 900,000 documents, which all have multiple features that can be compared using statistical measures and analysis techniques. At the same time, a data set of this size is suitable for more complex data analysis tasks.

Dirty data: None of the German recipe websites provides an API[3] for developers. For this reason, all data has to be extracted manually by using a web scraper, leading to so-called dirty values, which need to be preprocessed before further analysis. Additionally, each domain is using a different data structure in their recipes, which is why the individual data sets are not completely comparable. Even though these circumstances make it more difficult to access the data, they reflect the reality of data analysis, for most of the time one has to work with dirty data and inconsistent sources.

[3]the Application Programming Interface, which helps developers to access the website's data easily and extract it in a clean way

Similar features: Even though there are differences in data structures, these three websites are still similar enough to be comparable, as they share a certain amount of features. This is also a reason why these websites have been chosen over others that might have higher traffic, but worse comparability.

Example for digitisation: Recipe websites are a suitable example for showing how digitisation (or digitalisation) is affecting the economy, as described in section 1.1. For many years, the main source to access recipes have been printed books, magazines and word of mouth. Online cooking communities have changed the recipe business by enabling users to access thousands of recipes for free with any device that has an internet connection.

Insights for food services: This project will present solutions and results that could be useful for any company related to food services, as examining cooking communities could be an important part of their market research. For example, by understanding trends and preferences in the cooking communities, a food delivery service can adapt its supply and work more efficiently.

Large digital community: Cooking communities maintain large data sets and are frequently visited digital communities in the German web, as can be seen in Table 1, which shows that just these three websites accumulate approximately more than 50 million visits per month.

Table 1: Top three German recipe website traffic (estimations, October 2017)

Website	**No. of Scraped Recipes**	**Estimated Traffic per Month**		
		SimilarWeb [53]	IVW [39]	Alexa [1]
chefkoch	313,751	48,400,000	57,349,233	36,000,000
kochbar	330,716	8,800,000	840,450	8,400,000
eatsmarter	200,350	4,350,000	570,420	4,500,000

For the total number of recipes in Table 1, the numbers of recipes that have been available for scraping have been chosen (as of October 2017), as this reflects the amount of data that can be extracted successfully for further analysis. However, the websites themselves or other sources might show higher total numbers, which can be true but are not as relevant for this project. The sources for the traffic estimations, SimilarWeb[4], the German IVW[5] and Alexa[6], each use different methods for estimating traffic numbers, which is why the estimations may show significant differences. SimilarWeb gathers data from their global panel, partnering ISPs (Internet Service Providers), public data sources and direct measurement. Alexa mainly collects data from their global panel and direct measurements from cooperating websites. The German IVW only uses their standardised measurement method [40], which is not supported by all web pages and mainly used in Germany. From these sources, SimilarWeb is using the most diversified method to reduce uncertainty, but their data about German websites might be less precise than about American websites, as data

[4]One of the main providers of online market information ("the industry standard in market intelligence" according to their website [53])

[5]The German Audit Bureau of Circulation [39]

[6]Amazon's web research service, does not provide any statistics for free. However, WolframAlpha is using data from Alexa and providing small parts of that data for free [61].

protection policies are different in each region, thus requiring different measurement techniques. Due to this, neither data source might be completely reliable, but their estimates show at least partial similarity and a rough approximation.

1.3 Goals of this Work

The purpose of this work is to show the complete process of data extraction, cleaning, analysis and knowledge discovery for an exemplary business case in the digital food service sector. These steps include:

- Explaining theoretical foundation and definitions
- Presenting a business case and defining goals
- Implementing and performing data extraction
- Cleaning and analysing data set using different approaches
- Presenting and discussing results assessing their relevance
- Recommending service improvements and new features
- Reflecting on further possibilities for future work

First, the theoretical foundation will be explained, defining and presenting the principles that this project will follow, as well as the specific goals that are to be achieved. In the next steps, the implementation will be described showing the decisions that have been made as well as their results. The implementation consists of different separate steps that require their own frameworks and programming packages, which is why these steps will be examined separately. After the implementation, the obtained results will be reviewed and their value evaluated. This project is aiming to create business value and, thus, potential new features or enrichment of the existing service will be discussed, using the knowledge extracted during the project.
To achieve these goals in this type of data analysis project, multiple approaches will be presented. These approaches will cover a basic exploratory analysis, more advanced trend analysis, and also extend the scope by employing data mining techniques by using a standardised process model as a guideline. Finally, the project's success will be assessed by measuring the compliance of the project results with the initial goals.

2 Theoretical Foundation

This section will describe the theoretical principles that underlie this project to provide a better understanding of the used terminology and techniques. While basic data analysis can be performed using statistical and exploratory approaches to get an initial understanding of the data set, more advanced techniques like trend analysis using time series data can provide more insights if the data can be provided in a suitable format. Additionally, data mining techniques can be used for deeper pattern and knowledge discovery. All of these techniques can be seen as parts of the interdisciplinary domain of *Data Science*.

2.1 Exploratory Data Analysis

Classical descriptive statistics aim to define hypotheses and then perform an analysis using a defined toolkit to prove these hypotheses by testing. The toolkit includes measures of central value and dispersion (e.g. mean and variance), as well as analysis of distribution, correlation, and regression, among others (Dixon, 1950 [9]). Whereas most of these techniques are still being used frequently, the process of hypothesis testing has been criticised for being too stiff and unflexible. This has led to the emergence of exploratory data analysis (EDA), which has been defined by John Tukey in order to create a new approach to statistical analysis (Tukey, 1977 [56]). EDA focuses on two main differences compared to the classical approach: the emphasis on a flexible attitude during data analysis and an increased use of plots for visualisation. This means that when approaching a data set, instead of setting hypotheses before analysis (or even before data collection), the exploratory analyst should rather perform a set of techniques and visualisations to learn more about the data set. Based on these initial insights, further steps need to be performed for a more detailed analysis in an iterative manner. As an example, the key statistics for measures of central value (like mean and median), the minimum/maximum, and the quartiles (or other percentiles) can offer quick insights into the data distribution and the existence of outliers, while plots help to present these insights. Visualisations that are commonly used are histograms (for showing the distribution of values along one dimension), boxplots (for showing the distribution of mean, median and quartiles) and scatterplots (for showing dependencies between two or more continuous variables). These techniques can be useful to achieve a fast understanding of outliers, correlations and the general shape of the dataset, which then can help to ask more questions, such as:

- How are the attributes distributed, are there outliers and in which regard are they different from the rest?
- Which dimensions are correlated or independent and how do they influence the data?
- Could a subset of dimensions be used for regression or classification to predict one attribute?
- Which dependencies, if there are any, could provide deeper insights into the dataset?

By asking questions like these, the data can be explored further using a step-by-step approach during the process. Of course, using the exploratory approach does not mean that traditional statistical techniques must not be used at all. The insights from EDA could instead be used to produce more accurate hypotheses which can then be tested using the established techniques. This approach suits well for the analysis of large datasets that originally have been collected for a different purpose, as exploring new patterns is usually the main goal in such a project.
In this project, the data will be explored from a top-down approach: starting with the website in general, then their users and finally the single recipes. During this process, newly gained insights might also motivate to reiterate previous analysis steps.
From a business perspective in this project's use case, these analysis results can be used to find outlying recipes and users with regards to their quality and popularity, which is a main approach to find the best recipes a website has to offer.

2.2 Trends and Time Series

Time series analysis describes techniques that are used on datasets that provide values in a frequent manner, which are usually time steps. As an example, the temperature values provided by a sensor once per second can easily be interpreted as a time series and then analysed using the respective techniques. This type of analysis is often used to calculate differences and correlations between multiple series as well as moving averages and (auto)regressions. These techniques can be useful to find out if a dataset is stationary or has trends (e.g. seasonal or general) as well as to make a forecast for future data points. Hamilton has provided a detailed description of these techniques in his much-quoted publication from 1994 [21].
In the use case of this project, elements have to be found that occur regularly over time and thus can be converted into a dataset that can be interpreted as a time series. As the underlying data is a collection of documents in the first place, further data preparation has to be undertaken. This project will explore two possibilities to view the recipes from a temporal perspective:

- Extracting all recipes regularly over a period of time to see how the data changes
- Grouping all recipes by date of publishing and finding a regular time interval

In order to provide data for the first option, the recipe websites have been scraped once per month over a period of six months, around the beginning of each month. Thus, data has been collected from 1st of October until 1st of March, providing a data set that covers half of a year, which could also be useful for identifying temporal effects. In case the first option will not be able to provide useful insights, the second approach may help to understand how the website has evolved over the past years by analysing the publication dates of the recipes. This means that if there are enough recipes to establish a weekly or monthly interval over the last five to ten years, the data could provide a useful time series for further analysis. However, it is difficult to say a priori, if these approaches will yield usable data and relevant results, as certain effects could influence the data significantly, making it difficult to make valid assumptions. For example, such effects could result from the following cases:

- Data points do not occur regularly and frequently enough to create a consistent time series, which could lead to many empty time slots, degrading the quality of analysis results. This is rather unlikely with the amount of data that is available for this project.
- The period of time covered by data points could be too small, leading to a time series that is too short for meaningful analysis (only six time steps over six months).
- The websites' recipe collection grows so fast that a single recipe will be forgotten by the community in a few days. In this case, changes in the popularity of such a single recipe (with regard to its popularity metrics) could not be analysed.
- The user activity on the website is too low to generate distinguishable changes over the chosen time interval, which could happen if only a small number of recipes is being published per month or too few users utilise functions like rating a recipe or marking one as a favorite.
- Website traffic could be heavily distorted by external effects, which might be difficult to identify. These effects could be caused by TV shows or commercials, for example.
- The changes on a website could either be completely random without a consistent user base publishing recipes, or not showing any trends if recipe authors "dump" all their recipes on the website in a matter of a few days and then rarely return to the community (independent from time of the year or food trends).

Effects like these can occur for a multitude of reasons and make time series analysis difficult or impossible by biasing the results. However, if identified correctly, such effects might still provide insights about the user base and thus lead to different types of results that can still be relevant for the project. In any case, the trend analysis has to be performed with caution and awareness of the magnitude of effects that can distort the results.

2.3 Data Science and Data Mining

Data Science is an inter-disciplinary domain, which combines multiple fields like statistics, mathematics, and computer science. A data scientist's main goal is to extract information and knowledge from large data sets, which often already exist and are used or generated for a different purpose. In principle, the field contains all topics that are related to its main goal, e.g. techniques and frameworks for collecting, storing, processing, distributing, and analysing data as well as for presenting the results.
Usually, the data sets are larger than those used in traditional statistics, as they are not gathered as primary or secondary data manually by a single person or research group, but instead are (automatically) generated in the daily process of digital services and applications, driven by a large amount of users and events. For this reason, the established techniques in statistics for many years need to be enhanced by mathematics, learning algorithms, and computational ecosystems to ensure their

scalability and effectiveness. Due to these factors, data science or data-driven science has been proclaimed by Hey et al. (2009, [22]) to be a new, fourth paradigm in science, existing next to the better known experimental, theoretical, and computational sciences.
How much the amount of data has grown can be seen in many everyday use cases. For example, Twitter users are posting approximately 500-700 million tweets per day (in 2016 [31]), each of these tweets expressing a feeling or reaction, or notifying the user's followers about recent news. The tweets rapidly accumulate to a large amount of data, requiring the use dynamic databases that support streams of data. Using the Twitter API[7], these millions of text snippets can be accessed, downloaded, and analysed using data science techniques to, for instance, gain knowledge about the users' sentiment [19] or to predict stock prices [36]. The emergence of data science has already made an impact in our society, as recent presidential campaigns in the United States of America have used data from social networks to perform psychological analysis (or psychometrics [50]) to understand and influence the population and win elections [20].
The characteristics of data science have been defined years ago by naming the main attributes of what has induced it: large amounts of data, or *Big Data*. In a Gartner report from 2001, Doug Lancy has named the three V's of *Big Data* [30]:

Volume: Even though a particular size of what *Big* means is difficult to define, as it could be hundreds of MBs, GBs, TBs or even PBs of data. Any way, a high volume of data results from the accumulation of many data points (e.g. 100,000s of recipes) or large files (images or videos for each entry), for example. Certainly, the total number of data points (N, from a statistical view) and/or the (file) size of each data point surpasses those of traditional empirical experiments by a multitude.

Velocity: Large amounts of data are often generated or collected in a fast manner. A higher speed with which new data points occur can increase the accumulation significantly. Examples are data feeds from sensors or cameras which can record multiple times per second, and also streams of user generated data in social media (as already mentioned).

Variety: In probably every use case, big data consists of data sets of many different types and shapes. Not only can data points have different data types like string, integer, floating point, or Boolean, each entry can also have different formats, like different date formats for each country/region or alternating combinations of data types per entry. For example, some online recipes may provide one or more pictures, while others even include tutorial videos, all of which could be embedded just below the title (as presentation images for the finished dish) or between the preparation steps.

This project deals with multiple datasets that include more than 200,000 recipes each (as seen in Table 1), which all contain different data types like integers, single words, long texts, images or videos. Thus, with regards to volume and variety, these datasets can be seen as *Big Data*. However, the velocity with which new data points (recipes) occur and existing ones change strongly depends on the traffic and dynamics of each website, which have to be further analysed in the following

[7] accessible through: `https://developer.twitter.com/`

chapters, before a conclusion can be made. Even though there are more than 50 million visits per month, this does not mean that those visits result in dataset changes.
In the past years, the initial three V's have been extended to four [23], five [34], seven [58], or in an extreme (and not completely serious) case to 42 V's [52]. Even though not all of these V's seem practicable, there are some that should be mentioned, as they further explain the impact of data science:

Veracity: There is always a chance that the generated/collected data might be biased or false, as the sources providing the data might be faulty or being interpreted in a wrong way. Users of a service might be using false names to maintain anonymity or publish fake stories to attract attention and acknowledgement. In social media, recent years have brought large amounts of false or misleading news that contain wrong information to influence and polarise the population. Recipe data does not have that much potential to be abused politically, but still could be used as means of marketing, e.g. for a company's products, culinary celebrities, or TV shows.

Value: Data science processes can consume many resources as the process may not always be simple. Still, many businesses are not only integrating data analysis into their daily activities, but even building new teams and departments to specialise on these tasks, which leads to building new features, services, or products based on the insights. This indicates that the value created is worth the cost. Creating value from the data science process is also a main goal of this project. A more precise definition will be provided below in section 5.2.

Data Mining describes one part of *Data Science*, which is focused on pattern discovery from large data sets. The discovery of patterns often follows the principles of exploratory data analysis, which has been established by Tukey (1977 [56]), utilising different statistics and *supervised* or *unsupervised learning* algorithms to find insights in data. While traditional statistics is rather focused on setting up hypotheses first, then collecting and analysing data to confirm those, Tukey proposed to use existing data to explore and suggest new hypotheses that can be tested. This fits well to the modern approach of *Data Science*, which aims to learn as much as possible from existing data sets.
This means that data mining can include simple exploratory techniques as described in section 2.1, as the disciplines in *Data Science* all overlap to a certain degree. However, data mining techniques can be distinguished from exploratory techniques by regarding *learning algorithms* as their main component for discovering patterns. Data mining has been defined by Fayyad et al. in 1996 ([14]) as one step in the process of *Knowledge Discovery in Databases* (KDD), an exploratory approach to databases that has been around since a first workshop in 1989 [44]).
According to this definition, the term data mining includes following tasks:

Classification: The data points are all categorised into classes by using their features. For training, the classes are set a priori as an attribute by the supervisor (or user). Then, the model learns to find the correct class based on the combination of all other attributes. For example, the type of a vehicle (motorcycle, car, train, etc.) can be predicted by comparing each vehicle's dimensions, weight, motor type, and performance.

Regression: The features of a data point are used to predict a numerical target variable. During training, the supervisor provides the target variable, that the model can learn with and fit a continuous function to. For example, a person's lifespan could be predicted using multiple measurements from health examinations.

Clustering: This technique is used for dividing data points into groups that are similar with regards to (some of) their features. The training of a clustering model does not require predefined target classes or groups, but instead the model finds similarities "on its own". The result is a representation of the underlying features that can also be used to allocate new data points in one of these clusters.

Dimensionality Reduction: Each additional feature in a data set adds another dimension, which can lead to a high number of dimensions that requires large amounts of computational power to be analysed. This is why techniques like *Principal Component Analysis* (PCA) and others [57] are used to find the few dimensions that are independent (or least correlated) and thus the most significant. The results provide a data set with reduced dimensionality for faster analysis. There are many ways to reduce a dataset's dimensionality and sometimes a simple solution can be sufficient.

Anomaly detection: This task focuses on finding outliers in the data set compared to previous and predicted values. A well established use case is fraud detection [13], which is used, among others, for fighting credit card theft. Sometimes outliers can be useful as well, e.g. for finding recipes that have extraordinarily high popularity and ratings.

While classification and regression are categorised as *supervised learning* - which requires a prior definition of target classes or variables before training the model - clustering, dimensionality reduction and anomaly detection are categorised as *unsupervised learning* techniques, as the model learns distinguishable classes or outliers without a prior definition of targets.
However, the nomenclature in this field is far from being distinct. These learning tasks have also been included in topics like *Statistical Learning* [18], which focuses on a more detailed statistical and mathematical interpretation of these tasks, or *Machine Learning* [38], which extends older pattern recognition techniques with more complex models. An example for these models are neural networks, which have been around for many years as well, but only have become feasible after recent increases in computational power. The term data mining has been rather popular in the rise of KDD in the late 1990s and following years, but nowadays, *Machine Learning* (in connection with *Artificial Intelligence*) has gained even more popularity, as it has permeated products and services and thus reached mainstream [2]. The techniques of these fields have become much more accessible through online learning possibilities [28].
Still, as technology is advancing and trends are changing in an increasing pace, the terms data mining and *Machine Learning* may soon be or already are replaced by the buzzword *AI* (for artificial intelligence), but still be describing the same models and techniques that are being used in everyday analysis.

2.4 Cross-Industry Standard Process for Data Mining

The *Cross-Industry Standard Process for Data Mining* (CRISP-DM) is a process model that has been established to standardise the implementation of data mining (DM) in businesses [60]. The model has been generalised to be independent from industry sectors as well as the technology used. The process provides an overview of the steps necessary to perform a data mining project leading to a result that creates value for the business. Such projects can help to tackle problems in a new and more efficient way, enabling the business to scale or be more secure, for example.
CRISP-DM is only one of multiple process models for data mining. However, it has been proven to be the easiest to understand and implement, as its development by a consortium of companies (DaimlerChrysler and IBM amongst others) has been focused on practicality [60]. Also, research has shown that other, more complex process models can be condensed to the same amount of steps as the CRISP-DM model [29]. Additionally, this process model values a customer-oriented business model, even advising to present results in a customer-oriented way, as user experience and satisfaction is important for customer retention and sustainable success [45]. Thus, this project will utilise the steps of the CRISP-DM model as a guideline during implementation. The model consists of six steps (as described in Table 2, which can be performed in a cyclical way, with each cycle delivering a new and improved iteration of the deployed project. Due to the theoretical setting, this project will use the CRISP-DM model as a guideline, but will not pursue each step completely, as some of them require a running business and infrastructure for deployment. As this project is not being implemented on behalf of an existing business, the setting will be in a hypothetical company, which will be explained in depth in section 3.1.
Despite this project's hypothetical setting, the CRISP-DM model provides valuable advice on how to structure a project of this kind, e.g. starting by setting business and data mining goals, which require a deeper understanding of the setting and use case before moving on to the next steps.

Table 2: Steps of the CRISP-DM process model

Step	**Description**
Business Understanding	Define project objectives and business requirements, convert them into a DM problem definition.
Data Understanding	Explore data, select interesting subsets and identify problems in data quality.
Data Preparation	Finalise the dataset that will be used in the DM tool by cleaning, filtering, transforming and constructing new attributes.
Modeling	Apply the DM method(s) to the prepared data and tune parameters to improve performance.
Evaluation	Evaluate the obtained results from a business perspective, assessing their value for the business.
Deployment	Present the discovered knowledge in a customer-oriented way. If the project aims to produce software for further usage, deploy and perform monitoring and maintenance.

This project will use these steps as a guideline and pursue practically all of them in section 3: After defining the business and its requirements, business goals can be set and applied to data mining goals after the data understanding. Then, the data will be collected, prepared and analysed, which includes building models as well. During the evaluation, the results will be presented in a customer-oriented way. The only exception is the deployment, as explained above. That stage would actually require providing a product or service, which goes online on a company infrastructure and then is being monitored and maintained.

2.5 Business Value

As the title of this thesis states, the aim is to create *Business Value*. However, it might be unclear how this term is defined, as value in general is quite abstract and it is difficult to find a single definition of how to create value in a data analysis or mining project. For this reason, an independent definition will be proposed in this section to provide an understanding of the project's main goal.
To find a definition, the knowledge from the prior sections can be utilised, which leads to four different ways of how value can be created in the context of this project. The previous chapters have shown that:

- the gathered data can be used for economical (or other) benefit, as third parties might be interested in accessing them, e.g. for commercial or scientific use (section 1.1),
- the data analysis can enable a business to build new features, services, or products, which then generate new revenues (section 2.3),
- data mining techniques can be used to tackle internal/organisational problems and increase process efficiency, which is why businesses have a strong interest in implementing these techniques (section 2.4),
- customer-oriented businesses can use analysis results to improve the user experience and satisfaction constantly to achieve sustainable growth (section 2.4).

Consecutively, *Business Value* can be defined in the context of this data analysis project as the part of a result, which can lead to creating a valuable dataset for further usage, building new product features for a better user experience, or improving efficiency and capabilities in internal processes. For a concluding view, section 5.2 will summarise the results of the project in regards to these criteria.

3 Implementation

This section will present the frameworks, packages, and tools used in each step to perform this project, as well as explain implementation details with code examples. The Python code for this project can be viewed on GitHub [7] and is freely available for download.

3.1 Business Case and Goals

For this project, the perspective of a hypothetical business in the food service sector will be taken, inspired by multiple start-ups that employ similar business models in Germany, like HelloFresh[8], Kochhaus[9] or Marley Spoon[10]. More specifically, this young business will offer the delivery of groceries in boxes that are already assembled and portioned to serve the customer all ingredients for a particular recipe of their choice, which can then be prepared at home for a fresh meal. As an example, one customer might order a box with spaghetti, tomato sauce, fresh vegetables and salad and then prepare a meal of spaghetti with vegetarian bolognese and a salad for two persons. To provide such a service, this business needs a cooperation with multiple supermarkets and their delivery services, as well as a rich offer of recipes and features that satisfy and enthuse the customers. As there are multiple competitors in this sector already, this business needs a competitive advantage that this project aims to develop. Optimally, this service should inspire its users to cook more often, try out many high-quality recipes and enjoy cooking them.
To specify what the project should deliver, particular goals for this project can be set using guidelines from the CRISP-DM model explained in section 2.4. The first set of goals are business goals, which are high-level objectives. These goals focus on results that create value according to the definition described in section 5.2 and thus could affect the revenues directly or indirectly. The business goals have been summarised in Table 3.

Table 3: Business goals

Goal	Description
B1	Create a recipe dataset that can be considered a valuable asset
B2	Understand users by finding patterns on different recipe platforms
B3	Gain market insights by comparing cross-platform results
B4	Develop at least one new feature to gain a competitive advantage
B5	Propose future project ideas that extend the scope of this project

The second set of goals are data mining (DM) goals, which pursue the technical steps for implementation that are the key to achieving the business goals. The DM goals need to specify the tasks that are necessary to prepare and deliver the results. For this, the technical requirements in the DM goals need to be considered, which are specified in Table 4. This table has a two-dimensional indexing, which serves the purpose to show which data mining goals are connected to which business goals. This means that to fulfill B1, D1.1 and D1.2 have to be completed.

[8] `https://www.hellofresh.de/`
[9] `https://www.kochhaus.de/kochboxen/`
[10] `https://marleyspoon.de/`

For simplification, some DM goals have been condensed into fewer points, for example D1.2, which contains cleaning and preparing the dataset. In a standard software development process, these steps would be split into separate goals for clarity, as each step requires different skills and its responsibilities might be assigned to a different team member. However, this is irrelevant in this project (as tasks are not being split up) and the grouping of linked steps facilitates comprehensibility.

Table 4: Data mining (DM) goals

Goal	**Description**
D1.1	Implement scraper and extract data from different recipe platforms
D1.2	Clean and prepare dataset for analysis
D2.1	Implement and perform exploratory analysis
D2.2	Implement and perform trend analysis
D2.3	Implement and execute DM techniques
D3.1	Perform multi-page analysis for global trends
D4.1	Evaluate the results of each analysis technique
D4.2	Define ways how techniques can be used to develop new features
D5.1	Assess further techniques for future potential

3.2 Framework and Libraries

The code for this project has been written in the Python 3 programming language [16]. Python is easy to understand and useful to create small prototypes and proofs of concept, as the interface is simple and effective. Additionally, its open source community is highly active, delivering many useful packages for a variety of purposes. In this project, the package *scrapy* [51] has been used for extracting data from the source websites. This package has a large and active community, which is constantly working on improving the package by fixing bugs and adding new functionality. There are many tutorials available online, which show examples of how to use *scrapy*, like Duke's [10] and Rizvi's [48] blog posts. The official Scrapy documentation[11] can be useful to understand the framework as well. Of course, there are also other ways to scrape websites, like downloading all pages manually and then filtering the important data with regular expressions or using a different tool like *beautifulsoup* [47], which is not as popular and has a smaller developer community. The Python package *re* [17] has been used for implementing regular expressions, which is a small programming language embedded in Python for setting rules to match certain string patterns to an input (e.g. words or letter-symbol combinations). These expressions can then be used to filter e-mail addresses from a text, for example.
After extraction, the data has to be stored in a format that allows further usage and analysis. For this purpose, the extracted data has been exported in the json file format [26] and stored in a MongoDB database [11]. The MongoDB database supports document-based storage methods and is able to store collections of documents that are not necessarily in a uniform structure or may change over time (in contrast to a traditional relational SQL database, which is based on the SEQUEL language

[11] `https://docs.scrapy.org/en/latest/`

published in 1974 [8]). This means that every recipe will be handled as a single json object, which can be accessed individually. Thus, the different recipes can be handled in a more flexible way, as the data fields can vary among json documents and can change over time. MongoDB also provides tools for basic analysis, but as the extracted data needed manual cleaning first, these functions have not been utilised. For data cleaning and analysis, the json files can then be imported into a Jupyter Notebook file (formerly IPython) [43], which provides an environment for writing short code snippets in Python. This environment is perfectly suitable for doing a step-by-step data analysis, as it supports a stepwise execution of a few lines of code at once. In Jupyter, all existing Python packages can be imported and used the same way as they are in a traditional Python script. The main packages used in this project during analysis in Jupyter are *json* [15] (for importing and exporting .json files), *pandas* [41] (for more efficient analysis using a data structure called DataFrames), *matplotlib* [24] (for creating histograms, scatter matrices and boxplots), *scipy* [27] (for calculating correlation coefficients) and *scikit-learn* [42] (for machine learning algorithms).

3.3 Scraping the Web Pages

This section is only intended to be a short introduction into the syntax of *scrapy* and the challenges during development. The actual implementation of the scrapers for each website can be found on the GitHub page for this project [7].
Scrapy provides a powerful tool for extracting data from websites that do not offer an API for accessing their data. For implementing a scraper with this package, one has to get familiar with each website's HTML[12] code. In the case of the selected recipe websites, the scraper starts from the overview page of all recipes, which is the page where one can see the list of all recipes with picture, title and link. As there are more than a page-full of recipes on each websites, these lists need to be paginated, which leads to up to 29,000 pages of recipe lists that have to be iterated. To process all pages, the scraper works through the list and opens the link for each recipe entry, where all specified data fields are then extracted. Once a page is finished, the scraper moves to the next page using the link on the bottom of the website.
As the HTML tags define the position and role of each piece of text or other content hierarchically, the structures can become deeply nested and unclear. To address this issue, *scrapy* provides two different ways to access fields, of which both have their own use cases. One way to access HTML tags is with the CSS[13] name, which is the cleaner way to go, but also requires the website to be built cleanly. This means that each value on the website (e.g. date, title or rating) should have its own separate data field with the respective HTML tag. As an example, when a section (in HTML called `div` for division) on a website describes the recipe's preparation steps, then this division should have a class name assigned that describes this content, e.g. "preparation-steps". If the website has been set up correctly, then the text contained in this section can be accessed with the CSS command `css('div.preparation-steps::text')`, with `::text` expressing that the field's text attribute has to be accessed. If interested in other attributes, for example a link, `::attr(href)` can be used, as this describes links to other pages in HTML.
However, two out of the three selected websites (chefkoch and kochbar) have been

[12]Hypertext Markup Language - a language that is used for programming websites
[13]Cascading Style Sheets - a format for specifying layouts and designs for HTML websites

built in such a way that most of these tag class names are missing or inconsistent, making the structure unclear and difficult to access. This challenge can be tackled using the second way that *scrapy* provides for accessing HTML tags: the XPath address. Addressing tags with the XPath command requires their exact position in the HTML code's hierarchy. For example, a page has the division `article`, which contains three divisions called `textbox` and the third of these textboxes contains another division which holds the recipe's preparation steps. To access exactly this position, the identifier `/div/div[2]/div/text()` needs to be used, which leads to the text content of the respective field. As there needs to be a starting point when addressing like this, the XPath command requires an ID of the position where it should start. This ID needs to be unique, as otherwise, the code would produce an error instead of giving a result. In this project's case, one can start from the highest hierarchical level, which is called `kb-global-content` on the kochbar pages. In the next step, one can access the preparation steps with the following command: `xpath('//*[@id="kb-global-content"]/div/div[2]/div/text()')`.
Another challenge besides dirty HTML layouts is that recipe pages on the same platform may differ in their structure and "dirtyness". The reason for this is the fact that recipes can have different lengths and varying numbers of ingredients or preparation steps, which will change the page layout if the page has not been set up correctly. Alternatively, some users might provide more content for their recipes by adding pictures, sometimes even between the preparation steps. Because of this, the scraper has to be custom-made for each website and also adapted to all deviations that can appear on a single domain. This means that all possible errors need to be spotted and eliminated, which can require many hours of work and be quite difficult, if there are 100,000s of recipes that need to be checked. As an example, the addresses used in the XPath command from above might differ depending on if the user has added pictures to the preparation steps or not, possibly moving the text into the next division. Then, during a first test the scraper's log in the console will show that there have been errors on some pages, where the used command has extracted nothing. To solve this, all deviations among the webpages have to be found by testing and covered by rules that address each case. A similar approach needs to be pursued if some of the recipe's attributes have missing values. For example, many users do not provide the amount of time needed to finish the recipe, which leaves the field blank or even removes the field from the affected recipe page. This can again lead to a changed structure, which is why missing values need to be treated appropriately. The worst type of error can occur if the structure of the recipe page changes in such a way that the scraper still collects a value, but a wrong one. In that case, the scraper will not give an error message while scraping and the faulty data fields can only be detected later during data cleaning. Unfortunately, this has happened with the chefkoch dataset, which will be explained in more detail in section 4.1.1.

3.4 Data Cleaning and Structure

The cleaning of the extracted data is essential for producing a usable and valuable dataset, which is one of this project's business goals on one hand and also inevitable for further data analysis on the other. Usually, data cleaning consumes the most time as it requires an understanding of the data and the application of many different manipulations and conversions to achieve a clean dataset. During this process, one always discovers parts of low-quality in data and needs to find solutions to provide a feasible dataset for further analysis. For this reason the cleaning has to be performed as early as possible - preferably already in the scraper, even before storing the data. Most of the text fields on websites contain multiple line breaks and blank spaces between words, which decreases the dataset's quality and impedes readability. Thus, the scraper already performs some data cleaning by removing redundant blank spaces and all line breaks, so that each text field can be handled as one coherent string.
In the next step of the analysis, the data is loaded into a Jupyter Notebook for further cleaning. A brief evaluation of the data shows that (especially in the recipes from kochbar) recipe titles and preparation steps can include multiple symbols like hearts or stars, which the authors seem to use for highlighting their recipes. Unfortunately, this impedes text mining and analysis, which is why these symbols need to be removed manually. Additionally, all line breaks and redundant white spaces that were not detected during the first cleaning are removed at this point. Further preprocessing is performed by changing the separator in decimal numbers from the German (x, y) to the English version $(x.y)$, which is necessary for interpreting these values as numerics in Python. The largest share of dirty data can be found in the dataset from chefkoch, where practically all data fields that should contain numerical data are filled with words or characters, which need to be removed in a multi-step cleaning process.
The structure of each domain's dataset varies, as the websites have some differences in the information they hold. In general, each dataset has been organised as a list of Python dictionaries in the scraper's output. The dictionary resembles the contents of a recipe, with fields like "name" and "ingredients" containing the respective information. This format is also suitable for storing the data in .json files, which is the main format for the document-based MongoDB storage. An overview of the data structures for each domain can be found in Table 5. This table also shows how different the domains are in respect to the data fields they offer per recipe. Common attributes of all websites are highlighted in emphasised font. Obviously, these are data fields that are obligatory for a recipe, like name, ingredients, and preparation steps. Also, all websites have employed a rating system, which is essential to assess each recipe's quality. It is curious that all three websites also support the display of calories for each recipe, which should be quite hard to calculate precisely for a private user. Hence there could either be a built-in mechanism in these websites for calculating calorie counts based on the ingredients or the values will be provided by the users and thus be imprecise. This has to be investigated during exploratory analysis.
Aside from the common attributes, the rest of the attributes can be quite different between each domain. While chefkoch provides detailed information about the time (for preparation, cooking and resting) and metrics about the recipe's popularity (number of times printed, saved and shared), the domain lacks basic attributes like

number of clicks/visits or total preparation time, which need to be calculated or estimated separately. Additionally, some of the attributes are only filled very sparsely, for example, the total time in hours provided by kochbar (only a few hundred entries in more than 330,000 recipes) or total time in minutes provided by eatsmarter (also only a few hundred entries in more than 200,000 recipes), which renders them practically useless for any analysis. To avoid producing wrong or biased results with incomplete data, all data fields should be checked for missing and dirty data in the beginning of each analysis. It should be noted that kochbar contains an attribute that shows how many times a recipe has been marked as a favorite by a user. This attribute is quite useful and will be part of later analysis steps. For simplicity, it will be named "Favorite(s)" in the further sections.

Table 5: Data fields provided per domain
(common attributes are ***emphasized***)

Attribute	**chefkoch**	**kochbar**	**eatsmarter**
Average rating	✓	✓	✓
Calories	✓	✓	✓
Clicks (or visits)		✓	
Comment number		✓	✓
Date	✓	✓	
Difficulty	✓	✓	✓
(Marked as) Favorite		✓	
Images for each preparation step			✓
Ingredients	✓	✓	✓
Name	✓	✓	✓
Number of rating votes	✓	✓	✓
Preparation steps	✓	✓	✓
Price of ingredients		✓	
Printed/Saved/Shared	✓		
Subtitle	✓	✓	✓
Time (for preparation)	✓		✓
Time (for cooking)	✓		
Time (for resting)	✓		
Time (total, in minutes)		✓	✓
Time (total, in hours)		✓	
User name	✓	✓	
User registration date	✓		
User activity info	✓		

3.5 Feature Extraction

For most learning tasks, it can be very useful to extract further features from the data, or convert existing attributes into more easily usable features, e.g. using the text fields like recipe title, ingredients, or preparation. This is due to the fact that full titles of recipes, for example, can hardly be compared to another, as they all have different lengths and may use various words to describe the same things. Thus, the data will be much easier to compare when brought into a standardised form by extracting features that have the same shape and easily comparable values.
Feature extraction can include dimensionality reduction to reduce long recipe titles into a single word or it can add more dimensions to the dataset by splitting up a single attribute into multiple valuable columns or even generating new attributes by using an *unsupervised learning* model.

For this project, three ways of feature extraction will be used that are common in data mining:

- Splitting up multi-word titles into a list of single words, so that all words in the recipe titles can be used for further analysis, e.g. word counts. This is a simple way of dimensionality reduction, but can yield valuable results.
- Cleaning ingredient lists of each recipe and encoding them by splitting them up into single columns with a binary index (one-hot encoding) can provide valuable features for ML tasks. While this increases the dataset's size exponentially, the produced features can be used very easily for indexing and search, as well as data mining algorithms.
- The preparation steps can also be used to create new features (e.g. word embedding vectors), which can help in classifying the recipes. Word embedding models use the dataset's vocabulary to create a high-dimensional space, in which similar words and entities are placed close to each other.

Splitting up the recipe titles is required to perform an actual analysis of the recipe titles. As different users can name the same or similar recipes very differently in terms of title length, word choices, additional explanations, etc., one has to create a standardised format to enable comparability. For this purpose, the titles will be split up into lists of single words, which also helps in removing unnecessary stop words (i.e. common words like "and" or "the" which do not help in understand a recipe's content). Thus, each title can be reduced to a list of keywords that help to identify the ingredients of each recipe quickly. As many recipes only have a single-word title, the solution with best compatibility is achieved by selecting the single word out of a title that is the most frequent one in the vocabulary. This means that if a recipe is called "Cheeseburger with Sweetpotato Fries" and, out of these four words, "Cheeseburger" has the highest occurrence in the dataset's vocabulary, the title will be reduced to that word.

One-hot encoding is a technique that is used very frequently when extracting features from data, as it is compatible with practically any text data. The principle is very simple: One creates a separate column for every value that can appear in the selected subset. This means that if the dataset has 100 distinct words in the column "name", one can create 100 new columns that are named after each of those distinct words. After that, one needs to iterate over all entries in the dataset, during which a "1" is set into the respective column of every word that occurs in one entry. A basic example for this can be seen in Table 6.

Table 6: An example for one-hot encoding

List of Letters	**a**	**b**	**c**
(c)	0	0	1
(a, b)	1	1	0
(c, a)	1	0	1
(b, a, c)	1	1	1

As one can see in the example, one-hot encoding does not consider the order of the values, which is not problematic for recipe titles or lists of ingredients, because they are usually short and only consist of a few words. This type of encoding can also be utilised for searching, as one can easily filter by one column to search for all data points which contain the respective word. One-hot encoding works very well with different sequence lengths, as they are provided in the ingredients lists: some recipes only need three ingredients, while others require many more. The missing words are then simply filled with zeroes, which enables easy comparability. However, the larger the vocabulary grows from which the encoded words are chosen, the more columns are needed to encode the data accurately, which leads to an exponential increase in complexity and memory demand. In the kochbar dataset, for example, there are 175,034 distinct words used in recipe titles and 107,623 distinct ingredients. In order to featurise[14] the whole vocabulary or a longer text, like the preparation steps, using a word embedding technique like the recently published FastText model [4] should be considered.

Word embeddings are designed for converting a given vocabulary into a vector representation. This encoding results in a model that provides a single vector for each word. This vector can have 100 or more dimensions and represents the semantic position of that word in the feature space of the selected vocabulary. This means that all words from the vocabulary will be projected onto a high-dimensional feature space, in which similar words are placed closer together. The similarity of words is being detected by the way they are embedded in sentences, which means that words that are usually written next to the same words and have a similar function, will be positioned closely together in this space. For example, the words "cook" and "fry" are used in a similar way across different recipes, which is why their vectors are likely to point at close locations. Using such a framework, one could train a model on the complete texts of Wikipedia, which has been done by the authors of FastText in many different languages [35]. The main advantage of word embeddings in comparison to other encodings like one-hot encoding is the capability to encode the sequences that usually come before and after one word, which enables them to featurise not only the words, but also their context. Using this model, analogies between words can be found, often explained with two popular examples:

- $man + king - queen = woman$
- $Paris + France - Germany = Berlin$

The accuracy of such analogies depends on the text corpus that is used in the dataset as well as its quality and size, of course. If the word embeddings are trained on scientific papers in physics, the model will not be useful to find sensible analogies in recipe data. However, it could be an interesting option to train word vectors on the preparation steps in this project's datasets and then apply the trained vectors when processing the recipe titles.
In this case, a precise language model to find analogies between words is not necessary. Instead, these embeddings can be used to place similar recipes closer together.

[14] i.e. convert into quantifiable features

3.6 Data Analysis Tools

There is a vast landscape of different tools that can be used for data analysis, each with different benefits and disadvantages. Two restrictions that disqualify most analysis programmes are their capability of handling large input files and their compatibility with specific data formats. For example, Microsoft Excel is still being used by many companies for data analysis and rule-based models, as many employees are using the programme in their every day work already. However, not many datasets are suitable for Excel format, as it supports only a limited number of data points, and tables with many columns can quickly lead to very large file sizes. Of course, working on a *Data Science* project, one would not want to restrict the options to Excel's capabilities anyways, which is mainly a program for processing tables with limited functionality for feature extraction and pattern discovery. In principle, the following three data analysis frameworks can be considered as suitable for a project like this:

- Jupyter Notebook, as already introduced in section 3.2, is a powerful environment for prototyping in Python. Not only can it be used for data cleaning and feature extraction, but also for many types of analysis and visualisations, perfectly suited for creating tutorials as it also supports the Markdown format for writing structured texts between lines of code. Being based on Python, Jupyter is the most complex tool for analysis in this selection, as the programming language has to be understood well to handle different data structures and packages for analysis. In return, this framework offers the highest flexbility and compatibility.

- RStudio [49] is a popular software with graphical interface for using the statistical (pseudo-)programming language R and the main open-source tool for many researchers that perform various types of statistical analysis. R provides a simple and declarative programming interface, which enables users that are not experienced in programming to perform effective analysis. However, as R is designed for simplicity, it does not offer the flexibility to handle many different data types or file sizes, as it mainly supports text documents or text-based table formats (e.g. comma- or tab-separated (csv or tsv) files) and performs poorly when working with large data sets that exceed multiple hundred MBs, especially if the dimensionality is high.

- Tableau [54] is one of the most popular business intelligence tools, providing a large number of interfaces for different data formats and easily usable analysis tools in its graphical user interface that can generate attractive visualisations. Being focused on producing well presentable dashboards and overviews, Tableau's capabilities of data cleaning and supporting large file sizes are limited. Thus, most of the cleaning has to be done in advance and sometimes workarounds have to be found to be able to feed all the input data into the programme. Still, Tableau can also be useful for exploring data, as the software is able to recommend analysis techniques or visualisations that one might not think of in the beginning.

Even though there are many other analysis tools available, these three options represent the landscape of available frameworks: a normal programming language with suitable packages (Python / Jupyter Notebook), a simpler framework with declarative language that focuses on statistics/mathematics (RStudio), or a business intelligence tool that aims to provide great visualisations and quick deployment of popular analysis techniques (Tableau).
Unfortunately, RStudio's limitations in compatibility and scalability make its use unfeasible for most analysis steps in this project, as initial testing has shown. While there are useful functions in R that could be employed, e.g. easy-to-use time series analysis techniques, this framework requires the data to be cleaned and prepared completely before importing it into R. The main drawback is that json files are not supported optimally in R, forcing the user to try multiple packages with different APIs and functions, while adapting the code manually each time until a package is found that is compatible with the modified data set. At the same time, converting the recipe dataset into a csv file (which would work better in R) is only an option after extensive data cleaning, as the recipe texts contain many different symbols, making it difficult to select a separator symbol, which does not occur in the texts, for creating the file format. However, even after cleaning, R has shown great difficulties in handling the generated datasets, as they are quite large (e.g. the one-hot encoded feature set with 1,000 binary dimensions requires 1.8 GBs of space). Due to this fact, R has not been a suitable option for performing a more complex analysis on the generated feature sets - one of the main tasks in this project. This leaves only the tasks of analysing correlations and performing regression or similar tasks that use numerical inputs to be done in R, which all can be performed in Python with similar or less effort. Thus, most of the work (especially data cleaning) needs to be done in the Jupyter Notebook either way, making it more reasonable to perform the rest of the analysis in Python as well, saving the effort of setting up another framework and work environment. Additionally, Python provides better performance when handling large amounts of data, as there is a multitude of packages designed for scaling complex computations by using pipelining with multiple CPU threads or even GPU acceleration[15]. One other advantage of Python is the vast availability of original implementations of state-of-the-art algorithms, for example t-SNE (t-Distributed Stochastic Neighbor Embedding) [33], which is a popular tool for projecting high-dimensional data onto a two- or three-dimensional space for visualisation.
Tableau on the other hand, even though limiting json file imports to a maximum size of only 128 MBs, might still provide valuable visualisations that look better than those built in Python. Due to the restrictions, the data set still needs to be prepared by using different file formats or using only subsets of data. Like R, Tableau only provides limited capabilities to clean dirty datasets, which again requires the cleaning to be done in Python. Creating more complex plots like a scatter matrix with kernel density estimates require many steps in Tableau, while the same plot can be generated with a single line of code in Python. Unfortunately, Tableau does not support most of the data mining techniques like hierarchical or density-based clustering, because its functionality is only limited to a few techniques.

[15] A CPU is a personal computer's central processing unit, while a GPU, the graphical processing unit, is part of the graphics card and specialised in parallel computations due to its higher number of processor cores.

4 Analysis Results

This section will describe the most relevant insights obtained during each step of the analysis. Further analysis results that have not been used in this summary can be found in the Jupyter Notebooks, which have been uploaded to the GitHub repository [7].
First, the single-page exploratory analysis will be presented, which has been performed on each website's data set separately. The goal of this initial approach is to understand the statistics of the data (e.g. distributions and correlations), gather ideas and evaluate features for deeper analysis in the next steps. During exploration, the used techniques provide information about the quality and shape of the data, as well as the distributions, which is important to decide on which techniques can be used during further evaluation.
Second, the trends on each website will be analysed by comparing the changes in data over time, using the data sets that have been collected in a monthly interval over a period of six months, which has produced six snapshots of the databases. Over this time period, each website has seen user activity, which helps in analysing the website's traffic and identifying trends. Additionally, a second approach to create time series data will be evaluated by using the publication date of the recipes for the time axis. Following the time series analysis, section 4.2 will compare trends among the different websites in the search for "global" patterns.
Finally, the results of the data mining techniques selected for this project will be assessed and new product features, which are based on the analysis results, will be proposed.

4.1 Exploratory Analysis

The exploratory analysis is not only useful for developing a statistical understanding of the underlying data's properties, like central values (mean, median), distributions, or correlations, but also includes extracting specific information from the dataset by using queries. This means that general properties of the data will be viewed during the exploratory analysis, but this can lead to finding data points that can be interesting for the business case, too. For example, one might need to find the most interesting recipes for this business using more than just the total number of clicks. To achieve this, exploratory techniques can be used to find out which attributes correlate with a recipe's success. Thus, the main questions in this section are:

- What can be learned about the user activity on these websites?
- How is the success of a recipe measured?
- Which techniques can be used to select the most successful recipes?
- How can recipes be found that have a consistently high quality?

To answer these questions, a top-down approach will be taken by starting with general platform statistics, then moving to the dataset on a user level and eventually analysing single recipes. Early analysis has shown that kochbar is providing the most valuable and usable dataset, while chefkoch contains many dirty values. Eatsmarter has a clean dataset regarding the text fields, but the recipes lack some basic attributes like date or user name, as all recipes are being published by the website's

editorial team instead of a user community. Still, being one of the largest recipe websites focusing on healthy recipes, the data collected from eatsmarter might still be valuable.

Table 7: Platform statistics comparison

	kochbar	**chefkoch**	**eatsmarter**
Recipes R	411,489	319,578	151,095†
$\Delta R_{m=1}$	-6,740	1,338	-49,291†
$\Delta R_{m=6}$	82,830	5,803	-49,255†
Authors A	18,614	80,459	1**
$\Delta A_{m=1}$	-81	390	1**
$\Delta A_{m=6}$	1,397	1,699	1**
Votes V	7,532,852	2,559,803*	14,808
$\Delta V_{m=1}$	-170,418	19,591*	243
$\Delta V_{m=6}$	1,355,399	130,573*	1,736
total attributes	16	18	12
with missing data	4	4	5
with dirty data	5	11	2
recipes online since	08.03.2008	18.10.2000	not available**

* due to dirty data, these values could be calculated only on a subset

** the eatsmarter recipes are published without a date or author

† significant decrease due to website changes in February

4.1.1 Platform Statistics

Initially, it can be useful to view general statistics of each recipe platform to understand their size and capacity regarding recipes, users and user activity. The attributes that are available for extraction on each website have already been presented in Table 5. However, this does not necessarily mean that all of these attributes are provided for every single recipe or even in a clean and usable data format. Thus, the datasets have to be checked for missing and dirty data as well, which both need to be treated accordingly, depending on the case at hand.
Table 7 shows a comparison of general statistics for each website: total number of recipes, authors (users that have published at least one recipe) and aggregated votes, including their change in one month ($m = 1$) and six months ($m = 6$). These values can help to understand the amount of user activity on each website. Unfortunately, there is no common attribute for all three pages that indicates their traffic (as depicted in Table 5), but the numbers of authors and votes can help estimate how active the user communities are in comparison. Still, the table shows the following patterns that seem quite interesting:

a) Changes in recipe counts, including significant reductions

While kochbar has increased their recipe count by more than 80,000 recipes over the last six months (approx. 25% from October 2017 to March 2018), it seems that more than 6,700 recipes have been deleted during the last month. The reason for this is unknown, but it could be caused by the website administrators cleaning the database by deleting duplicates, low quality contributions or recipes that have

been reported due to inappropriate content. It is also possible that a larger group of users that had published a high amount of recipes deleted their accounts and their published recipes, possibly because they have stopped supporting the website (which again could be caused by several reasons). It also seems that eatsmarter has lost approximately 25% of their recipe database in February. Possibly, this significant reduction occurred because eatsmarter's website structure was changed, thereby reducing the amount of recipes that the scraper can collect. This side effect could be deliberate or accidental - either way it has caused this project's scraper to collect only 75% of the initial dataset from October. Alternatively, it is also possible that eatsmarter did actually reduce their recipe database, possibly by deleting duplicate or low-quality recipes as well. As eatsmarter is restricting searches over the complete recipe set and rather encourages browsing in preset categories, it is difficult to obtain an exact recipe count. However, after growing only by 36 recipes in 5 months, it seems unlikely that the administrators would delete almost 50,000 recipes on such a short notice. Meanwhile, the recipe count on chefkoch has shown a very homogeneous growth, with the monthly change being close to the all-time average growth over the platform's 17.5 years, which is considerably longer than any of the other websites.

b) Significantly different number of authors

Chefkoch has a significantly higher amount of users in comparison to kochbar, even though the total amount of recipes is lower. This means that in principle, the user community is bigger and has a smaller number of recipes per user. The kochbar dataset is driven by a few users that are publishing extraordinarily high amounts of recipes (including official accounts from the cooperating TV channel VOX with up to 10,000 published recipes). In comparison, chefkoch has a more homogeneous user base with more users that have been taking part in the community, even if they have not posted thousands of recipes, but only tens or hundreds. Similarly to chefkoch's increase in recipes, the number of authors has been growing very consistently, with $\Delta A_{m=1}$ and $\Delta A_{m=6}$ being close to the monthly average growth over the total time. The recipes on eatsmarter (which is also a food and recipe magazine that has existed since October 2010) are all published by an editorial team instead of a user community, which means that different authors cannot be differentiated. This has the advantage that one can expect less dirty data and inconsistencies among the recipes, but at the same time the dataset is lacking important metrics, like publication date or popularity metrics. In contrast, the user communities and popularity metrics on chefkoch and kochbar are encouraging the users to publish high-quality recipes to collect many votes and good ratings as an incentive. Even though there is no material or long-term benefit from collecting positive ratings, it is undeniable that users in social networks and communities enjoy appreciation and popularity. Without this incentive, the user activity on eatsmarter is low, with only around 240 users rating the recipes each month, even though there are more than 150,000 (or respectively 200,000) recipes available.

c) Amounts of dirty and missing data

Table 7 also displays the number of total, missing, and dirty attributes. In this context, *missing* denotes attributes that do not hold a value for every data point and *dirty* means that the attribute is either filled inconsistently, thus is unusable,

or requires multiple steps of cleaning before usage is possible. Text fields need cleaning in almost every dataset, because many use cases require texts to be purged of unnecessary symbols and stop words, especially if the data has been created by a user community. From this viewpoint, it is quite remarkable that the text fields from eatsmarter and chefkoch are relatively clean and the ingredients of each recipe are already provided in a list, without any symbols or superfluous characters. However, there are still enough examples for both cases (missing and dirty data) in all three datasets.

In the kochbar dataset, the column "time_hrs" (the time a recipe requires in hours) holds values for only 917 of the 413,769 recipes (more than 99% missing). The column "calories" is usually formatted like "2000 (500)" (values in kJ and kcal), but in more than 135,000 cases contains the word "Eiweiß" (translates to "protein") and in 28,902 cases just the symbol "-" instead, thus, is suitable for neither numerical nor text analysis with approx. 40% of the values being useless. Additionally, a large amount of recipe titles from kochbar is excessively modified with symbols (e.g. hearts, stars and exclamation marks), written in capital letters only or includes longer explanations, which should actually be written as a subtitle. The quantities of ingredients are not standarised in the kochbar dataset, i.e. there are more than 200 different words or abbreviations that are used for identifying amounts of ingredients, which all needed to be found and filtered manually during analysis. Another issue with this dataset is the attribute for average rating. Its evaluation shows that 25% of the recipes have achieved the maximum rating of 5.0 and 75% have a rating higher than 4.92. This means that the data of this attribute has a strongly skewed distribution with most of the information being lost due to the scale's cap at 5. For this reason, the average rating of recipes in the kochbar dataset is only useful for finding low-quality recipes that have a rating lower than 3 or 2.

The dataset from chefkoch has the poorest data quality of the three, as 11 of 18 columns contain dirty data. For example, the website has an inconsistent format for the time that a recipe requires, which sometimes contains days, hours and minutes, and sometimes only one or two of these and is always enclosed by text that indicates what the number implies. Thus, multiple conditions need to be considered when cleaning this attribute. Similarly, the popularity metrics (how many times a recipe was printed, saved or shared) contain the respective number, but also an asterisk and another value in brackets, which indicates how much the metrics have increased since last month. Even though these additional metrics can be useful, the website provides them in a dirty format instead of splitting the values into separate data fields that could be scraped much more easily and cleanly. The column for the calories of a recipe holds "keineAngabe" (not specified) for more than 202,000 recipes. The user activity column (i.e. how many recipes a user published and average publications per day in a single data field), holds the value zero for 69,819 out of the 319,578 recipes, even though all of these users have published at least one recipe, without which they would not be part of this dataset. Due to the rather "makeshift" format of the chefkoch HTML format, there has been an error during scraping, which did not occur in the error log during the process of collecting data, but only later when the analysis was performed. Due to the many inconsistencies and varying formats on the chefkoch website, most of the attributes for approximately 96,000 out of the 319,578 recipes could not be collected correctly, which has forced the analysis of this dataset to focus on the remaining 220,000 recipes.

The website eatsmarter has provided a quite clean dataset with regards to the text

fields, as the titles did not contain any symbols or unnecessary characters and the ingredients had already been split into clean lists of single words. However, from the relatively low number of attributes (12), only three are providing popularity metrics (average rating, number of rating votes and comment number). The preparation time for each recipe is similarly formatted as in the chefkoch dataset. Additionally, these three attributes have many missing values and the value changes between the months are so small that it is not possible to distinguish real trends from random occurrences. Thus, a deeper analysis of statistics and trends of this dataset has not provided valuable results. Despite these many drawbacks, the eatsmarter dataset can still be useful even without significant popularity metrics, as it provides a large set of relatively clean recipe texts. Additionally, the eatsmarter website (and magazine) focuses on rather "healthy" recipes, which are demanded by an increasing amount of users, as more and more people care about the quality and healthiness of their food.
In conclusion, one can see that the datasets are quite different in terms of data quality and the amount of information they provide. The dataset from kochbar has the most usable data, especially considering popularity metrics (e.g. number of clicks or recipes marked as favorite), while its text fields need extensive cleaning. This makes the dataset useful for trend and numerical analysis, but requires much more work for generating a database of consistently well-written recipes. Chefkoch provides a dataset that has cleaner text data, but almost only dirty numerical attributes, which need to be cleaned before further trends and statistical properties can be examined. The eatsmarter dataset provides clean text fields and a consistent level of quality over all recipes, as they are being published by one editorial team. Unfortunately, this website has very low rates of user activity and only a few popularity metrics for the recipes, which makes the dataset good for building a recipe database, but useless for analysing trends and statistics.

4.1.2 Authors

This section focuses on the analysis of the kochbar dataset, as the used analysis techniques and results for chefkoch are very similar[16] and eatsmarter does not provide authors for their recipes. The analysis of user activity on these websites has been led by three main questions:

a) Which users have contributed the largest number of recipes?

b) Which users are the most successful or efficient?

c) How can the recipes be found that have made these users so successful?

From a business perspective, the goal is to find the users that have published successful recipes consistently, for they could be a valuable source for selecting recipes to feature in this service. Separate datasets for each website have been created for this analysis by aggregating all numerical attributes per user. From the kochbar dataset, the attributes for the number of recipes, clicks (recipe visits), favorites (recipes marked as favorite) and rating votes have been aggregated and values like average rating, clicks per recipe, median clicks and votes per recipe calculated. Using the publication date of the recipes, one can also see quite easily for how long a user

[16]The complete detailed chefkoch analysis can be found in the GitHub repository [7].

has been active in the community. There are many more possibilities to calculate combinations of features, but not all of them will be useful, if they are too strongly correlated and practically describe the same property. It is important to analyse the different popularity metrics, as a high number of clicks may not necessarily mean that the recipe has a high quality. Even though the number of votes or favorites is generally lower than the number of clicks (as a user can only rate a recipe once, but return to the page multiple times), these values could be a better indicator for a recipes quality. This is based on the assumption that a user will most likely try a recipe before returning to the page and rating it or marking it as a favorite.

a) Which users have contributed the largest number of recipes?

Because the aim is to perform exploratory analysis, the newly created dataset should be examined by viewing the data and its distributions and dependencies. The top users on kochbar with regards to the amount of published recipes can be seen in Table 8. The full table can be seen in "Detailed_Analysis_Kochbar.ipynb" on GitHub [7].

Table 8: Top users from kochbar (excerpt)

User	**Recipes**	**Clicks**	**Favorites**	**Votes**	**Days**
Das perfekte Dinner	8,364	17,616,144	57,057	14,139	3,642
schnuffelohr	5,816	2,784,642	12,993	102,082	3,551
Chefkoch1940	2,801	1,696,411	7,568	59,274	3,218
imhbach	2,613	3,499,886	10,624	58,459	3,159
Das perfekte Promi-Dinner	2,319	4,766,553	11,025	2,917	3,579

As kochbar is working in cooperation with the German TV channel VOX, the official user accounts of the two cooking shows "Das perfekte Dinner" and "Das perfekte Promi-Dinner" are highly active and ranking first and fifth in the top users table (with respect to published recipes). The overview shows that more than 10,000 recipes have been published by the TV channel's two accounts. Three of the top five users have published their first recipe approximately ten years ago, while the other two have joined the community more or less a year later. This means that the top five authors have published between 0.6 and 2.3 recipes per day on average, which is a remarkable performance over a period of ten years. The fact that so many private users are able to keep up with the professional authors shows how actively those users are engaged in the community.
One can also see that the popularity metrics (clicks, favorites and number of votes) are not proportional to the number of recipes. Instead, user "imhbach" has generated approximately twice as many clicks as user "Chefkoch1940", even though they have published a similar number of recipes, and the recipes of user "schnuffelohr" have been rated more than 102,000 times. This could indicate that those recipes have been cooked successfully many times, possible due to their simplicity or clarity. The account of "Das perfekte Dinner" has accumulated more than 17.6 million visits and more than 57,000 favorites on their recipes, which is significantly more than the other accounts and probably due to the show's popularity and regular appearance on TV. However, this account has accumulated 14,139 rating votes, which is significantly

less than the user "schnuffelohr" has. This indicates that just having a high number of clicks or recipes does not necessarily mean that a user has published high-quality recipes.

b) Which users are the most successful or efficient?

In order to find the most efficient users, i.e. users that have gained a high amount of popularity with only a few recipes, one has to use other values than just the number of clicks of favorites. For this purpose, new attributes can be generated very easily by combining certain features through numerical division, for example. However, to limit the data's dimensionality, the focus should be on calculating sensible features. The analysis in this project has focused on the following five attribute combinations from the kochbar data: clicks per recipe, median clicks, votes per recipe, clicks per day and favorites per day. While clicks per recipe is a mean value that can easily be distorted by a single recipe that has a lot of clicks, the median shows the amount of clicks that is actually in the middle of all values. If a set of recipes has a strong difference between mean and median clicks, this means that there is probably a single recipe with a very high number of clicks, while the other recipes are not very popular. When searching for users that have published popular and good recipes consistently, controlling the disparity between mean and median can be very useful.
As mentioned above, a recipe's clicks might not be representative for its quality, which is why at least another popularity metric should be compared during analysis. The votes per recipe could be seen as an indicator of how many people have actually tried out a recipe and come back to rate it on the website. The number of clicks and favorites per day can be seen as a time-normalised attribute, which takes into account for how long a user has been active on the website. This can be useful, because users that have been active in the community for a longer time logically have had more time to accumulate clicks and favorites, which makes it more difficult to find relatively new users that have published high-quality recipes.
Using these newly calculated attributes, the user data's sorting can be changed to show the most efficient recipe authors. It makes sense to view users with a high "click efficiency" (high clicks per recipe) and a high "vote efficiency" (high number of rating votes per recipe). An overview that shows the top five users with highest click efficiencies can be found in Table 9. Those top five users show a very high number of clicks per recipe, but their median clicks is much lower in comparison. This means that these users most probably have published a single recipe that has been extremely popular and a few others that were not popular at all. A more extensive version can be found in the detailed analysis in the GitHub repository [7].

Table 9: Users with high click efficiency (kochbar)

User	Recipes*	Clicks	Clicks / Recipe	Median Clicks
mareikus	7	1,558,124	222,589.14	29,746
saaahraaa	9	1,494,471	166,052.33	7,254
Eni0	5	634,460	126,892.0	1,004
zaphi67	36	4,063,146	112,865.17	3,759
Kruemelmonster	9	967,351	107,483.44	12,022

*: The set has been filtered for users with at least 5 recipes

The scatter matrix in Figure 1 (appendix) shows pairwise distributions (scatter plots) for the calculated attributes[17] from kochbar and provides information about each attribute's distribution. The diagonal of the matrix shows the kernel density estimates, which are approximations of each attribute's probability density that can be used to identify the data distribution (similar to histograms). In general, one can see that the values are distributed unevenly and the density estimates seen in the scatter matrix show that all attributes in the kochbar dataset have an exponential distribution. This can be explained by a property that many communities like these have: Users that do not want to spend too much time searching will filter for the recipes with the highest ratings or favorites, which are more likely to be good. If they like the recipe, they will mark it as a favorite or rate it, which will again increase its position in the search results. Thus, recipes that have been able to gain a certain amount of popularity, will reach a broader audience and amplify their further gains in popularity. In addition to this, there is also a very large amount of users that have published only a few recipes in contrast to the very few users that have many thousands of recipes. This imbalance also affects the number of clicks, favorites and votes a user has collected, even if there is not a high correlation between these values. Considering this background, it seems natural that all attributes are distributed exponentially. In such circumstances, it is not beneficial to use different visualisation techniques to show the data distributions, e.g. box plots[18], which have been mentioned in section 2.1.

A closer inspection of the scatter matrix in Figure 1 shows that there seems to be a high correlation between the pairs clicks per recipe / median clicks, and clicks / clicks per day (both of which are pairs of similar attributes), but also favorites / clicks, and favorites / number votes. In general, it seems that the number of favorites is highly correlated with the other popularity metrics. To analyse the dependencies in more detail, a correlation coefficient has to be used that measures dependencies between two attributes. A very commonly used correlation coefficient is Pearson's r, even though it does not support missing values and is limited to showing linear dependencies in data. Expanding the analysis of correlations from only linear dependencies to all dependencies that can be represented with a monotonic function can be very interesting for this analysis, as most distributions in this dataset are non-

[17] recipe count, clicks, clicks per recipe, median clicks, average rating, number of votes, number of times marked as favorite, days since the first recipe, votes per recipe, clicks per day and favorites per day

[18] Even though box plots are used commonly during data exploration, they do not provide a good readability and interpretability on exponential distributions, because all important parts (mean, quartiles and whiskers) are then drawn to the very same spot, the data's maximum.

linear and exponential. Spearman's rho is a rank correlation coefficient which does exactly that. The pairwise correlations are shown in Table 10, with the respective p-Values in Table 15 (Appendix).

Table 10: Spearman's rho for user data (kochbar)

	R	C	$\bar{C}$	$\bar{\star}$	V	F	D	V/R	C/D	F/D
R	1.00	0.82	0.26	-0.43	0.88	0.82	0.11	0.42	0.80	0.80
C	0.82	1.00	0.73	-0.41	0.83	0.90	0.10	0.54	0.98	0.88
$\bar{C}$	0.26	0.73	1.00	-0.19	0.38	0.55	0.05	0.43	0.71	0.53
$\bar{\star}$	-0.43	-0.41	-0.19	1.00	-0.45	-0.44	-0.14	-0.28	-0.39	-0.42
V	0.88	0.83	0.38	-0.45	1.00	0.87	-0.08	0.77	0.82	0.87
F	0.82	0.90	0.55	-0.44	0.87	1.00	0.11	0.62	0.88	0.98
D	0.11	0.10	0.05	-0.14	-0.08	0.11	1.00	-0.29	-0.04	-0.02
V/R	0.42	0.54	0.43	-0.28	0.77	0.62	-0.29	1.00	0.56	0.64
C/D	0.80	0.98	0.71	-0.39	0.82	0.88	-0.04	0.56	1.00	0.89
F/D	0.80	0.88	0.53	-0.42	0.87	0.98	-0.02	0.64	0.89	1.00

(C = clicks, D = days, F = favorites, R = recipes, V = votes, $\star$ = rating, $\bar{}$ = mean)

The correlation matrix shows multiple high coefficients, which can be used to understand the dependencies in the user dataset. It is quite clear from these values that when publishing published more recipes, a user will aggregate more clicks, votes and favorites in his/her account. In general, the numbers of clicks, favorites and votes are quite similar, which indicates that they usually accumulate proportionally, even if not linearly. Thus, one cannot distinguish well between recipes with many clicks and recipes with many votes. The attributes for clicks per day (C/D) and favorites per day (F/D) have almost the same coefficients, similarly to their counterparts without time normalisation. The attribute for days (D) does not have significant correlations with any of the other attributes, which shows that the actual amount of days a user has been active in the community is relatively independent from the user's aggregated popularity metrics. This means that having been published a longer time does not necessarily make recipe more successful or popular than others. The coefficients for the average rating show why the dirty data of that attribute is hardly usable: If a recipe receives more rating votes, it is more likely that the rating will be a more precise representation of the community's opinion of it. This usually moves a rating away from an integer value towards a rating with multiple decimal values. However, if a new recipe is published, the author and his/her friends might rate the recipe with five stars, while few other people have seen that recipe. This means that new or unnoticed recipes can have a five star rating while established, frequently rated and popular recipes might rather have a rating of 4.94 out of 5 stars. Because kochbar is not treating such newcomer recipes in any way and users tend to sort recipes by their rating, a new recipe with only a single 5 star rating will appear as better than a popular recipe with 2,000 ratings that average at 4.94 stars. This is reflected in the inconsistent correlation coefficients for average rating and proves that when searching for popular recipes on kochbar, one cannot rely on the recipe rating, but instead should sort by clicks or number of favorite votes.
Finally, some of the exponentially distributed attributes can be transformed by applying the logarithm in order to move their distributions more towards a normal one. The histograms in Figures 2 and 3 (in the appendix) show that this does not work

for all attributes, but provides rather normal distribution for some of them. The main motivation to perform such a transformation is very simple: By normalising a distribution, one improves its compatibility with various analysis techniques. This is due to the fact that a high number of statistical (e.g. dimensionality reduction with principal component analysis (PCA)) and data mining techniques (e.g. linear regression) require normally distributed data to provide correct results. Apart from this technical perspective, it can also be useful to have normal distributions when searching for users or recipes that are far above average, as it is very simple to select the top 5% of successful recipes from such a distribution.

c) How can the recipes be found that have made these users so successful?

By using the insights gained above, a simple way to find the users with a high amount of clicks as well as a high click efficiency is to use the intersection of both datasets that have been created (top users and efficient users). It might be useful to select users that, for example, do not have more than 100 recipes, even if they have provided all of them on a consistently high quality level. By filtering the users by their recipe count, a manual quality control can still be performed and different recipe styles and food types can be offered, which is important in a business for providing a variety of high-quality options that can satisfy different tastes.
The top five users from the intersection of users with many recipes and those with high efficiency (from kochbar) can be seen in Table 11. It is easy to see that these users have a much higher amount of clicks per recipe and favorite votes (per recipe) than the top users, which means that their recipes have been much more successful on average.

Table 11: Most efficient top users from kochbar (excerpt)

User	Recipes*	Clicks	Clicks per Recipe	Favorites
Rejerfan_55	90	1,611,575	17,906.39	3,552
sweetyhoneypie	90	1,391,000	15,455.56	3,432
zimtzicke	86	1,302,073	15,140.38	3,215
princess-tanja	100	1,305,825	13,058.25	1,813
Backbiene	83	678,908	8,179.62	1,353

*: filtered for users with a maximum of 100 recipes

A closer look at user "Rejerfan_55" shows an exceptionally homogeneous collection of recipes with regards to the popularity metrics, as well as an intriguing variety of cake recipes. The recipes all have similar structures and levels of data quality, which is also important when adopting them for a food box service.
This much more complex approach to finding consistently well-written recipes helps to find not only the authors of very popular recipes, but also those that have been providing many high-quality recipes. This helps to avoid picking users that have published only one or two highly popular dishes, but also 10 or 20 bad ones. Of course, this does not mean that one should avoid recipes that have the highest number of clicks. In contrary, the best solution for a business will most probably be to combine the recipes that have aggregated many clicks with consistently well-written and popular recipes from efficient users to provide an attractive selection of options.

4.1.3 Recipes

It is easy to sort all recipes by one attribute to find the most popular ones, e.g. by clicks, favorites or comment number. However, it is necessary for deeper analysis to create a new data subset that contains all the numerical attributes, which can be used for statistical analysis. These attributes include: average rating, clicks, number of comments, number of days since publication, number of favorite votes, preparation time, and number of rating votes. As explained above, these attributes are distributed exponentially, except for the number of days, which has a rather skewed Poisson distribution. For these distributions, a box plot visualisation is not suitable[19], but a scatter matrix (Figure 4, appendix) can again provide some insights. The scatter plots indicate linear dependencies for the attribute pair comment number and number of votes. However, for finding non-linear dependencies, calculating Spearman's rho is, again, the most precise way. Table 12 depicts the pairwise correlation coefficients for the numerical values of all recipes (respective p-Values can be found in Table 16, appendix).

Table 12: Spearman's rho for recipe attributes (kochbar)

	$\bar{\star}$	c	co	d	f	t	v
$\bar{\star}$	1.00	-0.12	-0.23	-0.09	-0.21	-0.00	-0.24
c	-0.12	1.00	0.29	-0.07	0.56	0.05	0.30
co	-0.23	0.29	1.00	-0.07	0.60	0.11	0.85
d	-0.09	-0.07	-0.07	1.00	-0.02	0.03	-0.29
f	-0.21	0.56	0.60	-0.02	1.00	0.09	0.60
t	-0.00	0.05	0.11	0.03	0.09	1.00	0.07
v	-0.24	0.30	0.85	-0.29	0.60	0.07	1.00

(c = clicks, co = comments, d = days, f = favorites, t = preparation time, v = votes, $\bar{\star}$ = avg. rating)

This correlation matrix has only few high coefficients. The linear dependency that could be seen in the scatter matrix (between comment number and number of votes) can be confirmed, but otherwise only the attribute favorites has coefficients above 0.5. This means that when viewing single recipes, it is more likely for a recipe to be popular if it has been marked as a favorite from more users. This is quite interesting, as it shows that a high number of clicks does not necessarily mean that a recipe will generate more activity in the community in terms of rating votes and comments. Similarly to the user analysis, a logarithmic transformation could be performed here as well. This would not change the results of Spearman's rho, as it is not affected by monotonous transformations.

[19](which can be seen in the detailed analysis on the GitHub repository [7])

Text analysis can be performed by using the text fields of the recipes, which are the title, list of ingredients and preparation steps. As explained in section 3.5, there are different techniques to generate new features from text, depending on the text lengths and their vocabulary size. By using those, the following questions can be answered:

a) Which are the most frequent words in titles?

b) Can a detailed search function be built using the ingredients?

c) How can similar words be organised?

From a business perspective, generating useful attributes from text fields can be very valuable, as a multitude of new product features can be built using them. Recipes convey their main information through text, which is why this part of the analysis is essential for utilising the full potential of the data.

a) Which are the most frequent words in titles?

Almost all recipe titles consist of more than one word, which means that they need to be split up before the most common words can be counted. Additionally, some titles contain useless symbols or stop words, which have been used by some authors to distinguish their recipes from others or add long and redundant explanations. There are not only German stop words, but also French or Italian ones (e.g. "à la" or "di"), which are supposed to indicate a certain culinary style. However, as many authors possibly do not have a proficiency in those foreign languages, they have created various versions of the stop words by changing accents or misspelling. Even when using the German term for *grandmother's* ("Oma's"), some authors have used a variety of accent symbols for the s-genitive's apostrophe. These things seem to be quite minor, as they only affect a single or few characters or symbols, but they can completely change the evaluation of these texts, because variable spellings lead to different, seperately counted words.
After cleaning all datasets, an overview of the most popular words in recipes titles can be presented for all three websites, which is shown in Table 13 (the original words have been translated into English for simplicity). It can be seen that similar words are popular across all websites, for example salad, tomato, vegetable, stuffed and pasta/spaghetti. The word counts also show how these word counts are distributed, which are foreshadowed in how fast the word count decreases when descending the table towards lower-ranked words. The titles from chefkoch seem to be the most balanced regarding their word counts. They also include the largest ratio of sweet dishes (e.g. tart, cake and muffins). Even after cleaning, the number of distinct words in the dataset's titles are significantly disparate: While eatsmarter has 35,120 unique words in recipe titles, chefkoch has 82,065 and kochbar has 175,034. This suggests that kochbar probably still contains dirty data, because that number includes different versions of the same words due to singular, plural, different cases and typos, which all cause a slightly different morphology. There are techniques like stemming or lemmatisation to reduce different versions of a word to a single (word) stem or lemma, but they require some efforts to be implemented correctly and still would not be able to tolerate typos in words. A more flexible method is to use word embeddings, which will be described below (along with question **c**).
To reduce the dimensionality of the recipe titles, one can perform a very simple

technique: The different title lengths can be reduced to a single word by selecting the word with the highest word count in the respective vocabulary. For example, a "Vegetable Salad with Cheese" on kochbar will then be reduced to the word "Salad". These one-word titles can then be converted to a word vector (with the word embeddings) and used as a feature when applying data mining techniques. An interesting visualisation of most frequent words can be created by using word cloud figures, which only consist of words in different sizes and their size is in relation with their frequency in the vocabulary. Exemplary word clouds have been created for eatsmarter's titles (Figure 7, appendix) and the most popular ingredients on eatsmarter (Figure 8, appendix) and on chefkoch (9, appendix), which show that the community on chefkoch clearly prefers pastries over savoury dishes.

Table 13: Most frequently used words in recipe titles (translated)

chefkoch		**eatsmarter**		**kochbar**	
word	**count**	**word**	**count**	**word**	**count**
Tomato	3,223	Vegetable	4,732	Salad	6,018
Salad	3,146	Salad	4,499	Vegetable	4,370
Tart	2,695	Tomato	3,534	Stuffed	4,030
Vegetable	2,677	Stuffed	2,030	Cake	3,752
Cake	2,507	Cheese	1,814	Tomato	3,418
Sauce	2,480	Chicken	1,797	Spaghetti	3,218
Spaghetti	2,371	Asparagus	1,690	Asparagus	2,870
Stuffed	2,210	Apple	1,670	Pasta	2,785
Muffins	2,151	Pasta	1,638	Cheese	2,522
Pepper	2,139	Sauce	1,542	Salmon	2,464

b) Can the ingredients be used to create detailed search function?

In a similar way as above, the ingredients can be split up into single words and counted as well, which, again, results in very different numbers of unique words for each website: On average, the recipes' ingredients lists have an average length of 15.4 (chefkoch), 10.8 (eatsmarter), 74.8 (kochbar) before cleaning stop words and 6.4 (chefkoch), 10.7 (eatsmarter), 13.4 (kochbar) after cleaning. The main reason for kochbar having such a high length of ingredients lists before cleaning is that kochbar's ingredients lists also contain many different words and explanations for quantities, which are obviously not standardised on that website (in contrast to the others). The ingredient lists from chefkoch are halved after cleaning (on average), because half of the table entries are quantity indicators (e.g. "100g", "2"). However, they can be cleaned much more easily as they are standardised and fit into a single list entry per ingredient. While eatsmarter has 7,932 unique ingredients in the recipes, kochbar has 107,623 and chefkoch 148,507. These completely diverse statistics again demonstrate how dirty data can affect the results of an analysis and show the benefits of having a clean dataset like eatsmarter's.
From a business perspective, it would be very interesting to be able to search for specific ingredients - a function which none of the recipe websites provide. Using the ingredient search, one could filter all recipes with the ingredients that one has available at home, which can help the users to utilise existing ingredients, be more

sustainable and save money. To enable such a search function, the datasets can be converted by using one-hot encoding, which has been explained in section 3.5. Before performing such an encoding, however, one should select a subset from the vocabulary, because using all possible words for encoding would generate a high-dimensional dataset, which could need multiple TB of space and probably would not fit into any computer's memory. For instance, using only the 1,000 most frequent ingredients for encoding results in a csv table that requires 1.8 GBs of disk space. But, once created, the sparse table (i.e. contains many zeroes) can be used for quickly finding recipes that share a set of ingredients. For example, if a recipe with tomatoes, onions and flour is needed, one can just filter the table for all recipes that have a one in the three respective columns. Additionally, the resulting recipes can be sorted by their popularity using the techniques shown above.

c) How can similar words be organised?

As already mentioned, word embeddings provide a very useful and efficient way of projecting a complete vocabulary into a high-dimensional space. When using FastText [4] for model training, the default number of dimensions is 100. This means that all words are being placed in a 100-dimensional space and have a vector that hints at their position in this feature space. In comparison to one-hot encoding, a vocabulary of more than 100,000 words can be completely represented with just 100 dimensions instead of 100,000, which is much more efficient regarding memory and disk space usage.
First, a text corpus needs to be prepared for training a vector model. The corpus should contain as many documents as possible and complete sentences including stop words. This is important because the word embeddings for each word are being calculated with respect to the words that are occurring in its context. In the context of recipes, words of similar ingredients or cooking techniques are more likely to be used in the same way regarding their position in the sentence and their accompanying words, which means that words that are synonyms or describe similar things like *ham* and *bacon* or *fish* and *salmon* will be located very closely in the word model's feature space. A separate Python script is available on this project's code repository (`FastText_Model_Kochbar.ipynb` [7]), presenting the techniques that have been used to generate the corpus from all kochbar recipes, train different word embedding models using FastText and evaluate the model's accuracy on manually selected examples. The recipe texts from chefkoch or eatsmarter could have been used as well, but the kochbar dataset contains the largest text corpus by far and the generated model already has a satisfying performance.
From a business perspective, a word embedding model might not seem very useful at first. However, this model can be used to enrich the recipe dataset by adding the feature vector for each recipe's top title word. Thus, the dataset will not only contain the recipe titles, ingredients, preparation steps and popularity metrics, but also a 100-dimensional vector that places the recipe in a feature space, close to similar recipes (using the most frequent word from the recipe title). This newly created feature can then be used during data mining for clustering similar recipes.

This section has shown that explorative techniques can be used to understand users and recipes in more depth and find interesting dependencies. This knowledge can be used to select each platform's best recipes more precisely, as the selection does not rely on a single metric (like the average rating), by finding users that have published good recipes consistently. Using the scatter matrix, dependencies and

outliers can be found quickly and correlation coefficients help to verify these initial assumptions.
The detailed analysis of the kochbar dataset has revealed that recipes with a high number of clicks do not imply a good recipe quality, but rather a dish that is generally popular and often searched for. However, the recipes that have encouraged users to try them, return to the website and leave a comment or rating, are much more likely to be interesting and of high quality. The techniques presented above demonstrate how such consistently high-quality recipes can be found and extracted from the datasets. Additionally, new attributes have been created that can enable new product features or more complex data mining techniques.

4.2 Trend Analysis

This part of the analysis will focus on how the data has changed and evolved over time. For this purpose, two different time axes are available for use: The publication date of each recipe and the monthly dataset snapshots taken over the last 6 months. These two time dimensions might not provide a standard data format for a classical time series analysis, as that would require a large number of uniform timesteps. An example for a standard time series is stock market data, that shows a price for each listing per second, which can be aggregated for a day, week, month, etc. In the case of stock market listings, the price is a value that results from multiple factors and can remain unchanged, increase, or decrease. The same applies to data from a temperature sensor, for example. However, the features of the recipe datasets do not behave like that. Instead, they are either immutable after the recipe's publication (e.g. title and ingredients) or they increase over time by accumulating (e.g. clicks, votes, favorites). The trend analysis will present these two different approaches to viewing the data from a temporal angle and will aim to answer the following two questions:

a) How frequently are users publishing new recipes on the websites?

b) Which techniques can be used to detect the trending recipes of each month?

As the eatsmarter dataset does not provide publication dates for the recipes, it cannot be utilised to answer the first question. Additionally, further analysis has shown that the popularity metrics employed for identifying trends on each website are too rarely used on eatsmarter to be able to distinguish actual trends from random occurrences. This means that eatsmarter's user community is not very active and cannot be used to identify trending recipes. However, the two other datasets (from chefkoch and kochbar) that can provide a sufficient amount of insights.

a) How frequently are users publishing new recipes on the websites?

The easiest way to understand the amount of user activity is to compare how many recipes have been published each month and year. This can be performed quite easily be generating histograms with the correct amount of bins, which can also be seen as a time series representation. Visualisations for both datasets, chefkoch and kochbar, can be seen in the Figures 5 and 6 in the appendix. Both histograms show that the recipe communities have had a peak in activity in the beginning of 2009 (17,500 new recipes per month for kochbar and 2,500 new recipes per month for

chefkoch) and a steady decrease since then. There are two main effects that can be seen in the histograms: it seems that the websites have been losing their user commmunity since the peaks in 2009 (global trend) and that there exists a higher author activity in winter during each year, as the number of published recipes is usually lower during spring and summer months (seasonal trend).
At this moment, the number of new recipes seem to be limited to 200-500 new recipes per month for both websites according to the time series plots. It is quite curious that this number does not concur with the values from Table 7, which shows that the total number of recipes has increased by 82,830 for kochbar and by 5,803 for chefkoch. These numbers do not have to be the same of course, as it could be assumed that there are external effects influencing the values in the table, e.g. multiple thousand recipes being deleted from kochbar very recently. However, the increase in recipe count on both websites should be the same as the number of newly added recipes in the same time frame, because technically it should not be possible to post new recipes with an older date. Unfortunately, this seems to have happened, as both websites have grown much more in total than the number of new recipes in these months suggest. A separate analysis (which can be found in the notebook file `Kochbar_Dataset_Growth.ipynb` [7]) has confirmed this vast discrepancy, which has been detected while creating the dataset for trend analysis: while the total recipe count of the kochbar database has grown by 82,000 since October, the following months only contain 300 to 400 recipes with the respective publication date. The reason for this phenomenon is not completely clear, but the analysis shows that the snapshots taken after December contain more recipes in older months, e.g. approx. 4,000 more with a publication date in February 2009 only. This suggests that the administrators of the kochbar website are not managing their database in a clean way, which leads to inconsistent and redundant data filling the website with many more recipes than there actually are.
Concluding the insights from above, it seems that publication dates of recipes do not provide much information. The insights from this part of the trend analysis can be summarised in the following points:

1. significant changes in user activity have strongly skewed the distribution of recipe publications and have shown a particularly low amount in the last months,

2. almost all users publish their recipes in their first few days as a member and do not provide much afterwards, and

3. a large portion of recipes on kochbar is published by the sponsored TV channel accounts, not the user community, which means that those recipes cannot be used to understand the user community.

These effects make it difficult to obtain valuable knowledge from a trend analysis using the recipe publication dates.

b) Which techniques can be used to detect the trending recipes of each month?

In order to detect monthly changes in the data, new datasets with the recipe metrics for each month and the differences (deltas) of all metrics for each month have been generated using a separate Python script, which is available on the GitHub

repository as well (`Trend_Dataset_*.ipynb` [7]). The newly created trend datasets can be used to find recipes that have accumulated the largest amount in each metric for each month. This means that finding the recipes with the most clicks in December or with the most favorites in January is quite simple, as the dataset containing the deltas has to be sorted by the respective column. As previous analysis has shown that the popularity of a recipe in the kochbar dataset can mainly be described by clicks and favorites, the best recipes each month regarding both attributes have been presented in two tables. An excerpt of these two tables can be seen in Table 14 (titles are translated for simplicity). The data from chefkoch also provides a similar overview, as there are popularity metrics that can be used for detecting trends as well. However, the metrics provided by chefkoch are much more limited, as they do not include an indicator for traffic (e.g. number of clicks), but only how many times a recipe has been printed, saved or shared. These metrics cannot be seen as direct indicators of popularity at a certain time, as users can only save a recipe once and most probably do not print it every single time they cook it. Additionally, the number of shares is relatively low, which suggest that this function is not used very often or only by a small part of the user community.

Table 14: Trending recipes per month (kochbar)

	Top 5 Recipes by	
Month	**Clicks**	**Favorites**
October	Pancakes	Cheese Soup
	Chestnuts	Cheesecake
	Pumpkin Jam	Tomato Soup
	Roasted Deer	Kaiserschmarn
	Chili con Carne	Spaghetti Bolognese
November	Pancakes	Chocolate Cake
	Chestnuts	Cut-out Cookie
	Chili con Carne	Coconut Macaroons
	Goose Leg	Pasta Casserole
	Rhubarb Cake	Pan-fried Turkey Strips
December	Goose Leg	Pan-Fried Chicken Strips
	Pancakes	Fried Potatoes with Cheese
	Duck	Lasagne
	Coconut Macaroons	Stuffed Bell Pepper
	Goose Breast	Goose Leg
January	Pancakes	Quark Balls
	Chili con Carne	Potatoes au Gratin
	Cabbage Soup Diet	Meatballs
	Quark Buns	Puff Pastry Shells
	Meat Casserole	Meat-Cheese Soup
February	Pancakes	Hungarian Goulash
	Quark Buns	Baked Banana
	Plum Cake	Cheese Soup
	Meat Casserole	Muffins
	Pan-fried Schnitzel	Baileys Muffins

Table 14 (and the original table found online in the detailed analysis for kochbar [7]) indicates that the recipe for pancakes is usually the most popular recipe, followed by other all-time classic recipes, which are frequently visited throughout the whole year. However, it is obvious that more traditional Christmas dishes have seen high traffic in November and December (especially ones with goose, duck and deer), already starting in November. More specifically, the hearty dishes that are inspired by a "grandmother" style seem to be particularly popular. It is also quite interesting that one of the most frequented recipes in January is one for a cabbage soup diet that is supposed to make you lose 7 kgs of weight in one week (accompanied by similar recipes), which many people seem to be interested in after gaining weight during the Christmas holiday season.
This analysis of course only provides a short glimpse and can be expanded to a higher number of recipes per month and over a longer time period. Such an approach can reveal further interesting trends: the increasing popularity of pumpkin soup and chestnuts during autumn, as that is their season, and higher demand in cakes in January and Februar, possibly because of an increased amount of birthdays during that time. However, the analysis can help to find outliers as well, e.g. the rhubarb cake in November, even though rhubarb season is usually during late spring. This seems to have been caused by an external effect, which could be an appearance on a TV show or the recipe being featured on the website (kochbar regularly recommends "top recipes").
The users on chefkoch seem to favor cakes and tarts much more, as many corresponding recipes are trending in most of the months. Since chefkoch's popularity metrics reveal whether a user has printed or saved a recipe, this could imply the recipes that users are thinking to prepare in the coming days or weeks, which could foreshadow specific occasions. In principle, this could be even more interesting for a business than finding trends after they have appeared (by looking at the clicks), as it can be used to predict demand in ingredients and adapt the supply accordingly. However, these metrics have a few drawbacks, as users are not likely to trigger them every time they are actually planning to cook a recipe. Due to this, demand predictions solely based on these popularity metrics are not likely to be precise.
In conclusion, this trend analysis can contribute to finding trending recipes for each month and it also reveals the all-time classics like pancakes and Schnitzel, which only seem to be temporarily moved aside by seasonal competition. When analysing these trends, it can be useful to separate between seasonal trends (e.g. Christmas and autumn) and irregular trends in the community (e.g. the rhubarb cake or possible "birthday season" at the end of winter). From a business perspective, the seasonal trends may not be highly valuable, as most of them are quite obvious. Additionally, all seasonal trends are usually shared between different (German) recipe websites, because their users most probably live in the German-speaking countries. In contrast, the irregular trends can be very useful to find actual trends in the community, but can also be caused by random outliers in data due to external effects. Thus, when checking an irregular trend for validity, it might be sensible to compare the popularity of such recipes between different websites.
This analysis has illustrated that most of the food trends are based on seasonal effects, as the availability, taste and price of seasonal ingredients are naturally the best during their season, which motivates many people to include them in their diet during that time.

4.3 Data Mining

For arranging the recipes into larger classes, category labels need to be created and used to identify similar groups of recipes. This can also be utilised in the trend analysis shown above, for example to see if hearty dishes in general are more popular during Christmas season, or if salads are trending in spring, when people want to lose weight. An effective technique of ordering recipes in categories is clustering, which has already been mentioned in section 2.3 and will be presented in this section. Clustering is a data mining technique that is considered as an *unsupervised learning* method. This means that the model does not require previously labelled data for training, but can instead generate groups of similar data points on its own. There are different clustering algorithms to achieve this, which have to be selected depending on the underlying problem. A very popular algorithm is k-Means, which finds a preset number of central points (the k has to be set by the supervising user), where data points are concentrated, and create globular regions around them. However, k-Means requires that data is distributed normally and, thus, formed in a globular shape. If the clusters have different sizes, shapes and densities, k-Means is not able to provide useful results.

After exploring the data and creating new features in the analysis steps above, a density-based clustering algorithm seems to be a good option. Density-based clustering entails that the algorithm searches for groups that have a consistent density, instead of choosing central points and drawing a (hyper-) sphere around them. In the context of the recipe data, this means that the 100-dimensional feature vector for each recipe, which places the recipe inside that feature space, can be used to find groups of recipes that are positioned closely together. During this process, the clusters do not have to be of equal size or shape - their recipes just need to be placed in similar distances to each other. The most popular density-based algorithm is DBSCAN, published in 1996 and specialised for finding clusters in large and high-dimensional databases [12].

Before using the clustering algorithm, however, it can be useful to project the recipes onto a two-dimensional space to visualise and evaluate the results more easily. For this purpose, the technique t-SNE (t-Distributed Stochastic Neighbor Embedding) [33] is one of the best and most efficient available today. When applying t-SNE to a high-dimensional dataset, the algorithm performs a dimensionality reduction and projects the points onto a two- or three-dimensional space, while incorporating the relative position of these points in their original 100-dimensional feature space. A look at Figure 10 (appendix) should clarify this. The t-SNE representation has been trained with a perplexity measure of 10, PCA-initialisation, and 5,000 iterations for the eatsmarter dataset. The parameters have been found by testing and differ for each dataset in order to provide a satisfying result that is still readable. The plot clearly shows how similar dishes are placed closely together, for example different salad types, vegetables, fish, pasta and rice, or desserts.

The t-SNE projection provides a very good starting point for performing the clustering with DBSCAN, because similar dishes are already positioned more closely to each other and form groups of higher density. The DBSCAN model has yielded the best results when trained with a minimum cluster size of three data points and an epsilon of 15 on the t-SNE projection of the eatsmarter dataset. Again, each dataset has required its own set of parameters to generate satisfying results. Reaching at least ten clusters with sensible groupings has been the main goal during optimisa-

tion. Further information can be found in the detailed analysis notebook files [7]. The DBSCAN clustering results for the t-SNE projection of the recipes are depicted in Figure 11 (appendix). A comparison with Figure 10 demonstrates that the clustering has produced satisfying results, creating 16 sensible clusters. The number of clusters might be a little too high, as fruits have been split up into 4-5 different clusters. However, these groups of fruits are located separately in the t-SNE projection. It is much easier to combine a few different clusters into one subsequently, than manually splitting up one large cluster that contains different recipe types, especially if the dataset consists of 100,000s of recipes.

4.4 New Product Features

Three ideas for new product features, which can be implemented using the techniques developed in this project, will be introduced in the following:

a) Recipe search by ingredients

b) New recipe categoriser

c) Recommender for similar and accompanying recipes

While features a and c are aiming to provide added value for the user, feature b is useful for internal recipe management, as the category of newly imported recipes can then be identified automatically. Thus, they can be useful to create business value in different ways that have been defined in section 2.5. These ideas have already been hinted at during the analysis above, but the following explanations are intended to provide a more detailed view on the possibilities.

a) Recipe search by ingredients

Searching for recipes by ingredients can be very convenient for users that already have a few ingredients at home, but need ideas on how to use them. This feature can be especially successful with users that want to be sustainable and avoid throwing away perishable groceries that they have not consumed yet. The recipe data with one-hot encoding provides a perfectly suitable database for such a feature. By using filters on the respective columns, recipes that contain specific ingredients can be viewed very easily and quickly. Additionally, the more advanced techniques from the exploratory analysis can be used to not only present recipes with the highest popularity, but also those written by users that are known to have consistently provided very good recipes. Even though these might not be the recipes the majority of users has been searching for, they may be of better quality and could even be more special and exciting for the user. The insights from the trend analysis can be incorporated in this feature as well by using the identified seasonal and irregular trends to sort recipes by their probability to be trending at the moment.

b) New recipe categoriser

When managing a large recipe database for a business like the one proposed in this project, it can be very helpful to automate certain tasks that would require manual work and an employee's time, especially if those tasks are tiresome and unpopular. One of these tasks could be the the categorisation of newly found recipes, so they can be presented to the users in a structured way. A high amount of work would entail from having to read and classify each new recipes manually. Instead, the clustering techniques developed in this project can be used to automate this job and leave the employee the much easier task of assigning each cluster a label like *cakes* or *salads*. These labelled categories can then be used for presenting all recipes in a structured way, which also improves the user experience.

c) Recommender for similar and accompanying recipes

It has been described in the motivation in section 1.1 that recommendations can help users find better products and enjoy a better customer experience, as they need less time to find the required product. The findings from explorative analysis, trend analysis and clustering can be combined to generate recipe recommendations. This can happen in at least two different ways: by recommending recipes similar to what the user has viewed recently, or by recommending different recipes that can be combined with the viewed recipe to create a three course meal (or similar).
The former can already be performed by using trend insights and recipes from the consistent high-quality users to select recipes that belong to the same cluster or category. The latter also is not very difficult to implement, as it only requires a manual labeling of clusters / recipe categories that can be combined well. For example, if the user has selected a recipe from one of the clusters 1, 3, 5 that contain hearty dishes, it might be known that the clusters 12 and 15 contain light desserts that go well with such dishes. Additionally, known trends can be used to identify desserts or appetisers that are particularly popular during that specific time.

5 Discussion

The analysis has presented different techniques for knowledge discovery performed on this project's datasets. This section will evaluate the results and discuss whether the project's goals have been achieved. Additionally, the limitations of this project and ideas for future work will be presented.

5.1 Achieved Goals

The goals for this project have been set in section 3.1. Following the CRISP-DM process model, they have been split into business goals (Table 3) and their respective data mining goals (Table 4). It can now be reviewed, which of these goals have been achieved and if any task was unsuccessful. When performing this review, it makes sense to start at the lower level, as achieving the data mining goals is important to accomplish the respective business goals.
The goals D1.1 (implementing a scraper and extracting different datasets) and D1.2 (cleaning and preparing the dataset) have been achieved and six datasets have been scraped from each recipe website. Considering that some of these websites contain large amounts of dirty data, cleaning and preparing the datasets have taken a considerable amount of time, but eventually produced sufficiently clean and valuable datasets. While the kochbar dataset provides the best popularity metrics for analysing traffic and trends, the dataset from eatsmarter yields the best data quality for text fields, which can be used for visualisations or generating new texts, for example. This means that the business goal B1 (creating a valuable recipe dataset) has been achieved as well.
The goals D2.1, D2.2 and D2.3 (implementing and performing exploratory/trend/DM analysis) have been achieved with the implementations in Python, as presented in section 4. The results from these three approaches have helped to understand user behaviour and preferences over time (business goal B2), as well as revealed general statistics and properties of the different recipe websites. Even though a precise multi-page analysis has not been possible due to the quality differences between each domain and the lack of comparable attributes, the insights from the trend analysis can be generalised to understand global trends that are affecting the user community's preferences in general as well as over the year (seasonal trends). This process has revealed that seasonal trends are mainly dictating the trending recipes, while a few all-time classic recipes are highly popular over the whole year. These results may not be satisfying for those wanting to find trends that are completely different from seasonal patterns, which are often logical and obvious. Nonetheless, the business goal B3 (gaining cross-platform market insights) has been accomplished at least partially.
The evaluation of the analysis has not only led to the results mentioned above, but has also helped to find ideas for three new product features that can be useful for this project's business case. These features are closely related to the techniques used during analysis and can improve the user experience as well as the business' efficiency. Thus, the data mining goals D4.1 (evaluating analysis results) and D4.2 (defining how techniques can lead to new features) as well as the closely related business goal B4 (developing at least one new feature) have been achieved as well.
The goals D5.1 (assessing further techniques for future potential) and B5 (proposing future project ideas) will be achieved with the summary in section 5.4.

5.2 Created Business Value

Even though practically all goals for the data mining project have been achieved, the project's success in actually creating business value should be reviewed separately in this section. Following the definition of business value presented in section 2.5, there are the multiple ways of creating value with this project: creating a valuable dataset that could be sold or harnessed, building new product features that generate additional revenues, implementing solutions for increasing the business' internal efficiency, or increasing user experience by utilising the analysis results.
The implementation has shown that a very large recipe database with more than 900,000 cleaned recipes has been generated and used for generating new (data) features. Even though the market value of such a dataset is a matter of supply and demand, and thus difficult to assess, but the analysis has shown that these datasets can undoubtedly be very useful for a business. The evaluation has yielded multiple potential product features which can increase revenues by motivating users to order more or larger food boxes, or improve user experience by providing an intuitive product search and the possibility to save money by incorporating already owned ingredients into recipe recommendations. Additionally, an automatised way of categorising newly acquired recipes can help the business proposed in this project to manage recipes much more efficiently, as all datasets lack category labels for their recipes and are only sorting them manually on their websites.
In conclusion, this project has been able to successfully create business value with regards to every aspect of the definition proposed above.

5.3 Limitations

Unfortunately, a few drawbacks and limitations have been encountered during the project. The different dataset properties and qualities as well as the lack of comparable metrics have been the most difficult barrier. Not a single recipe popularity metric is shared among the different recipe websites, which has made a cross-platform trend analysis unfeasible. While the dataset from eatsmarter has been the worst with regards to the lack of numerical attributes and user activity, it has been valuable as a clean data source for texts, which has been used for clustering recipes, for example. The dataset from kochbar, in contrast, has shown dirty text fields that have required extensive cleaning, but in turn contained diverse and useful popularity metrics with significant user activity, which helped in identifying trending recipes. This shows that even though data quality and inconsistencies have been a problem, the datasets have provided different advantages, which has enabled a versatile analysis.
Another issue, connected to the first one, has been caused by the need to use scrapers for extracting datasets from each domain. Approximately a third of chefkoch's recipes have been scraped with an unnoticed error due to the website's low-quality data structure, which has been discovered too late for adapting the script. In order to maintain comparability, the scrapers have not been changed over the data collection period of six months. Unfortunately, some of the scraped websites seem to have been changed during that time, leading to inconsistencies in the scraped data. While eatsmarter seems to have lost 25% of their recipes or limited access for scrapers to those pages, kochbar appears to have populated their database with slightly different copies of already existing recipes. Adapting the scraper to such changes would impede comparability over time, but could produce more precise representations of each website. Either way, using a scraper is not an optimal solution, but probably

the best option available for websites that do not provide APIs for clean data access. One fundamental challenge is the goal to detect trends in these datasets. As described during the analysis, neither of the websites provides an attribute that indicates true changes in traffic and recipe popularity over time. Instead, multiple snapshots have to be taken to calculate the differences between specific time steps, which help in estimating such changes. This leads to various difficulties in quality control and comparability, especially as the data is collected by scraping.

5.4 Ideas for Future Work

Due to time and capacity limitations, this thesis cannot cover all that can be done with the recipe data collected for this project. There are further topics worth investigating in future work. First of all, the techniques that have been performed during this analysis can be repeated on a larger scale or with further additions, for example:

- An improved scraper for the chefkoch dataset can be implemented to eliminate the scraping errors, which would help to increase the dataset's size and quality.
- The FastText word embeddings have been on a corpus from the cleaned kochbar dataset only and have provided sufficiently good results. However, generating a corpus with twice the size by combining all three corpora might lead to an even better result, which is valuable when building a language model, and can be worth investigating.
- The trend analysis has produced a dataset that shows which recipes have grown the most each month with respect to their popularity metrics. It can provide further insights to use the trained cluster labels for grouping these recipes, as this would enable an analysis of the trending recipe categories.
- Of course, the project will provide more insights if continued for a longer time, as this analysis can only view a time frame limited to six months. Analysing a complete year would be much better for finding all seasonal trends and a second year for verifying those assumptions.
- To extend the trend analysis and confirm its hypotheses, selecting and scraping other recipe websites will be valuable, assuming that they provide an active user community and useful popularity metrics like the dataset from kochbar.
- The analysis has produced datasets with high-dimensional features, especially by one-hot encoding and using word vectors. Before actually deploying such large datasets in a business environment, it can be very useful to perform dimensionality reduction on these datasets to find their most important (statistical) components and reduce their sizes.

These are only a few ideas that can all be performed with the resources at hand to find more patterns and extract knowledge that can be used in the business. This project has successfully presented how these techniques can be employed to achieve valuable results and create business value.
However, there are also other ideas that would require expanding the project's scope by investigating and implementing further techniques:

- With a larger dataset that contains more popularity metrics (which, optimally, are normally distributed), further data mining algorithms can be employed. For example a tool for predicting a recipe's popularity can be built by implementing a regression model. This is useful for assessing the quality of newly written recipes that have not been online long enough to accumulate a sufficient number of popularity metrics yet.

- A more experimental approach can be pursued by implementing a recurrent neural network for training a language model from the recipes. Such a model could then be used to correct typos and clean sections of dirty data in the recipes, and also for implementing an "artificial intelligence" that can generate new recipes. A business value for the latter function might not be clear yet, but an "AI chef" could provide a highly valuable asset for marketing the business in technophile target groups. Such a model could easily incorporate additional information about trending recipes and user preferences depending on the date.

- The websites provide pictures and even videos with some of their recipes. Analysing these images and videos with regards to their quality and their dependencies with the recipe's popularity metrics can be useful to find out how to detect high-quality images and how much these affect a recipe's popularity.

- Recipes presented to this business' customers could also be enriched with more data sources, for example from a video platform like youtube. Together with an extraction of the most frequently used techniques and ingredients in recipes, these videos can be used to create cooking tutorials that users can access when ordering a food box that requires such techniques.

- There are many more recipe websites available in different countries and languages. By using a translation model, those recipes could be integrated into this project's database as well, expanding the dataset by a multiple. Naturally, those websites would rather show food trends in their respective regions, but can still be interesting for German users. In general, recipes could then be provided in different languages, which will be valuable when this project's business expands internationally.

In summary, it is obvious that this project can only provide a short glimpse into all the possibilities that come with such a rich and valuable dataset. Nonetheless, even this short glimpse can be sufficient - if executed correctly - to create business value and new product features.

6 Summary and Conclusion

All sections above already include their own summaries or conclusions to a certain extent. Thus, this section will present a higher-level summary and conclude the project with a few final words.

This project has shown how a complete *Data Science* project can be performed in a business context, starting with the implementation of scrapers for data extraction from different user community domains and ending with the proposal of new product features to create business value - based on data analysis results. The theoretical foundation for different analysis approaches has been researched and explained, and the complete project has been performed following the field's best practices and common guidelines. The CRISP-DM model provides a general process model that helps in structuring a *Data Science* or data mining project, which has been valuable for this project. The initial business and data understanding as well as the definition of goals have been useful to set the project's direction and evaluate its success.

The most difficult challenge during implementation has been the data extraction and cleaning to achieve a sufficient level of data quality, because neither website has provided a clean dataset or any standardisation. Additionally, due to their different data attributes and amounts of user activity, the websites have not been very well comparable to each other. Still, the datasets have been brought to such a size and quality that statistical tests and data mining algorithms have been trained with significant and successful results. The implementation in Python has been open-sourced and provides a detailed look into the programming code necessary for executing this project. The Jupyter Notebook files are particularly convenient for presenting code snippets and adding explanations to clarify the steps and their results.

The project's main objective of performing a multi-domain data analysis to create business value has been achieved successfully and has resulted in multiple promising ideas that can be implemented in the future.

References

[1] Alexa. About Us - Information. Insight. Advantage. https://www.alexa.com/about, (last accessed: 19.01.2018).

[2] Kevin Tynan Anurag Rana, Caitlin Noselli and Courtney Cytryn. Rise of artificial intelligence and machine learning. https://www.bloomberg.com/professional/blog/rise-of-artificial-intelligence-and-machine-learning/, 21.01.2016 (last accessed 05.01.2018).

[3] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA, 2007.

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[5] Robert M Bond, Christopher J Fariss, Jason J Jones, Adam DI Kramer, Cameron Marlow, Jaime E Settle, and James H Fowler. A 61-million-person experiment in social influence and political mobilization. *Nature*, 489(7415):295–298, 2012.

[6] Douglas Busvine. Facebook abused dominant position, says german watchdog. *Reuters Business News*, December 2017.

[7] Tolga Buz. RecipeScraper - Data Science Project for Scraping German Recipe Websites. https://github.com/tbuz/RecipeScraper, 2018 (last accessed: 06.04.2018).

[8] Donald D Chamberlin and Raymond F Boyce. SEQUEL: A structured English query language. In *Proceedings of the 1974 ACM SIGFIDET (now SIGMOD) workshop on Data description, access and control*, pages 249–264. ACM, 1974.

[9] Wilerid J Dixon and J Massey Frank. *Introduction To Statistical Analsis*. McGraw-Hill Book Company, Inc; New York, 1950.

[10] Justin Duke. How To Crawl A Web Page with Scrapy and Python 3. https://www.digitalocean.com/community/tutorials/how-to-crawl-a-web-page-with-scrapy-and-python-3, 29.09.2016 (last accessed 08.08.2017).

[11] Eliot Horowitz Dwight Merriman and Kevin Ryan. MongoDB. https://www.mongodb.com/, 2007 (last accessed: 15.12.2017).

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[13] Tom Fawcett and Foster Provost. Adaptive fraud detection. *Data mining and knowledge discovery*, 1(3):291–316, 1997.

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

[15] Python Software Foundation. json - JSON encoder and decoder. https://docs.python.org/3/library/json.html, 2017 (last accessed: 19.01.2018).

[16] Python Software Foundation. Python Language Reference, version 3.6.1. http://www.python.org, 2017 (last accessed: 19.01.2018).

[17] Python Software Foundation. Regular expressions HOWTO. https://docs.python.org/3/howto/regex.html, 2017 (last accessed: 19.01.2018).

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[19] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(2009):12, 2009.

[20] Hannes Grassegger and Mikael Krogerus. The data that turned the world upside down. *Vice Motherboard, Luettavissa*, 28, 2017. (available on http://motherboard.vice.com/read/big-data-cambridge-analytica-brexit-trump).

[21] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, 1994.

[22] Tony Hey, Stewart Tansley, and Kristin Tolle. The fourth paradigm: Data-intensive scientific discovery. 2009.

[23] IBM Big Data & Analytics Hub. Extracting business value from the 4 V's of big data. http://www.ibmbigdatahub.com/infographic/four-vs-big-data, (last accessed 01.01.2018).

[24] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[25] Inc. Instagram. Wie legt Instagram fest, welche Werbeanzeigen mir angezeigt werden? https://help.instagram.com/173081309564229?helpref=uf_permalink, 2018 (last accessed: 31.03.2018).

[26] ECMA international. The JSON Data Interchange Format. Technical Report Standard ECMA-404 1st Edition / October 2013, ECMA, October 2013.

[27] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python. http://www.scipy.org/, 2001 (last accessed 20.03.2018).

[28] Anthony Wing Kosner. Why Is Machine Learning (CS 229) The Most Popular Course At Stanford? https://www.forbes.com/sites/anthonykosner/2013/12/29/why-is-machine-learning-cs-229-the-most-popular-course-at-stanford/#ea19e9955b7c, 29.12.2013 (last accessed 05.01.2018).

[29] Lukasz A Kurgan and Petr Musilek. A survey of knowledge discovery and data mining process models. *The Knowledge Engineering Review*, 21(1):1–24, 2006.

[30] Doug Lancy. 3D Data Management - Controlling Data Volume, Velocity and Variety. https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf, 06.02.2001 (last accessed 05.12.2017).

[31] Quanzhi Li, Sameena Shah, Merine Thomas, Kajsa Anderson, Xiaomo Liu, Armineh Nourbakhsh, and Rui Fang. How much data do you need? twitter decahose data analysis.

[32] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[33] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[34] Bernard Marr. Big Data: The 5 Vs Everyone Must Know. `https://www.linkedin.com/pulse/20140306073407-64875646-big-data-the-5-vs-everyone-must-know/`, 06.03.2014 (last accessed 01.01.2018).

[35] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[36] Anshul Mittal and Arpit Goel. Stock prediction using twitter sentiment analysis. *Standford University, CS229 (2011)*, 15, 2012. (available on `http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf`).

[37] Christine Moorman. CMO survey. Technical report, Mimeo, Duke University, 2012.

[38] Nasser M Nasrabadi. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.

[39] German Audit Bureau of Circulation (IVW). General Information. `http://www.ivw.eu/englische-version`, (last accessed: 19.01.2018).

[40] German Audit Bureau of Circulation (IVW). Kontrollverfahren im Überblick [german only]. `http://www.ivw.eu/digital/kontrollverfahren-im-ueberblick`, (last accessed: 19.01.2018).

[41] The pandas project. pandas - Python Data Analysis Library. `https://pandas.pydata.org/index.html`, 2017 (last accessed: 19.01.2018).

[42] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[43] Fernando Pérez and Brian E. Granger. IPython: a System for Interactive Scientific Computing. *Computing in Science and Engineering*, 9(3):21–29, May 2007.

[44] Gregory Piatetsky-Shapiro. Knowledge discovery in real databases: A report on the ijcai-89 workshop. *AI magazine*, 11(4):68, 1990.

[45] Chatura Ranaweera and Jaideep Prabhu. On the relative importance of customer satisfaction and trust as determinants of customer retention and positive word of mouth. *Journal of Targeting, Measurement and Analysis for marketing*, 12(1):82–90, 2003.

[46] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[47] Leonard Richardson. Beautiful Soup. `https://www.crummy.com/software/BeautifulSoup/`, 2017 (last accessed: 19.01.2018).

[48] Mohd Sanad Zaki Rizvi. Web Scraping in Python using Scrapy (with multiple examples). `https://www.analyticsvidhya.com/blog/2017/07/web-scraping-in-python-using-scrapy/`, 25.07.2017 (last accessed 08.11.2017).

[49] RStudio. Homepage. `https://www.rstudio.com/`, (last accessed: 04.02.2018).

[50] John Rust and Susan Golombok. *Modern psychometrics: The science of psychological assessment*. Routledge, 2014.

[51] Scrapinghub. Scrapy Community Website. `https://scrapy.org/`, 2017 (last accessed: 19.01.2018).

[52] Tom Shafer. The 42 V's of Big Data and Data Science. `https://www.elderresearch.com/company/blog/42-v-of-big-data`, 01.04.2017 (last accessed 05.12.2017).

[53] SimilarWeb. Our Data. `https://www.similarweb.com/ourdata`, 2017 (last accessed: 19.01.2018).

[54] Tableau Software. tableau.com - Homepage. `https://www.tableau.com/`, (last accessed: 04.02.2018).

[55] Catherine E Tucker. Social networks, personalized advertising, and privacy controls. 2014.

[56] John W Tukey. *Exploratory data analysis*. Reading, Mass., 1977.

[57] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[58] Mark van Rijmenam. Why The 3V's Are Not Sufficient To Describe Big Data. `https://datafloq.com/read/3vs-sufficient-describe-big-data/166`, 07.04.2016 (last accessed 01.01.2018).

[59] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 37–42. ACM, 2009.

[60] Rüdiger Wirth and Jochen Hipp. Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, pages 29–39, 2000.

[61] WolframAlpha. Result page for chefkoch.de. https://www.wolframalpha.com/input/?i=chefkoch.de, (last accessed: 19.01.2018).

[62] Inc. Yelp. Yelp Dataset Challenge - Round 11. https://www.yelp.com/dataset/challenge, 2004-2018 (last accessed: 31.03.2018).

Code Repository

The complete source code for this project has been uploaded to GitHub [7], freely accessible for the open-source community. Adding the complete code for the project into this document would significantly increase the number of pages and only provide poor readability.
The GitHub repository is structured as follows:

1. Scrapers
    - One scraper script for each of the three selected websites, together with additional files that *scrapy* needs to function correctly.
2. Jupyter Notebook Files
    - Detailed Analysis files for each website (`Detailed_Analysis_*.ipynb`)
    - Trend Dataset Generation for chefkoch and kochbar (`Trend_Dataset_*.ipynb`)
    - Corpus Generation and FastText Model Training (`Fasttext_Model_*.ipynb`)
3. Additional files (in separate folders)
    - Exports of data subsets generated during data cleaning and feature extraction
    - Images of plots generated during the analysis
    - Word embeddings trained with FastText and used corpora as `.txt` files

The tables in this thesis document are only excerpts due to space limitations. The complete versions with higher numerical precision can be found in the respective Jupyter notebook files in the GitHub repository [7].

List of Tables

Figures and Additional Tables

Table 15: Spearman's rho: p-values for users (kochbar)

	R	C	$\bar{C}$	$\bar{\star}$	V	F	D	V/R	C/D	F/D
R	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$\bar{C}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
$\bar{\star}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
V	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
D	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02
V/R	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
C/D	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F/D	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00

(C = clicks, D = days, F = favorites, R = recipes, V = votes, $\star$ = rating, $\bar{}$ = mean)

Table 16: Spearman's rho: p-values for recipe attributes (kochbar)

	$\bar{\star}$	c	co	d	f	t	v
$\bar{\star}$	0.00	0.00	0.00	0.00	0.00	**0.45**	0.00
c	0.00	0.00	0.00	0.00	0.00	0.00	0.00
co	0.00	0.00	0.00	0.00	0.00	0.00	0.00
d	0.00	0.00	0.00	0.00	0.00	0.00	0.00
f	0.00	0.00	0.00	0.00	0.00	0.00	0.00
t	**0.45**	0.00	0.00	0.00	0.00	0.00	0.00
v	0.00	0.00	0.00	0.00	0.00	0.00	0.00

(c = clicks, co = comments, d = days, f = favorites, t = preparation time, v = votes, $\bar{\star}$ = avg. rating)

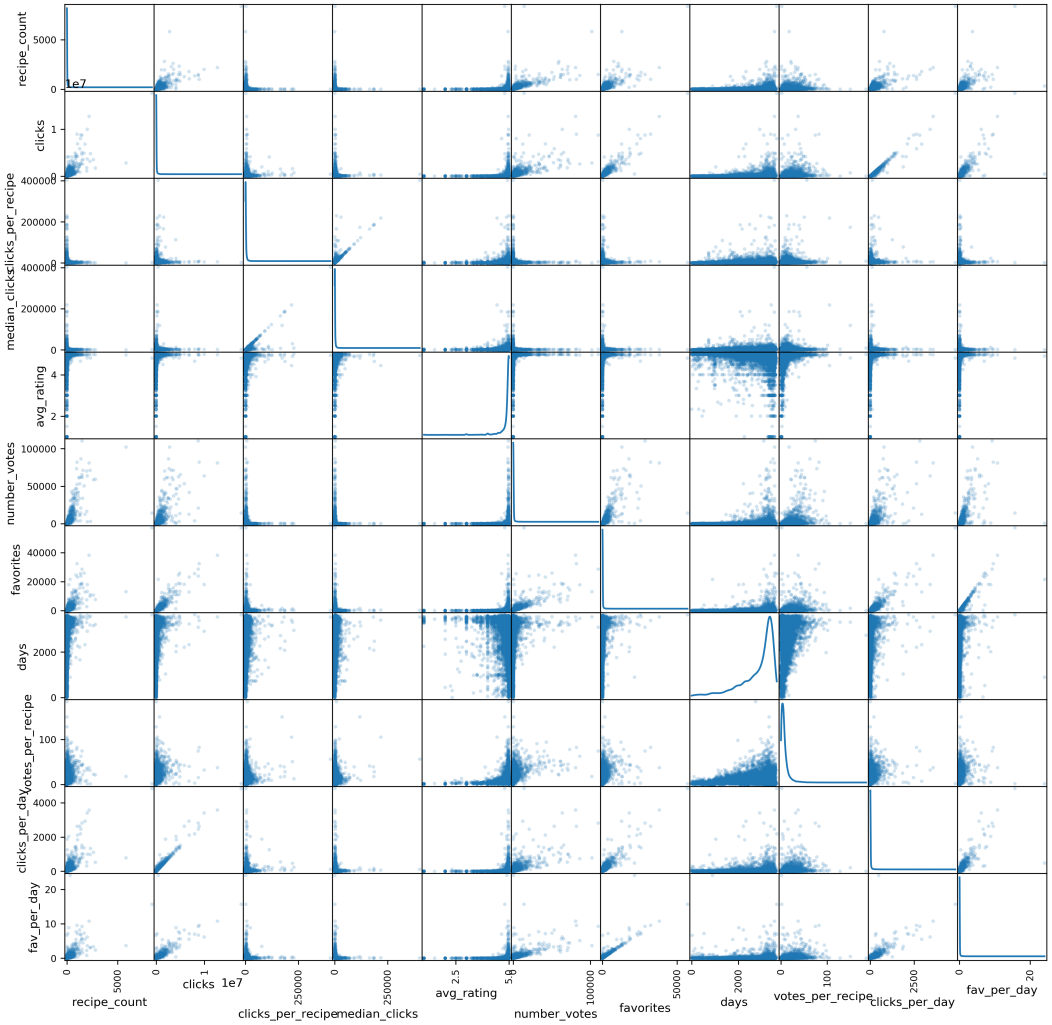

Figure 1: Scatter matrix for user attributes (kochbar)

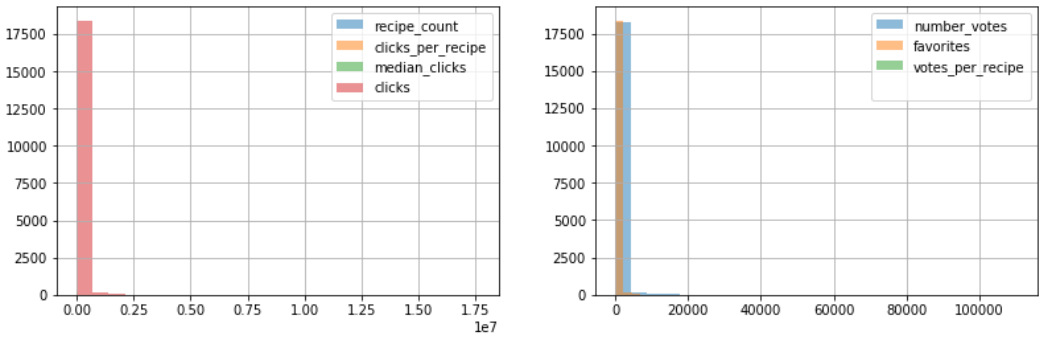

Figure 2: Histograms for user attributes (kochbar)

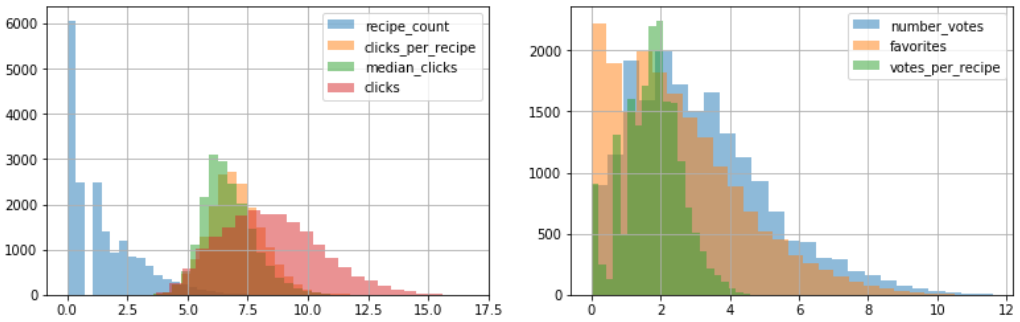

Figure 3: Histograms after logarithmic transformation

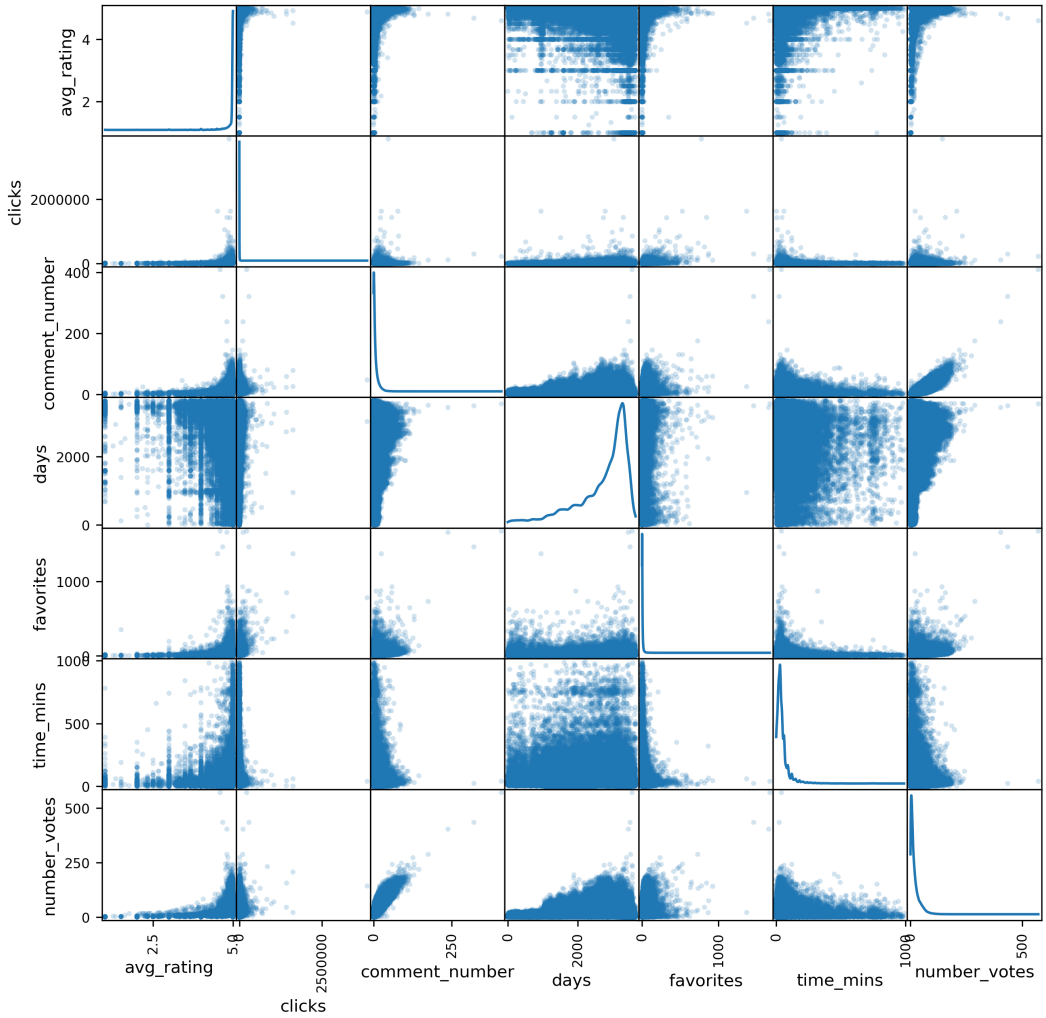

Figure 4: Scatter matrix for recipe metrics (kochbar)

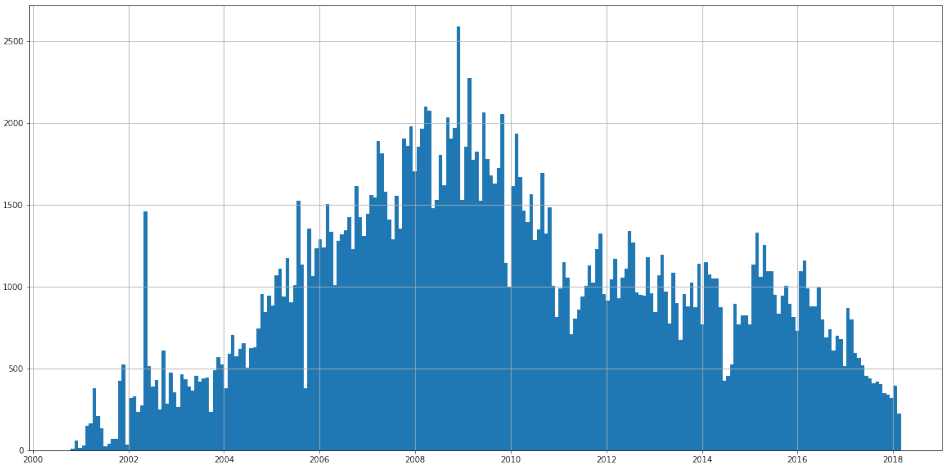

Figure 5: New recipe publications per month on chefkoch

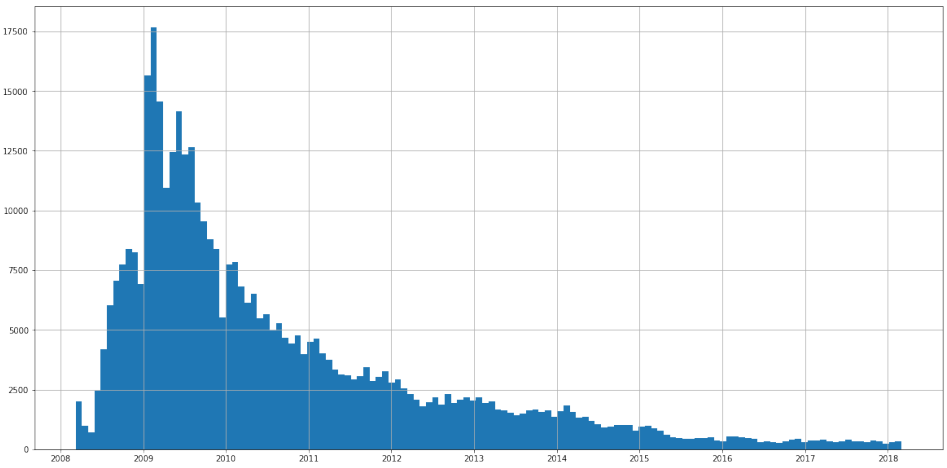

Figure 6: New recipe publications per month on kochbar

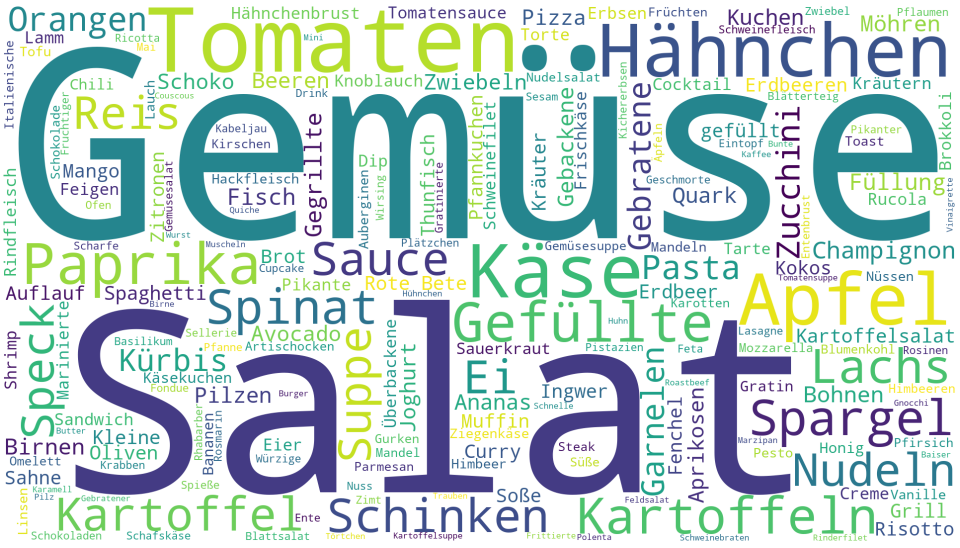

Figure 7: Most frequent words in recipe titles (eatsmarter)

Figure 8: Most popular ingredients on eatsmarter

Figure 9: Most popular ingredients on chefkoch

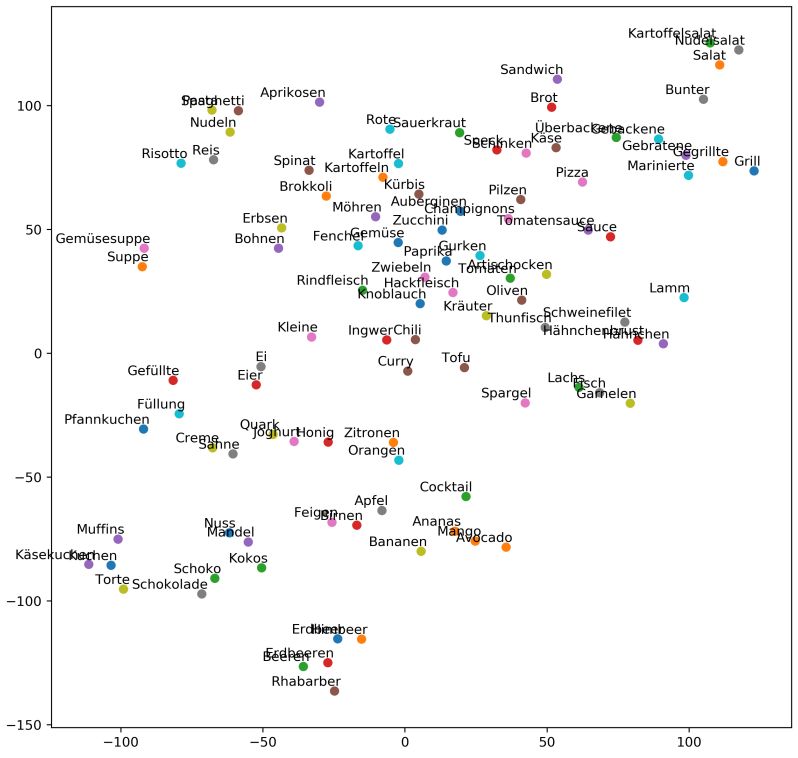

Figure 10: t-SNE results for top 100 recipes (eatsmarter)

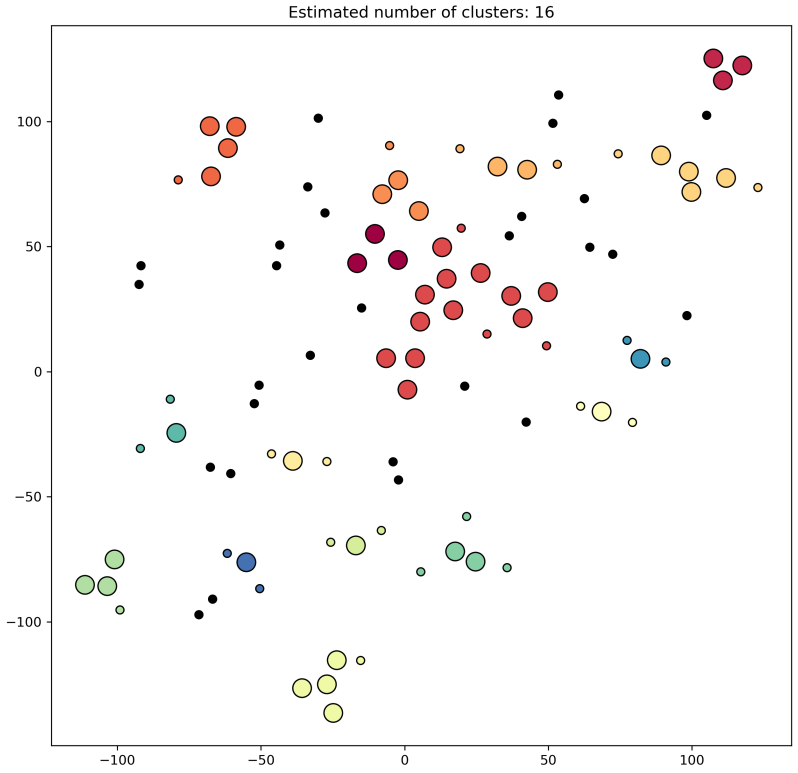

Figure 11: DBSCAN clusters for top 100 recipes (eatsmarter)