Generating New Product Recommendations in Online Shops with Topic Models

Dissertation

By Johanna Fischer

Catholic University of Eichstätt - Ingolstadt, Germany

Ingolstadt School of Management (WFI)

Chair of BA, Sales Management and Marketing

First Advisor:	Prof. Dr. Joachim Büschken
Second Advisor:	Prof. Dr. Thomas Setzer
Date of Disputation:	August 5, 2020

Abstract

Recommending products that are helpful to customers and tailored to their needs is of pivotal importance for successful online retailing. Online purchase data is typically used to generate such recommendations. This dissertation studies two topic models that use purchase data to make product recommendations. The Author Topic Model (ATM) and Sticky Author Topic Model (Sticky ATM) are applied to the purchase data of an online retailer of animal health products, and their predictive performances are contrasted with those of the benchmark methods Unigram, Bigram, and Collaborative Filtering (CF). This work focuses on the generation of new product recommendations. To increase novelty in recommendations, a new pre-processing approach is presented. The data is prepared prior to model application such that more novel products are included in the recommendations. A total of six data preparation variants are tested. The key finding is that topic models are very competitive with the benchmark methods and outperform them with the data preparation variant, where repetitively purchased items (repeat items) and customers with one item transaction (single-item customers) are eliminated from the data. Marketing practitioners should consider this pre-processing when implementing topic models as recommender models in their online shops.

Keywords: recommender systems, topic models, Latent Dirichlet Allocation, online purchase data, data preparation

Contents

List of Figures

List of Tables

List of Abbreviations

ATM	Author Topic Model
CF	Collaborative Filtering
DAG	Directed Acyclic Graph
GoM	Grade of Membership
LDA	Latent Dirichlet Allocation
Log-Likelihood	Logarithmized Likelihood
LSA	Latent Semantic Analysis
MAE	Mean Absolute Error
MCMC	Markov Chain Monte Carlo
MSE	Mean Squared Error
NDPM	Normalized Distance-Based Performance Measure
NMAE	Normalized Mean Absolute Error
PLSA	Probabilistic Latent Semantic Analysis
RMSE	Root Mean Squared Error
ROC	Receiver Operating Characteristic
RS	Recommender Set
SD	Standard Deviation
SKU	Stock Keeping Unit
Sticky ATM	Sticky Author Topic Model
SVD	Singular Value Decomposition

Part I.

Introduction

1. Motivation

1.1. Making Product Recommendations with Online Purchase Data

Online stores with a large product range are very attractive because they offer shoppers more choices to select from. However, a huge variety in products can also overwhelm customers and make purchase decisions difficult. To address this issue, online retailers incorporate recommender systems in their online shops; these systems have become standard features in many shop systems today. Familiar recommender systems, for example, are Amazon's "Customers who bought this item also bought" section or Zalando's display of a set of additional products to "Complete your look." These recommender systems both improve customers' shopping experience and play a vital role in companies' efforts to increase sales and competitiveness. Various studies have already provided evidence for such benefits for both customers and companies (Lee and Hosanagar 2014; Pathak et al. 2010; Fleder and Hosanagar 2009).

To generate product recommendations in online shops, purchase data is typically used. Online purchase data encompasses all past transactions of customers in an online shop. This data provides valuable insights for recommender systems and is beneficial in the context of online recommendations for various reasons. First, it contains information on the co-occurrence of items, i.e., which items are purchased together in a shopping basket. Such co-occurrences are used by recommender systems to identify a suitable set of products for recommendation. Second, it reveals differences in customer preferences. For example, it can be assumed that a customer who orders a new iPhone and the corresponding case by Apple puts more emphasis on style and design than on cost. However, a customer who purchases a Huawei smartphone with a case that was tested for a 12-foot drop seems to be more price-sensitive and safety-conscious. Both customers probably value distinct recommendations. Third, online purchase data is readily avail-

able in firms' data warehouses, from which it can be obtained with minimal effort and at almost no cost. This is important since online product recommendations must be continually updated in real time. For this reason, this dissertation focuses on making product recommendations solely based on online purchase data. Some papers have also tried to include additional product information, e.g., price or textual descriptions (Christidis and Mentzas 2013; Iwata and Sawada 2013). Such information, though, is often not easily available for every product or increases computational complexity.

1.2. Using Topic Models as Recommender Models

In this dissertation, topic models are applied to online purchase data to make new product recommendations in online shops. Topic models were originally developed for application with text data, in particular for finding the latent topics present in large text collections. Application examples in marketing can be found in Büschken and Allenby (2016) and Tirunillai and Tellis (2014), which analyze customer reviews respectively online chatter with topic models to better understand what customers are saying about a company's products and services. These models are not necessarily tied to text data but can also be used for other types of data with a similar structure, such as online purchase data. Specifically, online purchase data is matched to text data by setting items equal to words and customers' purchase histories to text documents.

Scholars have already successfully applied topic models to purchase data to explain future buying behavior. The literature for this can be split into two categories. One stream of literature analyzes *offline purchase data* with topic models. In particular, scanner data of supermarkets (Hruschka 2014; Hruschka 2016; Christidis, Apostolou, and Mentzas 2010) and stationary retailers (Schröder 2017; Ishigaki et al. 2015) has been evaluated for product recommendations and cross-selling activities. A second stream of literature analyzes *online purchase data* with topic models. For instance, Sun et al. (2013) used

group purchasing data from a social media platform to predict which group purchasing event a customer is most likely to join. Iwata and Sawada (2013) presented a topic model that included price information of items and applied it to purchase records from an online service for managing household accounts. Jacobs, Donkers, and Fok (2016) analyzed purchase data from a Dutch online retailer for drugstore products to predict what a customer would buy next. This latter study came closest to this dissertation because similar data and models were used. It will be subject of discussion later in this work.

Topic models are different compared to other recommender models in identifying the *latent topics* that are present in purchase data and how much each customer is interested in those topics. Based on this information individual product recommendations can derived. A *topic* manifests itself as a collection of items which are frequently purchased together across customers. For example, running shoes, sport socks and drinking bottles might often co-appear in orders of customers interested in "running" while charcoal bags and aluminium foils co-appear often in shopping baskets of those interested in "barbecue". The term *latent* refers to the fact that topics must not be clearly assigned to one specific subject. Instead, it is possible that a topic consists of products of different categories which at first glance seem to have little connection. For instance topic models could reveal a topic "surfer's clothing and barbecue" because people who are interested in both activities generally like to spend much time outside.

In this dissertation, the topic models ATM and Sticky ATM (Büschken and Allenby 2016) are selected for analysis. Both models include authorship information, i.e., customers' entire purchase histories are considered instead of single orders. This provides the models with a more informative data basis. In addition, the Sticky ATM model accounts for stickiness at the item level. This means that the same topic assignment can carry over for a certain sequence of items ordered by a customer. Such an assumption might better

reflect customers' purchasing behavior of buying by categories. Both topic models are compared to the benchmark models Unigram, Bigram, and CF. Unigram is the simplest method, which calculates marginal probabilities across all ordered items. Bigram is the recommender method currently used by the online retailer considered for analysis in the empirical part of this dissertation. CF is the most frequently applied method in practice.

1.3. Extending Existing Research

The literature is increasingly concerned with how to generate useful recommendations for customers. Product recommendations often fail to meet the needs of customers or do not really represent a true recommendation because the displayed products are already known to a shopper. A central reason why recommender systems miss their target is the evaluation method by which they are assessed and selected. Recommender systems are typically evaluated based on their *accuracy* in predicting the next products a customer will buy. This procedure favors the selection of models that recommend popular items rather than unpopular items. The latter, however, are generally less likely to be discovered by customers and thus are more likely to be novel and useful to them. Research is increasingly finding that *accuracy* alone is not sufficient and that an evaluation *beyond accuracy* should be carried out (McNee, Riedl, and Konstan 2006; Adamopoulos 2013; Burke, Felfernig, and Göker 2011, p. 16; Jannach, Lerche, et al. 2015, pp. 428-429; Gunawardana and Shani 2009). This dissertation follows just such an approach. To move beyond accuracy, it presents a new *pre-processing* approach in which the input data is prepared *prior* to model running such that only the data relevant for new product recommendations is fed into the models. This new approach provides an alternative view to the more common *post-processing* approaches in the literature. Those approaches re-rank the final results of a recommender algorithm *after* model running to increase the representation of novel items in recommendations.

2. Research Objective and Structure

The main objective of this dissertation is to investigate the application of topic models to online purchase data for the purpose of making *new* product recommendations. Specifically, the following questions are addressed:

1. Are topic models effective in recommending novel items to customers? How do they perform when predicting non-novel items? Are there recommender models that are better suited for the one task or the other?

2. How do the two topic models perform? Can the Sticky ATM model outperform the ATM model?

3. Which data preparation is most suitable for generating new product recommendations? And why? Are there differences between the recommender models?

The present work comprises four parts. Part I, *Introduction*, describes the motivation behind the topic of this dissertation. It also discusses the research objective and structure of the work. Part II, *Theoretical Foundation*, provides an overview of recommender systems and topic models. The objective is to lay the necessary theoretical foundations for generating new product recommendations with topic models in the empirical analysis. In Part III, *Empirical Analysis*, the data used for analysis, the distinct data preparation variants tested, and the procedure for splitting the data into training and test data are described. The various recommender methods for the model comparison, including the performance measure to evaluate the models, are explained in detail. Finally, the results of the empirical analysis are presented and discussed critically. Part IV, *Conclusion*, summarizes the main findings of this dissertation and offers ideas for future research.

Part II.

Theoretical Foundation

3. Recommender Systems

With the rise of the Internet in the mid-1990s, e-commerce websites began to develop in addition to stationary retailing. One of the most popular e-commerce companies is the American firm Amazon. With revenue of about 233 billion US Dollars in 2018 (Rabe 2019), it is among the largest online retailers in the world. The success of Amazon largely rests on a vast product assortment of about 280 million products (US market) (Jordan 2017) originating from distinct categories such as electronics, apparel, and books. Even though a large product offering is valued by customers because it provides more freedom and self-determination, it might also overwhelm customers (Ricci et al. 2011, p. 2). There is a pressing need for recommender systems to help customers make the best choices. In particular, such systems support consumers by pointing out new and not yet experienced items that might be relevant to them. Such personalized recommendations do not only lead to higher customer satisfaction and loyalty (Srinivasan, Anderson, and Ponnavolu 2002) but also to higher sales for companies. For instance, some studies show that consumers are twice as likely to select a product when it is recommended (Senecal and Nantel 2004; Postma and Brokke 2002). Furthermore, recommender systems increase the diversity of products purchased by individuals (Lee and Hosanagar 2014; Fleder and Hosanagar 2009), leading to additional sales of niche products. All in all, the use of recommender systems is vital to online retailer to survive in today's competitive marketplace. For more than two decades, therefore, Amazon has continuously strived to improve its recommender algorithms, making them ever more intelligent and helpful to people (Smith and Linden 2017).

The chapter provides an introduction to the most important theoretical basics and terms associated with recommender systems. It starts with a definition of recommender systems and their common goals (3.1). It continues with an explanation of the required data sources for recommender systems (3.2) and an overview of the available recommender

techniques that are currently used (3.3). Finally, it elaborates on how to evaluate recommender systems (3.4), including a discussion on why accuracy metrics alone are not enough to make useful product recommendations to customers.

3.1. Definition and Goal

Recommender systems were developed to provide helpful recommendations to users. Ricci et al. (2011, p. 1) defined recommender systems as follows:

> Recommender Systems [...] are software tools and techniques providing suggestions for items to be of use to a user.

This definition is very general and applies to recommender systems in distinct application contexts, including that of product recommendations in this dissertation. In the above definition, the term "items" refers to the specific objects that should be recommended. This can be *products* in an online store, *movies* or *songs* on a streaming platform, *content* on a news website, or *friends* to follow on social media. Such items are displayed by a recommender system to an individual "user" such as an *online shopper*, *viewer*, *listener*, *reader* or *social media user* who is faced with an immense variety of alternatives. Most important in the above definition is that suggestions need to be "of use," that is, they should be relevant somehow to a customer and catch his or her interest. How such usefulness can be improved and measured for a recommender system is discussed in greater detail in Chapter 3.4.3.

With respect to the goal of recommender systems, one needs to differentiate between users and companies. From a user's point of view, the main goal of a recommender system is to make users aware of useful items and draw their attention to them. This reduces the cost of their search for items and ultimately results in higher user satisfaction.

From a company's point of view, the main goal of a recommender system is to increase sales and profit. This can be achieved by displaying appropriate recommendations at the right time and place.

3.2. Data Sources

Recommender systems make use of various kinds of data to build their algorithms. The type of data can be diverse and come from distinct knowledge sources. Whether a data source can be ultimately exploited or not depends heavily on the selected recommendation technique (see Chapter 3.3). In general, data about items and users run into the recommender models (Ricci et al. 2011, pp. 7-10). With respect to *items*, simple ratings can be used. Some techniques also consider item information such as price, brand, category, and available attributes. Other techniques include textual product descriptions and tags added by users, such as how they feel about a product (e.g., "absolutely satisfied" or "would not buy it again"). Systems that exploit information on *users* make use of sociodemographic attributes such as age, income, gender, occupation, and place of residence. Based on such characteristics, the similarity among users can be computed and similar users categorized in homogeneous groups that receive the same recommendations.

The most frequently discussed data basis in literature is *ratings*, which can take different forms (Ricci et al. 2011, p. 9; Bodapati 2008, pp. 78-79):

- *Numerical* or *ordinal ratings*: A user votes for an item in form of a numeric value (e.g., 1-5 stars on Amazon) or on an ordinal scale (e.g., strongly agree, agree, neutral, disagree, strongly disagree).
- *Binary ratings*: A user expresses positive ("like") or negative ("dislike") valence for an item. Such entries are typically labeled with either 1 and 0, respectively.
- *Unary ratings*: A user does not explicitly specify a rating, but indicates it through

his or her behavior. For instance, a customer who orders a product in an online shop can be assumed to like that product and therefore choose to buy it. This type of rating is special in that only positive valences are observed, but nothing is known about which items a user did not like. This is different from the other types of ratings, where negative feedback can be expressed through a low rating or "dislike" (Bodapati 2008, pp. 78-79; Hu, Koren, and Volinsky 2008, p. 264).

In the first two types of ratings, a customer explicitly states a rating; in the third type, in contrast, the rating is only expressed implicitly through behavior. The literature also uses the terms *explicit rating* and *implicit rating* to point out this difference. The online purchase data analyzed in this dissertation is of the unary type.

3.3. Recommendation Techniques

Depending on the kind of data used, five different classes of recommendation techniques can be differentiated: collaborative filtering models, content-based recommender systems, knowledge-based recommender systems, demographic recommender systems, and hybrid recommender systems (Aggarwal 2016, pp. 8-20, 29-224; Ricci et al. 2011, pp. 10-14).

The most common and widely used method in practice are *collaborative filtering models*. Such models rely on rating data and make recommendations on the basis of similarities between users *or* items. These are referred to as *user-based collaborative filtering* or *item-based collaborative filtering*. Collaborative filtering methods can be further subdivided into *memory-based* (or *neighborhood-based*) *methods* and *model-based methods*. The first method makes use of simple similarity measures and matching procedures to identify neighboring users or items. In the second method, machine learning and data mining techniques, such as decision trees, rule-based methods, or Bayesian methods, are applied.

Another category includes *content-based recommender systems*, which rely on item de-

scriptions. The term "content" refers to such descriptions. For instance, an online book retailer could use product details such as author, genre, and price or extract relevant keywords from a book description to find similar items for recommendation. There is one central difference from the previous method: in collaborative filtering models, similarity is based on the rating data of *all other* users, while in content-based recommender models, similarity is based on the items liked by *one* user only, that is, the rating data of other users is not required. The latter method is therefore often used in cold-start scenarios where little information on the behavior of other users is available.

A third class of recommender approaches includes *knowledge-based recommender systems*. Such recommender systems are used for items that are not purchased very often, such as automobiles, real estates, or expensive luxury goods. For such items, not sufficient rating data may be available and also preferences may change over time, which does not lead to reasonable recommendations when using historical data. Unlike the previous two approaches, in knowledge-based recommender systems, users take an active role in specifying their requirements ("knowledge") for a new item. This is usually done through an interface where different options can be selected, e.g., as in a car configurator. The specified customer requirements are then matched to different item descriptions and the most similar items are offered to the customer.

A recommender technique that leverages demographic information about users is called a *demographic recommender system*. Even though this approach cannot be used to recommend specific items and does not produce the best results on a stand alone basis, it significantly improves recommendations when combined with other recommender systems.

This leads to the last approach, *hybrid recommender systems*, which combines various

aspects of the previous methods in one new method. The objective is to benefit from the respective strengths of the single methods and at the same time to compensate for their weaknesses. In the literature, a large variety of hybrid recommender systems (Ansari, Li, and Zhang 2018; Liu, Lai, and Lee 2009; Burke 2007) has emerged in recent years.

3.4. Evaluation

As presented in the previous chapter, a variety of recommendation techniques are studied by researchers and used on commercial online platforms. To compare those recommendation techniques with each other and find the one that suits a particular use case best, evaluation methods are needed against which the different recommendation techniques can be assessed.

This chapter provides an overview of the most common evaluation methods used in practice and by the research community today. It starts by distinguishing evaluation methods in *offline evaluation* and *live user experiments*. Most of the published papers focus on offline evaluation, which is also applied in the empirical part of this work. This approach is further deepened by surveying various *accuracy metrics*, which are part of the offline evaluation process. Some recent works, however, criticize accuracy metrics as not leading to the most satisfying experience for a customer when used alone. Instead, further dimensions that contribute to usefulness of recommendations should be considered. The last subsection is dedicated to this discussion and moves *beyond accuracy* by reviewing a range of non-accuracy measures, including novelty and serendipity.

3.4.1. Offline Evaluation and Live User Experiments

The evaluation of recommender systems can be conducted either by using offline evaluation, live user experiments, or a combination of both (Herlocker et al. 2004, pp. 12-13; Shani and Gunawardana 2011, pp. 257-297; Aggarwal 2016, pp. 225-254). These two

basic forms of evaluations, including their corresponding benefits and shortcomings, are reviewed more in detail below.

Offline evaluation deploys historical data, such as online purchase data, and uses it to predict future user behavior. The historical data set is partitioned into training data and test data, where the training data is used to build the model and the test data is used to calculate the predictive performance of the model. Test data is only considered at the very end and does not enter the training process. Otherwise, predictive performance is typically overestimated. To compute the predictive performance, a wide variety of accuracy metrics is available; these are discussed in greater detail in the next section. Offline evaluation is the simplest form of assessing the performance of recommender systems and is mostly used in the design phase to filter out inappropriate recommender models that are not further tested in the more costly live user experiments. The big advantage of offline evaluation is that it does not require a set-up that has real users. This saves time and costs. The downside is that only questions on predictive power can be answered and actual reactions of customers to recommendations are not measured. Nevertheless, due to its simplicity, offline evaluation remains the most widely accepted evaluation approach in the exploration of recommender systems.

Unlike offline evaluation, which makes use of historical data, *live user experiments* involve users in the evaluation process. The live user experiments are further divided into online evaluation and user studies (Shani and Gunawardana 2011, pp. 263-267; Aggarwal 2016, pp. 227-229). These differ in the way in which users are recruited. In the *online evaluation* users are real users of recommender systems, e.g., shoppers in an e-commerce shop, and their behavior when interacting with such systems is measured without their knowledge. In *user studies*, users are actively recruited and are aware of being part of an experiment. Users are asked to interact with different recommender systems and

then provide their feedback about them, e.g., in a questionnaire or personal discussions. While user studies provide deeper qualitative insights about recommender systems, their results can be biased because of users' active awareness that they are participating in a test study. Online evaluations are far less prone to such biases and are the only method that measures the true response of users to recommender systems.

3.4.2. Accuracy Metrics

Offline evaluation makes use of accuracy metrics to compare the performance of different recommender systems. According to del Olmo and Gaudioso (2008, p. 792), such metrics measure "the quality of nearness to the truth or the true value achieved by a system." Selecting an appropriate accuracy metric, however, seems to be very challenging. Herlocker et al. (2004) described two major challenges. First, researchers are confronted with a range of questions that are not sufficiently answered in the currently published literature. These include:

> Will a given metric measure the effectiveness of a system with respect to the user tasks for which it was designed? Are results with the chosen metric comparable to other published research work in the field? Are the assumptions that a metric is based on true? Will a metric be sensitive enough to detect real differences that exist? How large a difference does there have to be in the value of a metric for a statistically significant difference to exist? (Herlocker et al. 2004, p. 19)

Second, the selection of a suitable accuracy metric is further complicated by the large number of published metrics. There is no standardized metric as of yet that researchers commonly refer to. For a better overview, though, Herlocker et al. (2004) classify the various accuracy metrics into three categories: predictive accuracy metric, classification accuracy metrics and rank accuracy metrics. A similar categorization can be found, for

instance, in Jannach, Zanker, et al. (2011) or del Olmo and Gaudioso (2008). Each of these categories is briefly explained.

- *Predictive accuracy metrics* measure how close the predicted rating of a recommender system is to the user's true rating. The most popular measure is the mean absolute error (MAE), which calculates the average absolute deviation between the predicted scores and the actual rating values given by users. Alternatively, some direct variations of MAE, such as mean squared error (MSE), root mean squared error (RMSE), and normalized mean absolute error (NMAE), can be used. Predictive accuracy metrics are typically applied for recommender systems with explicit rating data; for example, they can be used in movie recommender systems to evaluate the predictions of star-ratings.

- *Classification accuracy metrics* measure how often a recommender system makes correct or incorrect decisions about whether a user likes an item. This requires the recommender system to generate a list of relevant items for a given user. If the user's next selected item is found in this recommendation list, it is considered a correct decision; otherwise, it is an incorrect decision. In other words, classification accuracy metrics focus on whether a recommender system properly predicts that a user will add an item from a recommendation list into the shopping cart. The best-known classification metrics are precision, recall, and F1 metric, which combines the former two into one metric. Also, ROC analysis is mentioned as an alternative metric in the literature.

- *Ranking accuracy metrics* extend the classification accuracy metrics by taking an item's relative position in a recommendation list into account. This means that a correct prediction for an item in the first position receives a greater reward than correct predictions for items in the lower positions. Commonly applied ranking metrics are correlation metrics, half-life utility metrics, lift indexes, and the NDPM

measure.

To sum up, all three accuracy metrics are used to measure different tasks. Predictive accuracy metrics are suitable when the closeness of predicted ratings to true ratings should be identified. Their main emphasis lies in accurate *rating predictions*. Classification and ranking accuracy metrics focus on measuring the ability of a recommender system to make successful *decisions*. While classification accuracy metrics view all items in a recommendation list as equally good, ranking accuracy metrics additionally consider the ranking order of items in the list.

3.4.3. Beyond Accuracy

The previous section showed that there exists a large variety of different accuracy metrics for the evaluation of recommender systems. An increasing number of scholars, however, have noted that using these metrics alone does not lead to the most useful recommendations for customers (McNee, Riedl, and Konstan 2006; Adamopoulos 2013; Burke, Felfernig, and Göker 2011, p. 16; Jannach, Lerche, et al. 2015, pp. 428-429; Gunawardana and Shani 2009).

The following example illustrates this: Imagine you are buying groceries from an online retailer on a regular basis. On this retailer's website the recommender system only displays products that you are already familiar with through previous purchases. Even if these recommendations are exactly the products you are going to buy next – that is, the accuracy of this recommender system is one hundred percent – this would still be a poor recommender system because it does not inform you about new, not yet experienced products.

Researchers have therefore proposed to move beyond evaluation based on accuracy and consider additional aspects that contribute to usefulness. The following table lists some

frequently discussed measures (Kaminskas and Bridge 2016; Aggarwal 2016, pp. 231-235; Shani and Gunawardana 2011, pp. 281-293; Herlocker et al. 2004; Ge, Delgado-Battenfeld, and Jannach 2010; Kotkov, Wang, and Veijalainen 2016):

Measure	**Explanation**
Novelty	Measures whether a recommendation is a new possibility for a user.
Serendipity	Measures how positively surprised a user is by a new recommendation.
Diversity	Measures in a set of recommendations how diverse the items are.
Coverage	Measures the proportion of items that a recommender system can recommend.
Confidence	Measures how much faith a system has in its recommendations.
Trust	Measures how much faith a customer has in a sytem's recommendations.
Privacy	Measures how much private information is disclosed through a system's recommendations.
Learning Rate	Measures how quickly (i.e., with how much learning data) a recommender algorithm can produce good recommendations.

Table 1: Measures for Improving Recommendation Usefulness

The goal of this dissertation is to generate product recommendations that are *new* to customers. The measure of *novelty* should therefore be examined in greater detail. According to Kapoor et al. (2015) and Kotkov, Veijalainen, and Wang (2016), novelty can have three different meanings in recommender systems:

- *Item novel to recommender system*: A new item that was added to the product program.
- *Forgotten item*: The user has forgotten that he consumed the item some time ago.
- *Unknown item*: The user has not consumed the item before.

The third meaning is the most common in the literature and is used in this dissertation.

Closely related to novelty is *serendipity*. Rather than displaying items that a user did not know before, serendipity aims at recommendations that truly surprise a user. For instance, if a customer always buys books from one writer and a recommender system

recommends him a newly published book by this writer, it would be novel but not necessarily serendipitous. When the same user, however, is recommended a book from a completely different genre and he discovers that it appeals to him, then this recommendation would be serendipitous. Recommendations that are serendipitous are, by definition, also novel (Adamopulos and Tuzhilin 2014). For this reason, both measures are frequently discussed in literature together. Nevertheless, it is important to clearly differentiate between these two concepts when it comes to their measurement. While novelty measures whether an item has not been discovered yet, serendipity additional measures whether an item is unexpected for a user. For example, Ge, Delgado-Battenfeld, and Jannach (2010) and Maksai, Garcin, and Faltings (2015) conceptualize serendipity by dividing it into "unexpectedness" and "usefulness". Other authors give slightly different definitions. Zhang, Séaghdha, et al. (2012) describe serendipity as the "unusualness" or "surprise" of recommendations. Aggarwal (2016, p. 234) views serendipity as a "departure from obviousness". In the remainder of this section, novelty and serendipity are analyzed in greater detail. The literature does not always clearly differentiate between those two concepts. For this reason, it makes sense to review them together. First, some research to increase novelty and serendipity in recommendations is presented. Subsequently, various methods for measuring novelty and serendipity are discussed.

Improving Novelty and Serendipity. To improve novelty and serendipity, researchers modify their recommender algorithms in various ways. First, a very simple approach is to *filter out non-relevant items* from a recommendation list (Shani and Gunawardana 2011, pp. 286). For novelty, those would be items that a customer already knows, e.g., items the customer has previously purchased. Still, though, such an assumption might be too simplistic and not capture novelty in its entirety. A customer probably knows far more products than just the ones purchased. With respect to serendipity, non-relevant items would be "obvious" items. Those could be, for example, items from the same man-

ufacturer, product category, or brand. Defining what is obvious, however, can be difficult because it might differ from user to user and change over time. Another approach builds on the idea of *item popularity*, i.e., that popular items are generally less likely to be new and their influence should therefore be reduced. For example, Herlocker et al. (2004) presented a new ranking approach that gives less weight to popular items:

> An alternative would be to divide each [item] probability [of a user] by the probability that an average member of the community would like the item, and re-sort by the ratio. Intuitively, each ratio represents the amount that the given user will like the product more than most other users. Very popular items will be recommended only if they are likely to be exceptionally interesting to the present user. Less popular items will often be recommended, if they are particularly interesting to the present user. This approach will dramatically change the set of recommendations made to each user, and can help users uncover surprising items that they like. (Herlocker et al. 2004, p. 43)

Similar work was done by Abdollahpouri, Burke, and Mobasher (2019) who developed a re-ranking approach for long tail promotion. Their approach can be applied to the output of any recommender system as a post-processing step and increases the representation of less popular items in recommendations. A good discussion on further related works, promoting less popular items from the long tail, can be found in Kaminskas and Bridge (2016, pp. 2:13-2:14). Third, another strand of literature focuses on the *similarity* of recommended items to a user profile. The idea is that items that are more similar to the ones purchased by a customer are less likely to be surprising and should therefore be avoided as recommendation. Instead, more distant, dissimilar items should be recommended. For example, Adamopulos and Tuzhilin (2014) proposed a new approach that recommends unexpected items based on how distant they are from the set of items

expected by a user. Similarily, Vargas and Castells (2011) developed a formal framework for improving distance-based item novelty.

Measuring Novelty and Serendipity. With respect to measuring novelty and serendipity, the most obvious way would be to conduct some user studies (Shani and Gunawardana 2011, pp. 285-288; Aggarwal 2016, pp. 233-234). For instance, novelty could be measured by explicitly asking users whether they are already familiar with a recommended item. For serendipity, additional questions on the unexpectedness of recommendations would be necessary. The fraction of recommendations that are novel respectively serendipitous can then be used as an evaluation measure. Although such studies provide meaningful insights, they are often not conducted. The reason for this lies in the many challenges that those studies entail (Kaminskas and Bridge 2016, pp. 2:17), e.g., recruiting a sufficiently large number of participants, formulating clear survey questions, or designing the correct experimental set-up. Specifically, serendipity has been reported in the literature to be difficult to explain to study participants and is therefore often omitted from analysis. Some understanding is therefore also gained through offline evaluation by designing the appropriate metrics. Kotkov, Wang, and Veijalainen (2016) differentiated *component metrics*, which measure subcomponents such as novelty or unexpectedness, and *full metrics*, which measure serendipity as a whole. In their paper, they provided a good overview of the most common metrics. One major criticism in the literature, however, is that to date, there is no widely accepted metric that captures all the facets of novelty and serendipity together. Even full metrics often disregard some subcomponents. Many researchers consider this an open issue that has yet to be solved. For this reason, Herlocker et al. (2004) gave rather vague recommendations on how to improve currently used metrics. A good novelty metric should capture how well a recommender system makes a user aware of previously unknown items. Good serendipity metrics should measure the extent to which recommendations broaden a user's interests over time.

4. Topic Models

Topic models are statistical models for automatically extracting the latent themes, referred to as *topics*, that are hidden in large collections of unstructured data. These models have increased in popularity over the last decade because of the unparalleled growth of data and the inability of humans to process big data manually. The first topic model was introduced by Blei and Jordan (2003) which is called Latent Dirichlet Allocation (LDA). It was developed for application to text data, in particular for finding the main topics present in newspapers and scientific articles. Later papers also apply topic models to online text sources, e.g., blog entries (Mehrotra et al. 2013; Yano, Cohen, and Smith 2009), customer online reviews (Büschken and Allenby 2016), and social media conversations (Jaradat and Matskin 2019; Abinaya and Winster 2014).

Topic models are not necessarily tied to text data and have been extended to various other data sources, such as images (Wang, Blei, and Li 2009; Rasiwasia and Vasconcelos 2013; Zhang, Lu, et al. 2011), videos (Hendel, Weinshall, and Peleg 2010; Niebles, Wang, and Fei-Fei 2008; Wang, Ma, and Grimson 2007), and music data (Hu 2009; Hu and Saul 2009). A relatively new research stream in marketing applies topic models to purchase data with the objective of predicting future customer behavior (Jacobs, Donkers, and Fok 2016; Hruschka 2014; Hruschka 2016; Ishigaki et al. 2015; Schröder 2017; Sun et al. 2013). This last application of topic models is further investigated in the empirical section of this dissertation.

This chapter focuses on the LDA model because it is the most basic topic model, and all other topic model extensions that have been developed in recent years rely on this model. The language of text data is used to explain LDA, as this is the most common across the literature. For text data, the entities *words* or *terms*, *documents*, and *corpus* are used (Blei, Ng, and Jordan 2003, p. 995). The basic unit of a text is its *words*. Building a

sequence of different words results in a *document*. For instance, a document could be *one* paper or *one* review on a website. The collection of all documents comprises the *corpus*.

The remainder of this chapter is organized as follows: First, the basic idea and assumptions (4.1) of LDA are explained and compared with related models (4.2). It follows a precise description of the technical details (4.3) as well as how posterior inference (4.4) via Gibbs sampling (4.5) is conducted for LDA. After this, an overview of different evaluation methods is presented (4.6); these are necessary to assess the results of the LDA model. This chapter is supplemented by the detailed appendix A. 1 - A. 3, to which references are made at the appropriate places in the text.

4.1. Basic Idea and Assumptions

The basic idea of LDA (Blei, Ng, and Jordan 2003, p. 996) is that each document is characterized by its own mixture of topics (θ_d), and each topic is characterized by a distribution over words (ϕ_k). Consider the example of hotel reviews on booking.com. In those reviews, people generally talk about topics such as the *room*, *staff*, *breakfast*, *dinner*, *spa area*, *location*, etc. However, not every guest will write about all those topics equally. One review might be dominated by the description of a bad experience at *dinner* and an extensive criticism of how the *staff* handled the problem. Another review might stress the wonderful *spa area* and the excellent *location* of the hotel close to bars and sightseeing attractions. Both reviews will have their own mixture of topics, which are obviously very different. In addition, each revealed topic is described by distinct words. For instance, in the topic *room*, words such as "bed," "mattress," "comfortable," "view," or "quiet" might prevail, whereas in the topic *location*, words such as "central," "sightseeing," "close," "walking," or "airport" might be more likely. The output of the LDA model provides us with this information. A huge advantage of LDA is that the data does not require any labeling in advance. Instead, the model is left alone to learn the latent topics

from the original text data. This is referred to as "unsupervised learning."

According to Blei, Ng, and Jordan (2003) and Blei (2012), LDA makes the following assumptions:

- **"Bag-of-words"** or **exchangeability assumption**: This means that documents are seen as a collection of words without any structure ("bag-of-words"). Thus, the order of words in a document does not matter and can be arbitrarily changed.
- **The order of documents does not matter**: This assumption is less frequently stated but is related to the first assumption. Besides words, documents can also be exchanged, meaning that the order of documents in the text corpus can be neglected.
- **The number of topics is assumed to be known and fixed**: Before running the LDA model, the number of topics needs to be specified.
- **Topics are uncorrelated**: Topics are independent from one another and do not show correlations. The independence of topics comes from the selection of a Dirichlet distribution for the prior distribution. To model interdependence between topics, a logistic normal distribution is required as in the correlated topic model (Blei and Lafferty 2007).

Since the introduction of the LDA, many topic model extensions have been developed that relax and extend the above stated assumptions, in order to discover more sophisticated structures in text. A good overview of the most common extensions can be found on David M. Blei's website `http://www.cs.columbia.edu/~blei/`.

4.2. Delimitation to Related Models

The LDA is frequently discussed in the context of other related models. The following figure provides an overview of those related models and compares them with the LDA model.

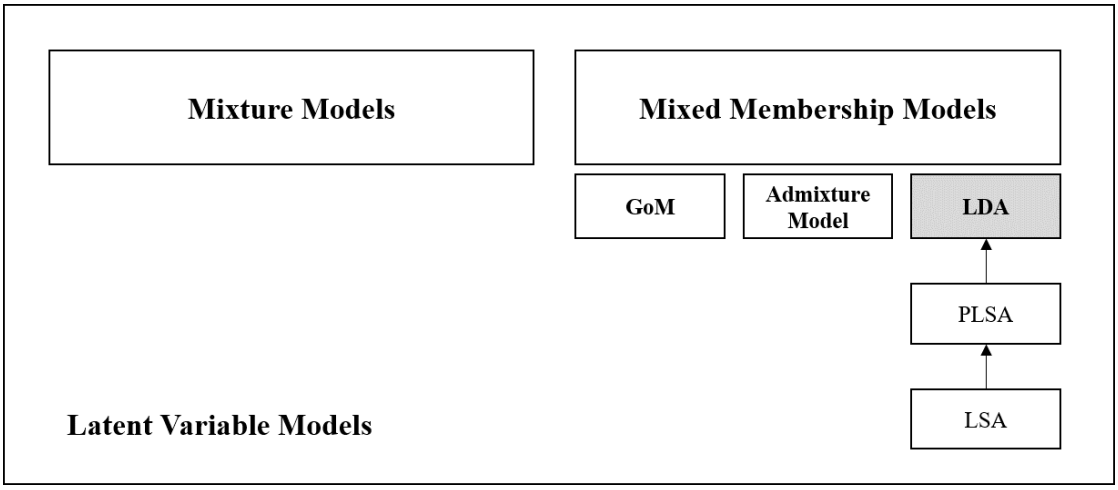


Figure 1: Models Related to LDA[1]

LDA is a Latent Variable Model. LDA belongs to the wider field of latent variable models. Blei (2014, p. 203) defines latent variable models as "probabilistic model[s] of hidden and observed variables, where the hidden variables encode hidden patterns in [the] data". In LDA, the *hidden patterns* to be found are the *topics* that are present across different text documents.

LDA is a Mixed Membership Model. LDA can be summarized under the more general framework of mixed membership models. This framework was created by Erosheva (2002) and Erosheva, Fienberg, and Lafferty (2004) and encompasses three different models: the Grade of Memership (GoM) model, the Admixture model and the LDA model. The GoM model was the first example of a mixed membership model and was developed by Woodbury, Clive, and Garson (1978) for classification of medical diagnosis data. In

[1] Author's own illustration, based on subsequent elaboration.

the early 2000s, the Admixture (Pritchard, Stephens, and Donnelly 2000) and the LDA (Blei, Ng, and Jordan 2003) models were developed simultaneously. Both models address clustering problems, albeit in distinct application fields: the Admixture model in genetics and the LDA in text analysis. All three models assume that an observational unit (document) does not only belong to a single cluster (topic) but to all clusters in different proportions (Airoldi et al. 2015, pp. 3-4). The word "mixed membership" alludes to this fact, meaning that each document is characterized by a mixed membership of different topics.

LDA is the Further Development of LSA and PLSA. The Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (PLSA) can be viewed as the predecessors of the LDA model. The LSA was developed by Deerwester et al. (1990) in the context of information retrieval from text data. The basic idea is to take a term-document matrix and decompose it, using singular value decomposition (SVD), into a term-topic matrix and topic-document matrix. This leads to a significant reduction in dimensionality. PLSA, as its name suggests, is the probabilistic variant of LSA (Hofmann 1999). The model uses a probabilistic approach instead of SVD to find the latent topics. Probabilistic models are set through a generative process that explains how words in documents are generated based on hidden topics. This generative process also enables the estimation of unknown parameters and makes PLSA a solid statistical model (Lee, Baker, et al. 2010, pp. 3-5; Alsumait et al. 2010, pp. 185-186). In addition, PLSA finally addresses the problem of polysemy: words in a topic can simultaneously appear in other topics (Hofmann 1999, p. 50; Lee, Baker, et al. 2010, p. 4). However, the model has one major drawback: there is no generative model at the document level (Blei, Ng, and Jordan 2003, pp. 994-995). This leads to two problems. First, it remains unclear as to how to incorporate new, unseen documents into the model. Second, the number of parameters to be estimated grows linearly with the size of the corpus, resulting in

problems with overfitting. LDA overcomes the shortcomings of PLSA by introducing Dirichlet priors (α and β) over the per-document topic distributions (θ_d) and per-topic word distributions (ϕ_k), making it a *full* generative probabilistic model.

LDA is (Not) a Mixture Model. It is important to distinguish between mixture models and LDA. According to Blei (2009), mixture models only assign *one topic* to a document and all the words in that document are dependent on that topic. This is different in the LDA model, where each document comprises a *mixture of topics*, and the words in that document can belong to different topics. In text analysis, the mixture model has proven to work well, especially for very short texts such as tweets (Mazarura and Waal 2016). For longer texts, which usually cover more diverse topics, the assumptions of LDA are more appropriate. Interestingly, the mixture model can be considered as a special case of the LDA model (Galyardt 2015, p. 45; Blei 2009). This is when very small hyper-parameters (α) close to zero are selected for the draws of the topic distributions (θ_d) from a Dirichlet. Such small values lead to extremely sparse topic distributions, meaning that a document is represented by *one* topic with very high probability, close to 1, and by all other topics with very low probabilities, close to 0. The LDA model is then almost equivalent to a mixture model.

4.3. Model Description

In this chapter the technical details of the LDA model are described. This can be done in three ways: graphically (4.3.1), linguistically (4.3.2), and mathematically (4.3.3).

4.3.1. Graphical Representation

The LDA model is most easily depicted by the "Directed Acyclic Graph" (DAG) in Figure 2 (Blei, Ng, and Jordan 2003, p. 1006; Blei and Lafferty 2009, p. 74; Blei 2012, p. 81), which is a standard way to visualize complex hierarchical probability models

(Rossi, Allenby, and McCulloch 2006, pp. 67-70). The *nodes* represent the variables of the model, where unknown variables are unshaded and observed variables are dark gray. The light gray *squares* constitute the hyper-parameters[2] for θ_d and ϕ_k. *Arrows* show the conditional dependencies between the variables. The *rectangles* are the "plates" of the model and indicate replication for the enclosed variable(s). This notation is therefore often referred to as "plate notation" (Rossi, Allenby, and McCulloch 2006, p. 67).

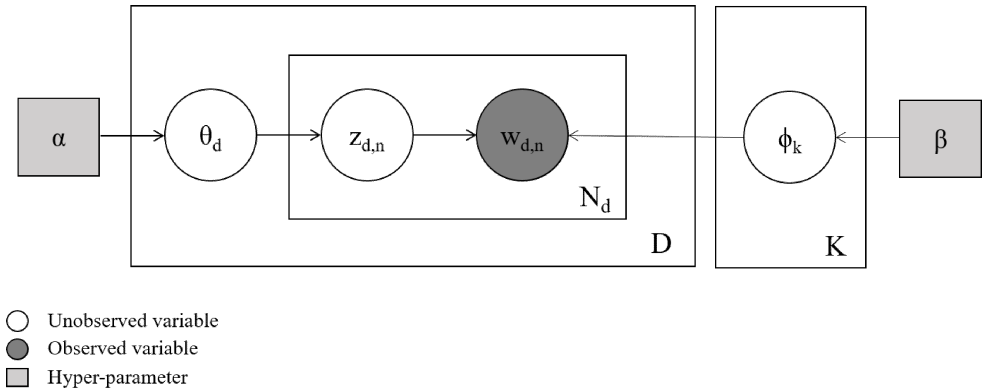


Figure 2: Directed Acyclic Graph of LDA[3]

All variables and symbols used in LDA are detailed in Table 2. The table not only contains the variables of the DAG but also those variables used later in this chapter. It is intended as a reference table for the entire chapter.

[2]In the literature, the term "hyper-parameter" is interchangeable with "concentration parameter," "pseudo count," "hyper-prior," or only "prior" (e.g., see Wallach, Mimno, and McCallum 2009).

[3]In the original paper by Blei, Ng, and Jordan (2003), the authors formulate two different models: *unsmoothed* LDA and *smoothed* LDA. In the *unsmoothed* LDA, no hyper-parameter is placed on the per-topic word distribution ϕ_k. Thus, ϕ_k is not part of the inference process but needs to be estimated using maximum likelihood (Geigle 2016, p. 4). Today, the *smoothed* LDA, as described in Figure 2, is the most common, as it has the advantage of a full Bayesian inference process, i.e., Dirchlet priors are set on the per-document topic distribution θ_d and the per-topic word distribution ϕ_k.

Symbols	Description
$w_{d,n}$	n^{th} word in document d
$z_{d,n}$	topic assignment of the n^{th} word in document d
θ_d	per-document topic distribution
ϕ_k	per-topic word distribution
α	Dirichlet prior for θ_d (hyper-parameter)
β	Dirichlet prior for ϕ_k (hyper-parameter)
D	number of documents in the corpus
d	document index
N_d	number of words in document d
n	word index on document level
K	number of topics
k	topic index
V	number of unique words in the corpus (=vocabulary)
v	unique word index
W	number of words in the corpus
w_d	vector of all words in document d
w_i	i^{th} word in the corpus
i	word index on corpus level
z_i	topic assignment of the i^{th} word in the corpus
z_{-i}	topic assignments of all words *except* for z_i

Table 2: Notation Used for LDA

4.3.2. Generative Process

Another way to describe the LDA method is by its generative process, which is the "imaginary random process" (Blei 2012, p. 78; Blei and Lafferty 2009, p. 73) by which the model assumes that the observed data ($w_{d,n}$) was produced. It is defined as follows:

1. For each topic k:
   a) Draw a distribution over words $\phi_k \sim Dirichlet(\beta)$.
2. For each document d:
   a) Draw a vector of topic proportions $\theta_d \sim Dirichlet(\alpha)$.
   b) For each word:
      i. Draw a topic assignment $z_{d,n} \sim Multinomial(\theta_d), z_{d,n} \in \{1, ..., K\}$.

ii. Draw a word $w_{d,n} \sim Multinomial(\phi_{z_{d,n}}), w_{d,n} \in \{1, ..., V\}$.

In the *first step*, the variable ϕ_k of the *topic plate* is generated. This is achieved by sampling from a Dirichlet distribution with prior β. The process is repeated for *every* topic available. The *second step* generates the variables of the *document* and *word plate*. The topic distribution θ_d is drawn from a Dirichlet distribution with prior α for a document. Subsequently, for each available word in that document, the topic assignment $z_{d,n}$ and the word $w_{d,n}$ are drawn from a Multinomial distribution using θ_d respectively $\phi_{z_{d,n}}$. Thereby, the topic assignments can take values in the range of 1 to K and the words values in the range of 1 to V. This entire second step is repeated for *every* document in the corpus. An implementation of the generative process in R is displayed in Appendix A. 1.

4.3.3. Joint Distribution

The LDA method can also be mathematically described by the joint distribution of all observed and hidden variables, given the hyper-parameters (Blei 2012, pp. 79-80; Heinrich 2008, p. 17; Carpenter 2010, p. 2; Geigle 2016, p. 5; Darling 2011, p. 2):

$$p(\phi, \theta, z, w|\alpha, \beta) \tag{1}$$

Based on the conditional dependencies specified in the DAG and generative process, the joint distribution can be factored out as follows:

$$= p(\phi|\beta) \cdot p(\theta|\alpha) \cdot p(z|\theta) \cdot p(w|\phi, z) \tag{2}$$

Taking into account the plates of the model, this yields the expression:

$$= \underbrace{\prod_{k=1}^{K} p(\phi_k|\beta)}_{\substack{topic \\ plate}} \cdot \underbrace{\left[\prod_{d=1}^{D} p(\theta_d|\alpha) \cdot \underbrace{\left(\prod_{n=1}^{N_d} p(z_{d,n}|\theta_d) \cdot p(w_{d,n}|\phi_{1:K}, z_{d,n}) \right)}_{\substack{word \\ plate}} \right]}_{\substack{document \\ plate}} \tag{3}$$

In the following, the four single factors of the joint distribution are explained in greater detail, where $p(\phi|\beta)$ and $p(\theta|\alpha)$ are Dirichlet distributions and $p(z|\theta)$ and $p(w|\phi, z)$ Multinomial distributions (Wang 2008, pp. 6-7; Heinrich 2008, p. 21).

Considering the definition of the Dirichlet distribution (compare Appendix A. 3 and Wang 2008, p. 6-7), the first two factors can be expressed as:

$$p(\phi|\beta) = \prod_{k=1}^{K} p(\phi_k|\beta) = \prod_{k=1}^{K} \frac{1}{B(\beta)} \prod_{v=1}^{V} \phi_{k,v}^{\beta_v - 1} \tag{4}$$

$$p(\theta|\alpha) = \prod_{d=1}^{D} p(\theta_d|\alpha) = \prod_{d=1}^{D} \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1} \tag{5}$$

$B(\beta)$ and $B(\alpha)$ are the normalizing constants of the Dirichlet distribution. β_v and α_k constitute the hyper-parameters for each unique word v respectively topic k. $\phi_{k,v}$ is the probability of word v appearing in topic k and $\theta_{d,k}$ is the probability of topic k being present in document d.

Using the Multinomial distribution (see Appendix A. 3 and Wang 2008, p. 6-7), the other two factors become:

$$p(z|\theta) = \prod_{i=1}^{W} \theta_{d_i, z_i} = \prod_{d=1}^{D} \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}} \tag{6}$$

$$n_{d,k}= \sum_{i=1}^{W} I\{d_i = d \wedge z_i = k\}$$

$$p(w|\phi, z) = \prod_{i=1}^{W} \phi_{z_i,w_i} = \prod_{k=1}^{K} \prod_{v=1}^{V} \phi_{k,v}^{n_{k,v}} \tag{7}$$

$$n_{k,v}= \sum_{i=1}^{W} I\{w_i = v \wedge z_i = k\}$$

$n_{d,k}$ counts the number of times topic k was assigned to words in document d (generally speaking, it counts the number of times topic k appears in document d) and $n_{k,v}$ denotes the number of times word v was associated with topic k. All counts of $n_{d,k}$ are collected in a DxK matrix respectively all counts of $n_{k,v}$ in a KxV matrix. The resulting matrices are denoted as the N^{DK} and N^{KV} matrix.

4.4. Posterior Inference

The goal of LDA is to compute the posterior distribution, which is defined as the conditional distribution of the hidden variables (ϕ, θ, z), given the observed data (w) and the hyper-parameters (α and β) (Blei 2012, pp. 79-81; Darling 2011, p. 3; Geigle 2016, p. 6):

$$p(\phi, \theta, z|w, \alpha, \beta) = \frac{p(\phi, \theta, z, w|\alpha, \beta)}{p(w|\alpha, \beta)} \tag{8}$$

A key problem in LDA is that this posterior distribution is intractable to compute (Blei, Ng, and Jordan 2003, p. 1003; Blei 2012, p. 81). The numerator is the joint distribution of all variables and can be computed easily. The marginal probability in the denominator is problematic, however. It is the probability of observing all the data in the corpus. A closer look at the form of the denominator shows the difficulty of calculation (Geigle

2016, p. 6; Heinrich 2008, p. 17):

$$
\begin{aligned}
p(w|\alpha,\beta) &= \int_{\phi}\int_{\theta}\sum_{z} p(\phi,\theta,z,w|\alpha,\beta)\ d\theta\ d\phi \\
&= \int_{\phi} p(\phi|\beta)\cdot\int_{\theta} p(\theta|\alpha)\cdot\sum_{z} p(z|\theta)\cdot p(w|\phi,z)\ d\theta\ d\phi
\end{aligned} \tag{9}
$$

The problem of computation lies in the summation over the latent topic assignments (Blei 2012, p. 81; Geigle 2016, p. 6). In theory, the sum would be over all possible ways of assigning each observed word in the corpus to one of the topics. This, however, involves an exponentially large number of combinations that make this sum and, thus, the denominator analytically intractable to compute.

Fortunately, there are a number of approximation methods available that can be used to approximate the posterior distribution. For LDA, two different approximation methods can be distinguished: sampling-based algorithms and variational inference (Blei 2012, p. 81). A very common algorithm is Gibbs sampling, which is a sampling-based algorithm. This sampling method is used in the empirical part of this dissertation and is therefore explained in greater detail in the following chapter. Variational inference was applied in the original LDA paper of Blei, Ng, and Jordan (2003, pp. 1003-1005). While variational inference is often favored because of its faster computational speed, the Gibbs sampler is generally known to be more accurate.

4.5. Gibbs Sampling

Gibbs sampling was first introduced in 1984 in a paper by Geman and Geman (1984) in the context of image processing. The algorithm belongs to the Markov Chain Monte Carlo (MCMC) methods and is used for difficult calculations, which are instead replaced by a sequence of easier calculations (Casella and George 1992, p. 167). The Gibbs sampler approximates the posterior distribution by iteratively sampling from the full

conditional distributions ("full conditionals"), which are defined as the distribution of a parameter conditional on all other known and unknown parameters (Walsh 2002, pp. 16-17; Gilks and Wild 1992, pp. 342-343; Gelfand and Smith 1990, pp. 400-401; Casella and George 1992). Those full conditionals have a much easier form than the posterior distribution and facilitate computation. Sampling needs to be conducted for many iterations until the stationary state of the Markov chain is reached. This is when the sampled values approximate (or "converge to") the desired posterior distribution. Initial draws of the Gibbs sampler need to be discarded, as they are poor estimates. Those draws are referred to as the "burn-in period."

In LDA the *collapsed Gibbs sampler* is most commonly used because of its high efficiency. In this sampler the parameters ϕ and θ are integrated out ("collapsed") which leads to a faster convergence of the model. The complete derivation of the collapsed Gibbs sampler is explained in detail in Appendix A. 2. At this point in the text only a short summary of the sampling scheme is presented.

Steyvers and Griffiths (2007) and Griffiths and Steyvers (2004) summarize the entire sampling process for the collapsed Gibbs sampler in four major steps:

1. **Initialization:** First, z_i needs to be initialized by assigning each word i to a random topic between 1 and K. This constitutes the initial state of the Markov chain. Based on this information, the count matrices N^{KV} and N^{DK} can be created, which are required for the next step.

2. **Gibbs Sampling:** The Gibbs sampler is now run for a number of iterations. One iteration involves a pass over all words across all documents:

   For *each* word i:

   a) Decrement the count matrices N^{KV} and N^{DK} by 1 for the entries of the *current*

topic assignment z_i.

b) Sample a new topic from the full conditional:[4]

$$p(z_i|z_{-i}, w, \alpha, \beta) \propto \frac{n_{k,v}^{(-i)} + \beta_v}{\sum_{v=1}^{V}(n_{k,v} + \beta_v) - 1} \cdot \frac{n_{d,k}^{(-i)} + \alpha_k}{\sum_{k=1}^{K}(n_{d,k} + \alpha_k) - 1}$$

c) Increment the count matrices N^{KV} and N^{DK} by 1 for the entries of the *new* topic assignment.

This sampling procedure is repeated for many iterations until the Gibbs sampler starts to approximate the target distribution.

3. **Burn-in:** The burn-in period refers to the initial draws of the Gibbs sampler, which are poor estimates of the posterior distribution and therefore need to be eliminated from the results. After this burn-in period, the samples start to converge around the target distribution and can be used to get ϕ and θ.

4. **Obtaining ϕ and θ:** Parameters ϕ and θ can finally be obtained by a Dirichlet draw, as follows:[5]

$$Dir(\phi_k|n_k + \beta) \; for \; each \; topic \; k = 1, \; ..., \; K$$

$$Dir(\theta_d|n_d + \alpha) \; for \; each \; document \; d = 1, \; ..., \; D$$

[4]Compare Equation (26) in Appendix A. 2; $n_{k,v}^{(-i)}$ and $n_{d,k}^{(-i)}$ are counts that do not include the current topic assignment z_i.

[5]Compare Equation (27) and Equation (28) in Appendix A. 2.

Alternatively, the formulas for the posterior means can be used:[6]

$$\phi_{k,v} = \frac{n_{k,v} + \beta_v}{\sum_{v=1}^{V}(n_{k,v} + \beta_v)}$$

$$\theta_{d,k} = \frac{n_{d,k} + \alpha_k}{\sum_{k=1}^{K}(n_{d,k} + \alpha_k)}$$

According to Heinrich (2008, p. 23), there are three ways to receive the model parameters ϕ and θ after the burn-in period: (1) reading out only a single sample, (2) averaging a number of samples, and (3) leaving an interval of I iterations ("thinning interval" or "sampling lag") between subsequent read-outs and averaging across those.

4.6. Evaluation

A vital step in topic modeling is the evaluation of the model results. According to Reisenbichler and Reutterer (2019), scholars perform evaluations for various reasons; these are described below:

- **Analysis of Model Fit**

  One major objective in topic modeling is to evaluate the predictive performance of the model, i.e., how well a model performs on new, unseen data. Therefore, the data is split into training data and test data (also known as "held-out data" or "non-training data"), where the training data is used for learning the model and the test data is used for evaluating the model. The data can be divided either on the corpus level, i.e., taking entire documents for the test data, or on the document level, i.e., separating a document into two parts where the first half goes into the training data and the second half into the test data. The latter procedure is

[6] Compare Equation (29) and Equation (30) in Appendix A. 2.

referred to as "document completion" (Wallach, Murray, et al. 2009, p. 1109).

A widely used metric for evaluating topic models is *perplexity*; see, for example, Blei, Ng, and Jordan (2003, p. 1008) or Grün and Hornik (2011, p. 7):

$$perplexity(w) = exp\left(- \frac{log\ p(w)}{\sum_{d=1}^{D} \sum_{v=1}^{V} n_{v,d}} \right) \tag{10}$$

The numerator is the *log-likelihood* and the denominator the *total number of tokens* in the data, where $n_{v,d}$ counts how often word v occurs in document d. The lower the perplexity score, the better the model. In the above equation, the training data is considered for the in-sample fit, and the test data is considered for the out-of-sample or predictive fit. The log-likelihood is calculated using Grün and Hornik (2011, p. 7):

$$log\ p(w) = \sum_{d=1}^{D} \sum_{v=1}^{V} n_{v,d} \cdot log \left[\sum_{k=1}^{K} \theta_{d,k} \phi_{k,v} \right] \tag{11}$$

Instead of computing the perplexity, as in Equation (10), some scholars only use the log-likelihood in Equation (11) for their model fit, such as Büschken and Allenby (2016, pp. 961-962) or Hruschka (2014, pp. 10-12). The interpretation of the log-likelihood is opposite to the perplexity. The higher the log-likelihood, the better the generalization performance of the model to new data. Other possible evaluation metrics are summarized in Wallach, Murray, et al. (2009), which includes importance sampling, harmonic mean and annealed importance sampling, Chib-style estimation, and a left-to-right evaluation algorithm.

- **Analysis of Convergence**

  Checking whether a topic model converges is pivotal for receiving reliable estimates. A standard way to determine the required number of iterations for convergence is

to calculate the perplexity, as defined in Equation (10), for every iteration.[7] For instance, Chen and Wang (2014, pp. 5-11), Sun et al. (2013, p. 72), and Asuncion et al. (2009, pp. 31-33) follow this procedure in their analyses. In plotting the perplexity against the iterations, convergence can be declared when the perplexity decreases and then levels off.

- **Analysis of Optimal Parameter Settings**

  Numerous studies discuss two parameter settings: *selection of hyper-parameters* and *selection of number of topics*. The selection of the hyper-parameters and their influence on the model results are discussed in detail in Appendix A. 3 and is therefore not further explained here. Griffiths and Steyvers (2004, p. 5231) and Steyvers and Griffiths (2007, p. 441) explain how to select the optimal number of topics. A widely used approach is to run the topic model for different topic numbers and calculate the perplexity, as defined in Equation (10). The topic number that achieves the best perplexity value is then selected as a result and examined in closer detail.

- **Analysis of Clustering Output**

  Substantial research has been conducted on evaluating the clustering output. A major focus lies on assessing *topic coherence*, i.e., the quality of a topic with respect to its interpretability. Chang et al. (2009) presented an evaluation method with human judgement, called a "word intrusion task." In this task, a person is presented six randomly ordered words and given the task of finding the word that is out of place or does not belong to the others (i.e., the "intruder"). The set of words displayed to a user is constructed by: (1) selecting a topic from the model; (2) choosing the five most likely terms from that topic; and (3) adding an intruder

[7] Calculating the perplexity for *every* iteration can be very time-consuming. To speed up calculation, very frequently only *every* i^{th} iteration (e.g., *every* 10^{th} or 100^{th} iteration) is used. Standard packages or softwares for topic models usually have an option to indicate the iteration interval at which evaluation should be conducted.

word from a pool of words with low probability in the current topic. Even though it is useful, this method can be very time-consuming. Subsequent works, such as Newman, Lau, et al. (2010), Newman, Baldwin, et al. (2010), and Mimno et al. 2011, have therefore developed metrics that automatically measure topic coherence with near-human accuracy. Besides topic coherence, there exists a plethora of other topic diagnostic metrics, e.g., topic size, word length or similarity measures, that assess the model output. Boyd-Graber, Mimno, and Newman (2015, pp. 238-241) provided a good overview of the most important metrics.

- **Analysis of Computational Performance**

  Due to the combination of big data and complex models in the topic modeling context, computation constitutes a major challenge. A growing number of papers therefore have compared different estimation methods with respect to computational time or have developed new estimation methods that are faster (e.g., Ansari, Li, and Zhang 2018; Ishigaki et al. 2015).

Part III.

Empirical Analysis

5. Data

This chapter takes a closer look at the data that is used in the empirical analysis. It starts with discussing some descriptive statistics and special characteristics of the data (5.1). Subsequently, the selected data preparation (5.2) is explained in detail, including why a new pre-processing approach is necessary for making new product recommendations. Finally, the division of the data into training and test data (5.3) is described. The chapter is accompanied by Appendix B. 1 which provides some additional data statistics.

5.1. Data Description

For the empirical analysis, purchase data from an online retailer of animal health products was used. The product assortment ranged from special animal food and veterinary medicine to useful animal accessories for pets, such as cats and dogs, as well as larger animals, such as horses. The complete order history from January 2012 until January 2016 was obtained. Only transactions with a status of "complete" were kept. Those were transactions for which the delivery of the product and payment were fully completed and the product's return window had elapsed. "Canceled," "pending," or transactions in any other status were removed from the data set. Table 3 shows the summary statistics of the data set.

	Total	**Median**	**Mean**	**SD**	**Min**	**Max**
Number of orders	135,456	-	-	-	-	-
Number of customers	96,239	-	-	-	-	-
Number of SKUs (total)	264,717	-	-	-	-	-
Number of SKUs (unique)	4,758	-	-	-	-	-
Number of orders per customer	-	1	1.41	2.63	1	603
Number of SKUs per order	-	1	1.95	3.40	1	148
Number of SKUs per customer	-	1	2.75	10.82	1	1,347
Number of orders per SKU	-	4	33.37	206.53	1	6,140

Table 3: Descriptive Statistics of the Data Set

The data set comprised of a total of 135,456 orders, which were placed by 96,239 customers. Across these orders, a total of 264,717 items were dispatched. The number of unique items equaled 4,758. The data set illustrated the typical purchasing behavior of online shoppers; most customers placed very few orders with a small number of items. As reported in the lower part of the table, the mean number of orders per customer was 1.41, with 1.95 items per order on average. When the orders were aggregated across a customer, the average number of items became 2.75. A median of 1 for the number of Stock Keeping Units (SKUs) per customer further shows that the majority of the customers in the data set only placed one order with one item (*single-item customers*); that is, for many customers there was very little information available. The large range for the number of SKUs per customer, from 1 to 1,347, is also noteworthy. This implies that information on co-occurrence of items is not evenly distributed across the customer base. The last value in the table, the number of orders per SKU, indicates the number of orders in which an item appeared. The distribution of this value exhibits a very long tail. Most of the items were present in four or fewer orders. Some SKUs, however, also appeared in many orders, up to a maximum of 6,140 orders.

Additional analyses revealed that customers frequently bought the same item multiple times within an order or across orders. Such items are referred to as *repeat items* in this work. The data comprised 40% repeat items when repeats were calculated at the order level. When repeats were considered across a customer's orders, they comprised 53% of the data. Figure 3 shows these percentages.

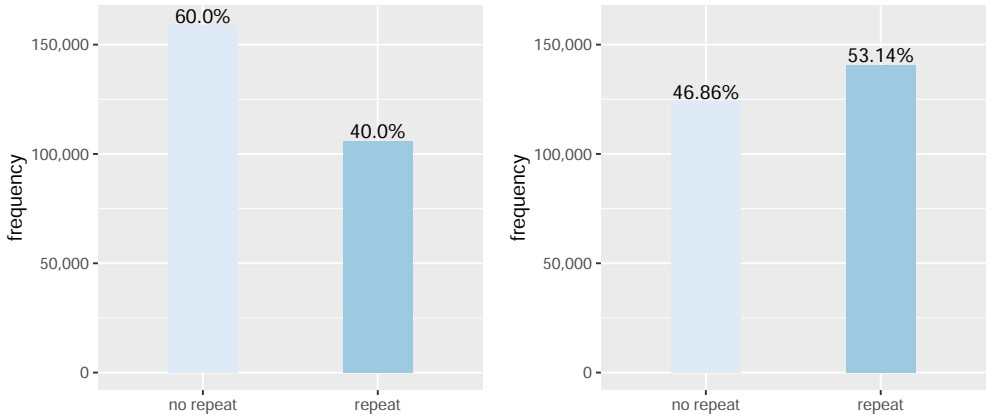


Figure 3: Percentage of Repeat Items on Order (Left) and Customer (Right) Level

5.2. Data Preparation

There is disagreement in the literature about which data preparation should be used for recommender systems. A comparison of the data pre-processing of several authors shows very different approaches. Some authors removed repeat items (Jacobs, Donkers, and Fok 2016, p. 397), while others left them in (Ishigaki et al. 2015, p. 13). A few papers eliminated very frequent or rare items (Hruschka 2014, p. 269; Schröder et al. 2017, p. 41). In some analyses, researchers did not work with single items, but combined items of the same category and brand into a "category-brand combination" (Jacobs, Donkers, and Fok 2016, p. 397; Hruschka 2014, p. 269; Hruschka 2016, pp. 3, 7; Schröder 2017, p. 30; Christidis, Apostolou, and Mentzas 2010, p. 7). Other authors deleted customers with only a few purchased items (Iwata and Sawada 2013, p. 566; Sun et al. 2013, p. 72).

The selected data preparation method, however, can have substantial effects on the success of a recommender algorithm. Herlocker et al. (2004) noted this and stated that the data should be selected in such a way that a recommender system can fulfill the task for which it is being evaluated as well as possible. The goal of this dissertation is to evaluate

topic models in terms of how good they are at generating *new* product recommendations for customers. For this reason, only data that contributes to novelty of recommendations should be considered for analysis.

Two data characteristics require a closer examination when it comes to new product recommendations: First, the data contained many *repeat items*, i.e., the same item is purchased multiple times within an order or across orders of a customer. The problem with repeat items is that they do not inform a recommender system about new items and at the same time prevent product recommendations from the long tail. Such long tail items, however, are critical for businesses as they are less likely to be discovered by customers and often indicate new, emerging consumer preferences. Second, more than half of the customers bought only one single item in their entire purchase history. Such *single-item customers* are also not very informative. They do not provide information about item interaction and are therefore of little use in identifying suitable new items for cross-selling.

To investigate whether such *repeat items* and *single-item customers* should be eliminated for new product recommendations, six different approaches to pre-processing were considered prior to running and comparing the models in this dissertation. Repeat items were either left in the data, removed at the order level, or removed at the customer level. Single-item customers were also either left in the data or removed from the data. Table 4 reports the data preparation variants as well as their abbreviations, which are used throughout this work. Note that ALL is the variant without pre-processing for which descriptive statistics are shown in Table 3. The extent of data preparation increases when moving down in the columns and from left to right. 11USC constitutes the most selective variant of data preparation.

	Customers			
	With Single-Item Customers		**Without Single-Item Customers**	
Items	**ALL**	All SKUs	**11ALL**	All SKUs
	USO	Unique SKUs at the order level	**11USO**	Unique SKUs at the order level
	USC	Unique SKUs at the customer level	**11USC**	Unique SKUs at the customer level

Table 4: Six Variants of Data Preparation[8]

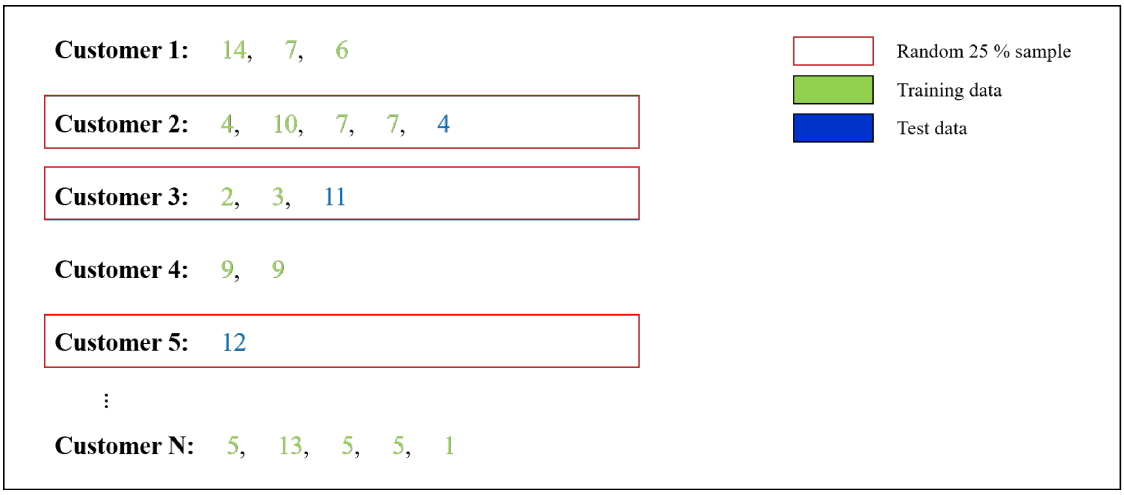
This dissertation is closest to the study of Jacobs, Donkers, and Fok (2016) but sets a different focus by making *recommendations* instead of predictions. The pre-processing used by Jacobs, Donkers, and Fok (2016) is discussed below as well as why a different data preparation method is necessary for a recommendation task. In their paper, the researchers considered "unique category-brand combinations" at the order level. This refers to a customer buying multiple items from the same category-brand combination within an order and the multiple counts of those items are ignored; that is, they are set to 1. This involves two data preparation steps: (1) assigning products of the same category-brand, such as different fragrances of the same deodorant brand, to one category-brand combination; and (2) summarizing the same category-brand combinations within an order to a single purchase. This pre-processing procedure, however, has some shortcomings in a recommender context. In the first data preparation step, Jacobs, Donkers, and Fok (2016) ignore different variants of products even though they might address different customer needs. Furthermore, with regard to actual item recommendations in the online shop, it is not clear which specific product should be recommended. This needs to be left to a random selection of one of the products in the category-brand combination. In the second data preparation step, it is assumed that buying multiple items from one category-brand combination *within* an order provides no information with respect to predicting future purchases. However, buying multiple items from one category-brand

[8]The "11" in the abbreviations stands for the fact that single-item customers were eliminated from the data; these are customers with only "1" order and "1" purchased SKU.

combination *across* orders is assumed to do so. This is counterintuitive for recommending new products. For a recommender task, the number of purchases of an item should also be irrelevant across orders when it is within orders.

5.3. Splitting Data into Training and Test Data

The data was split into training and test data. The training data was used for learning the models and the test data for evaluating the models. For the division of the data, a random 25% sample of all customers was drawn. These randomly drawn customers are referred to as *holdout customers*. For each of the holdout customers, the last SKU in the purchase history was pulled out and used as test data. This data division mimics the situation in an online shop where a company wants to predict the *next* item a customer is going to buy, given his past purchase history. Figure 4 illustrates this data splitting.

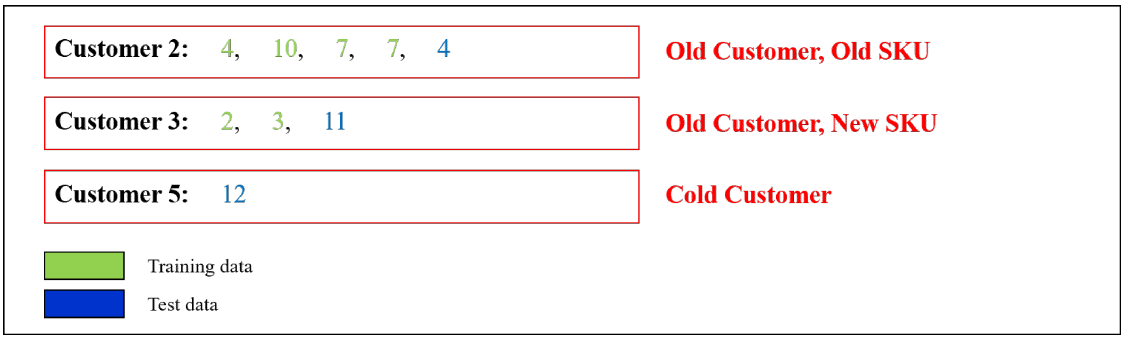


The figure shows customers 1 to N in the data and their purchased products, which are sorted by the time ordered. The customers in red boxes belong to the random 25% sample and comprise the holdout customers. From those holdout customers, the last item was pulled out for the test data. All other SKUs, including those not in the 25% sample, went into the training data. Model evaluation is later based on the correct prediction of the holdout items in the test data.

Figure 4: Example of Data Splitting

A closer look at the holdout customers reveals three different subgroups. There are holdout customers for which a minimum of one item is observed in the training data.

When an item is predicted for these customers, it can either be a non-novel product the customer has already purchased (i.e., it is present in the training data) or a novel product this customer has *not* purchased yet (i.e., it is not present in the training data). The first subgroup is called *Old Customer, Old SKU*, and the second subgroup is called *Old Customer, New SKU*. If a holdout customer has only ordered one single item, this item goes into the test data (i.e., no purchases are observed for this customer in the training data). This subgroup is called *Cold Customer*. Examples and definitions of each subgroup are provided in Figure 5.



The subgroups are defined as follows: *Old Customer, Old SKU* are holdout customers with a minimum of one item in the training data and for which an item is predicted that was purchased by this customer before. *Old Customer, New SKU* are holdout customers with a minimum of one item in the training data and for which an item is predicted that was *not* purchased by this customer before. *Cold Customer* are holdout customers for which no items are observed in the training data.

Figure 5: Subgroups of Holdout Customers

This distinction of holdout customers is essential for revealing differences in model performance with respect to these subgroups. Most important for evaluating a model's ability to recommend *new* items is the subgroup *Old Customer, New SKU*. It is the focus in this work.

6. Methods

This chapter introduces the recommender models that are compared in this dissertation. The first section elaborates on the usage of topic models (6.1) as recommender models. It starts by explaining how topic models can be applied to online purchase data and which notation and model settings are used. Afterward, the two topic models, ATM and Sticky ATM, are presented along with a discussion of how recommendations are generated with them. In the second section, the benchmark models (6.2) Unigram, Bigram, and CF are described. The chapter closes with describing the performance measure (6.3) used to evaluate the various recommender models.

6.1. Topic Models

6.1.1. Application to Online Purchase Data

The use of topic models for text analysis or recommendation activities stems from the same idea: In text analysis, topic models aim to detect frequently co-occurring words that relate to latent topics. In a recommender context, topic models aim to detect frequently co-occurring products that relate to latent customer interests. With this analogy, a mapping of topic models from text data to online purchase data is possible. This is demonstrated in Table 5:

Text Data	**Online Purchase Data**	
	ATM	**Sticky ATM**
Document	Customer Purchase History	Customer Purchase History
Word	Product	Product
Topic	Topic (= Customer Interest)	Topic (= Customer Interest)

Table 5: Application of Topic Models to Online Purchase Data

In both topic models, a customer's purchase history is treated as a *document*[9] and each

[9]Note that by defining a *document* as a customer purchase history, a customer's single orders must be pooled into one document. The identification of customers across different orders was possible through their email address, which is required for each order in the online shop.

purchased product as a *word*. A *topic* represents a collection of items that are likely to be bought by a particular set of customers to whom they are of interest. For the sake of simplicity, the term *topic* is further used in the context of online purchase data. Other authors have changed the term to *motivation* (Jacobs, Donkers, and Fok 2016), *latent activity* (Hruschka 2014), *latent trait* (Schröder 2017), or *customer's purchasing preference* (Sun et al. 2013).

6.1.2. Notation and Model Settings

The notation used for topic models in the empirical part of this dissertation is summarized in Table 6 and compared to the standard notation commonly seen for text data. The parameter $x_{c,n}$ is observed; it stands for the n^{th} product of customer c. The parameter $z_{c,n}$ is unobserved; it stands for the topic assignment of the n^{th} product of customer c. In a recommender context, topic models characterize each topic k by a distribution over products (ϕ_k) and determine an individual topic distribution (θ_c) for each customer c. The parameters α and β comprise the Dirichlet priors for ϕ_k and θ_c. They are fixed and specified prior to running the model. The number of customers, the number of products per customer, the number of topics, and the number of unique products are indicated by the uppercase letters C, N_c, K, and P. The last three parameters ϵ, ψ_k and ζ_n are only necessary in the Sticky ATM for modeling stickiness.

Both topic models were run for multiple topic numbers from $K = 5-100$. For $K = 5-30$, each single topic number was considered, and for $K = 30-100$, every tenth topic number was considered. Each model was run for 500 iterations. The priors were set to $\alpha = 1/\mathrm{K}$ and $\beta = 200/\mathrm{P}$.

Symbols	Description for *Purchase Data*	Symbols	Description for *Text Data*
$x_{c,n}$	n^{th} product of customer$c$	$w_{d,n}$	n^{th} word in document d
$z_{c,n}$	topic assign. of n^{th} product of customer c	$z_{d,n}$	topic assign. of n^{th} word in document d
θ_c	per-customer topic distribution	θ_d	per-document topic distribution
ϕ_k	per-topic product distribution	ϕ_k	per-topic word distribution
α	Dirichlet prior for θ_c (hyper-parameter)	α	Dirichlet prior for θ_d (hyper-parameter)
β	Dirichlet prior for ϕ_k (hyper-parameter)	β	Dirichlet prior for ϕ_k (hyper-parameter)
C	number of customers (in the data)	D	number of documents (in the corpus)
N_c	number of products per customer c	N_d	number of words in document d
K	number of topics	K	number of topics
P	no. of unique products (product assortment)	V	no. of unique words (vocabulary)
ϵ $(epsilon)$	Beta prior for ψ_k (hyper-parameter)	ϵ $(epsilon)$	Beta prior for ψ_k (hyper-parameter)
ψ_k (psi)	probability of stickiness for topic k	ψ_k (psi)	probability of stickiness for topic k
ζ_n $(zeta)$	stickiness indicator of n^{th} product of customer c	ζ_n $(zeta)$	stickiness indicator of n^{th} word in document d

Table 6: Notation for Topic Models with Purchase Data vs. Text Data

6.1.3. ATM

The Author Topic Model (ATM) extends the basic LDA model of Blei, Ng, and Jordan (2003) by including authorship information. This is achieved by pooling all orders of a customer into one document. The word *author* in the ATM model thus refers to a *customer* in the context of recommender systems. The output of the ATM differs from the LDA in the topic distribution θ, which is obtained for each customer and not only for each individual order. This allows for more customized recommendations. The generative process of the ATM is defined as:

1. For each topic k:
   a) Draw a distribution over products $\phi_k \sim Dirichlet(\beta)$.
2. For each customer c:
   a) Draw a vector of topic proportions $\theta_c \sim Dirichlet(\alpha)$.
   b) For each product:
      i. Draw a topic assignment $z_{c,n} \sim Multinomial(\theta_c), z_{c,n} \in \{1, ..., K\}$.
      ii. Draw a product $x_{c,n} \sim Multinomial(\phi_{z_{c,n}}), x_{c,n} \in \{1, ..., P\}$.

Figure 6 shows the DAG of ATM. For the estimation of the model, collapsed Gibbs sampling is applied as outlined in Chapter 4.5.

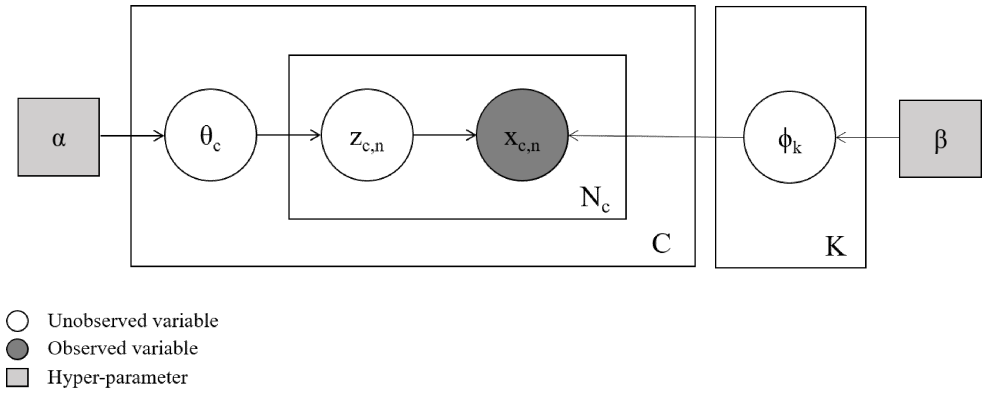


Figure 6: Directed Acyclic Graph of ATM

6.1.4. Sticky ATM

The Sticky ATM additionally accounts for stickiness at the item level (Büschken and Allenby 2016, pp. 957-958). This means that the same topic assignment can carry over for a certain sequence of items ordered by a customer. In other words, customers often stay with one topic over a number of successively purchased items before they change the topic from which they buy. Consider the following example: a customer wants to groom his dog and puts a sponge, shampoo, and a brush in the online shopping basket of an online retailer. One week later, he realizes he is almost out of dog food. He orders a bag of dry food, three cans of wet foot, a mix of dog treats, and two chewing bones. This customer clearly buys by topic. The first three items refer to the topic *dog's grooming*. The topic switches to *dog food* for the remaining items.

The Sticky ATM model accounts perfectly for this buying behavior. Stickiness is modeled by drawing a stickiness indicator ζ_n for each item in a customer's purchase history. If $\zeta = 1$, an item exhibits stickiness. That is, the topic of the previous item carries over to the next item. If $\zeta = 0$, no stickiness is observed. A new topic assignment is then drawn for this item with the customer's topic distribution θ_c. This indicates the shift of buying from an old to a new topic.

Figure 7 illustrates this process. The upper panel describes the general case, where ζ is unknown. In the middle panel, a sequence of sticky and non-sticky topic assignments are displayed. The lower panel shows a sequence of only non-sticky topic assignments (ζs are always 0). In this case, the Sticky ATM is equivalent to the ATM. ψ is the probability of stickiness for each topic k, which is drawn with prior ϵ from a beta distribution. The entire generative process for the Sticky ATM model is defined as follows (Büschken and Allenby 2016, p. 971):

1. Draw ψ_k from $Beta(\epsilon)$ $\forall$ k iid.

2. Draw ϕ_k from $Dirichlet(\beta)$ $\forall$ k iid.

3. Draw θ_c from $Dirichlet(\alpha)$ $\forall$ c iid.

4. For the first product p_1 of customer c:

    a) Draw z_1 from $Multinomial(\theta_c)$.

    b) Draw x_1 from $Multinomial(\phi_{k=z_1})$.

    c) Draw ζ_2 from $Binomial(\psi_{k=z_1})$.

5. For products x_n, $n \in \{2, ..., N_c\}$, of customer c:

    a) If $\zeta_n = 0$: draw z_n from $Multinomial(\theta_c)$; if $\zeta_n = 1$: set $z_n = z_{n-1}$.

    b) Draw x_n from $Multinomial(\phi_{k=z_n})$.

    c) Draw ζ_{n+1} from $Binomial(\psi_{k=z_n})$.

6. Repeat steps 4 and 5 for all customers c (except for draw of ζ_{N_c}).

Uncollapsed Gibbs sampling is used for posterior estimation. Detailed information on the sampling scheme can be found in Büschken and Allenby (2016, pp. 971-974).

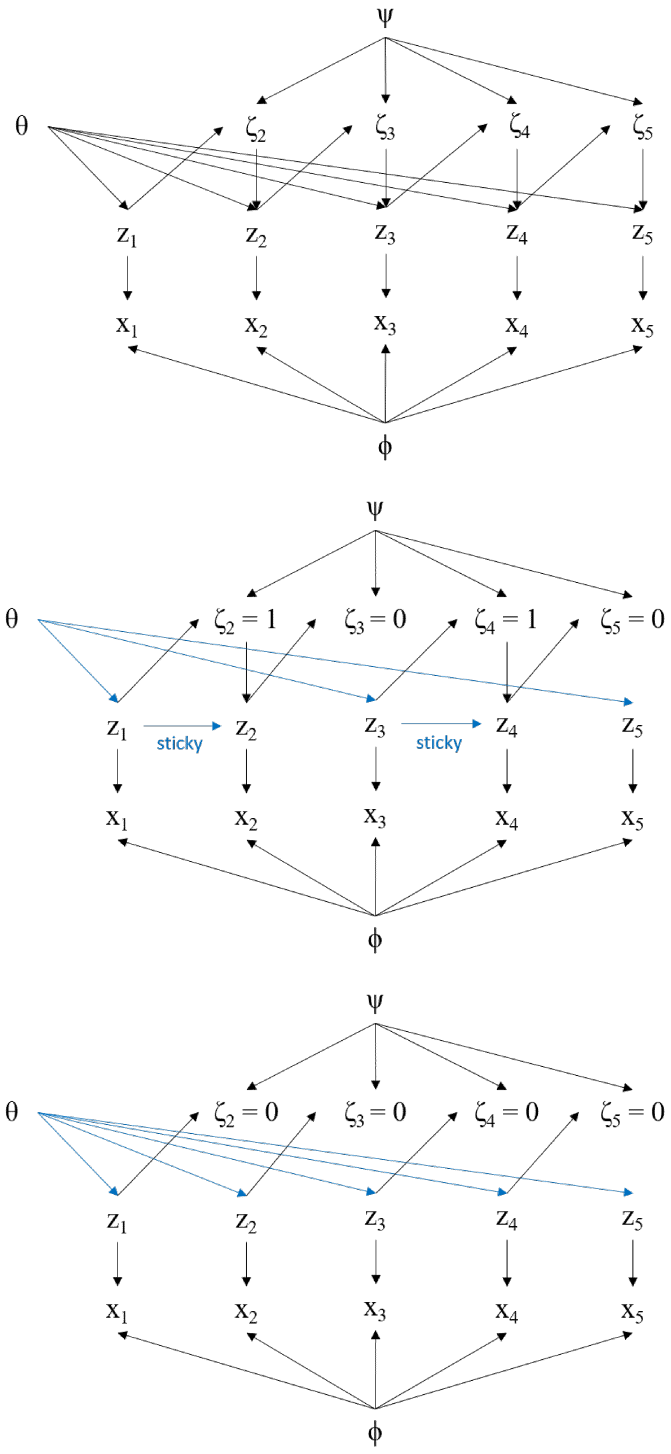


Figure 7: Directed Acyclic Graph of Sticky ATM[10]

[10]Adapted from Büschken and Allenby (2016, p. 959).

6.1.5. Recommendation Generation

In this dissertation, two different approaches for generating recommendations for each customer were considered: First, a customer received *recommendations based on the most probable items of a topic*. This was achieved by drawing a topic from a Multinomial distribution with the customer's topic distribution θ_c. The most probable items in this drawn topic were then displayed to the customer. Note that with this approach, recommendations were limited to coming from *one* topic only. The entire procedure is described as pseudocode in the following:

Algorithm 1

input: matrices PHI (P x K) and $THETA$ (K x C)
output: customer-specific recommendation list

for each customer c **do**

    get $theta\_c \leftarrow THETA[\ ,\ c]$

    draw topic from multinomial:
    $k \leftarrow which.max(rmultinom(1, 1, theta\_c))$

    get $phi\_k \leftarrow PHI[\ ,\ k]$

    order phi_k descending:
    $recomlist \leftarrow order(phi\_k, decreasing = true)$

    print $recomlist[1 : N]$

end for

Figure 8: Pseudocode for Recommendation Generation Approach 1

Second, a customer received *recommendations based on the probability* $p_{c,p}$ that a customer c would purchase product p:

$$p_{c,p} = \sum_{k=1}^{K} \phi_{k,p} \cdot \theta_{c,k} \tag{12}$$

The formula relates to the importance of an item p in topic k ($\phi_{k,p}$) multiplied by the importance of a topic k for customer c ($\theta_{c,k}$). The topics are "integrated out" through the

summation across $k = 1, \ldots, K$. As a consequence, recommendations are not restricted to one topic anymore but might include high probability items from *different* topics a customer is interested in. This procedure was also selected by Jacobs, Donkers, and Fok (2016, p. 392), Hruschka (2014, p. 266) and Schröder et al. (2017, p. 41). The pseudocode is listed below:

Algorithm 2

input: matrices PHI (P x K) and $THETA$ (K x C)
output: customer-specific recommendation list

begin

    $PROB \leftarrow PHI \cdot THETA$

end matrix multiplication

for each customer c **do**

    get $p\_c \leftarrow PROB[\ ,\ c]$

    order p_c descending:
    $recomlist \leftarrow order(p\_c, decreasing = true)$

    print $recomlist[1 : N]$

end for

Figure 9: Pseudocode for Recommendation Generation Approach 2

6.2. Benchmark Models

The topic models were compared to three benchmark models, Unigram, Bigram, and CF. Unlike topic models, these methods do not identify the latent topics a customer is interested in but calculate item probabilities based on item counts, item sequences and item combinations. Unigram comprises the simplest method and is equivalent to marginal probabilities. Bigram and CF make use of conditional probabilities. Whereas Bigram conditions on the last item in a customer's purchase history, CF conditions on similar customers.

6.2.1. Unigram

Unigram is the most straightforward method for making recommendations. Items are aggregated across all customers; subsequently, the marginal probability is calculated for each unique item as follows:

$$p(x_i) = \frac{c(x_i)}{X} \tag{13}$$

where $c(x_i)$ stands for the counts of item x_i across all customers and X equals the total number of items in the data. Recommendations are then based on the top N most probable items. With the Unigram model, each customer receives the same items as recommendations since information at the customer level is ignored.

6.2.2. Bigram

Bigram models first-order dependencies of items, i.e., it takes a look at which products are purchased one after the other. For the technical implementation of this method, the purchase histories of all customers are divided into their sequences. For instance, a customer's history with items [5, 4, 10] would be divided into the sequences (5, 4) and (4, 10). It is essential to maintain the order in which items are put into the shopping basket.

To make product recommendations, the conditional probabilities are required of the most likely (e.g., top 10) items given a customer's last observed item. These conditional probabilities are calculated as follows:

$$p(x_i|x_{i-1}) = \frac{c(x_{i-1}, x_i)}{\sum_x c(x_{i-1}, x)} \tag{14}$$

where $c(x_{i-1}, x_i)$ counts the number of sequences of item x_{i-1} preceding item x_i and is normalized by the sum of all bigrams that share the same first item x_{i-1}.

Recommendations in the Bigram model are not the same for all customers but depend on a customer's last observed choice. Thus, customers with the same last observed item (i.e., same conditioning argument) receive the same recommendations. Even though information at the customer level is considered now, this method still lacks individual item probabilities for each customer as in the topic models.

6.2.3. Collaborative Filtering

The idea of Collaborative Filtering (CF) is to compare a focal customer with similar customers, i.e., customers who displayed similar purchasing behavior, and learn which other products they purchased. Various types of implementation are possible. In this dissertation, the CF technique proposed by Jacobs, Donkers, and Fok (2016) is used. In their application, a focal customer is matched to other customers who purchased the same products and at least one additional product. All items from those matching customers are aggregated, and then the most likely products are recommended to the focal customer. Matching on the complete purchase history, however, is not always possible. Customers are therefore matched on parts of their purchase history, namely on unique items, or *singlets* (CF-1) and unique sorted pairs, or *doublets* (CF-2). For recommending products, one product more than the matching product(s) is always required. Thus, when products should be recommended by matching on singlets, counts of doublets are needed and by matching on doublets, counts of triplets are needed. A customer's history is therefore replaced by its singlets, doublets, and triplets. For instance, the purchase history [3, 3, 16, 42] has the following:

- Singlets (3), (16), and (42)
- Doublets (3, 3), (3, 16), (3, 42), and (16, 42)
- Triplets (3, 3, 16), (3, 3, 42), and (3, 16, 42).

After disassembling the purchase histories into their components, individual-specific product scores are computed for each customer. The products are ranked according to these scores and the top N products are displayed to the customer. Formally, the score of product j for customer c by matching on product combination of size k is (Jacobs, Donkers, and Fok 2016, p. 401):

$$s_{cj}^{k} = \sum_{h \in H_c^k} \frac{c(h,j)}{\sum_{p=1}^{J} c(h,p)}. \tag{15}$$

The number of customers who bought product combination h and j, $c(h,j)$, is normalized by the total sum of product combinations comprising h and any other product $p = 1, ..., J$, $\sum_{p=1}^{J} c(h,p)$. The parameter h either stands for a singlet (when $k = 1$) or a doublet (when $k = 2$). The above fraction is calculated for every product combination h of a customer c, and all single values are then summed up to the final score. H_c^k collects all product combinations h of size k of customer c. If a product combination h is never bought with an additional product across all customers, i.e., $\sum_{p=1}^{J} c(h,p) = 0$, it is regressed to a lower value of k. For cold-start customers, i.e., customers with no purchases yet, matching is not possible and the most frequent singlets are used for recommendation.

6.3. Performance Measure

To evaluate the performance of the single models, recommender sets of different sizes were built for each customer. These recommender sets contained the highest ranked items for a customer. When only one product recommendation should be made, a recommender set of size 1 was suitable. Online shops, though, often display multiple products to a customer. Here, recommender sets of size 3, 5, or 10 would be more realistic. Recommender sets were compared with the test data. If a customer's holdout item was present in the recommender set, this was considered a *hit*. The hit rate for a model M was calculated

as follows:

$$hit\ rate_M = \frac{number\ of\ hits_M}{number\ of\ holdout\ items} \tag{16}$$

where the number of hits generated by model M is divided by the total number of holdout items. In a model comparison, the model that achieves the highest hit rate is considered the best performing model. In other words, this model shows the highest number of hits relative to the total number of hits possible.

Hit rates were calculated for all holdout customers and again separately for the different subgroups stated in Chapter 5.3. When predicting items for the subgroup *Old Customer, New SKU*, only those products were included in the recommender set that were *not* bought by the corresponding customer in the past. This was achieved by eliminating the customer's previously bought items from her/his individual product ranking before the recommender sets were built.

7. Results

The results section is subdivided into two parts. First, some descriptive results of the topic models (7.1) are presented. This includes a look at the convergence, optimal topic number, and revealed item and topic distribution. Second, the different recommender methods are compared to each other (7.2) by assessing their hit rates. Results are displayed for all holdout customers and the different subgroups.

7.1. Topic Models

7.1.1. Convergence and Optimal Topic Number

It is essential to ensure that the topic models converged. Otherwise, the estimates would not be useful. Convergence for ATM and Sticky ATM was usually observed within the first 200 iterations. Every topic number was run for 500 iterations. The initial 400 iterations were discarded and the remaining 100 iterations were used for analysis.

Another central step was to determine the optimal topic number for each topic model; this is called *model selection*. In this dissertation, model selection was based on the log-likelihood of observing the test data (predictive fit), as described in Chapter 4.6 in Equation (11). The predictive fit was calculated for each topic number in the range of $K = 5 - 100$. The optimal topic number was then the one with the highest predictive fit. Note that calculating the log-likelihood for the training data (in-sample fit) was not appropriate here, since the objective of this dissertation is to *predict* the next item a customer would buy.

Model selection was conducted for each of the six data preparation variants separately. Figure 10 shows an example of the procedure for data preparation 11USC. For both topic models, it can be observed that the log-likelihood sharply increased for small topic

numbers and then leveled off until a maximum was reached. After the maximum, the log-likelihood slowly decreased again. The results imply that choosing a very small or very high topic number caused the predictive performance to deteriorate. The log-likelihood of both topic models was almost equivalent. The Sticky ATM reached its maximum at $K = 25$ and the ATM at $K = 40$. These topic numbers were then used to generate recommendations and compute the hit rates. For the remaining data preparation variants, comparable model selection results were achieved.

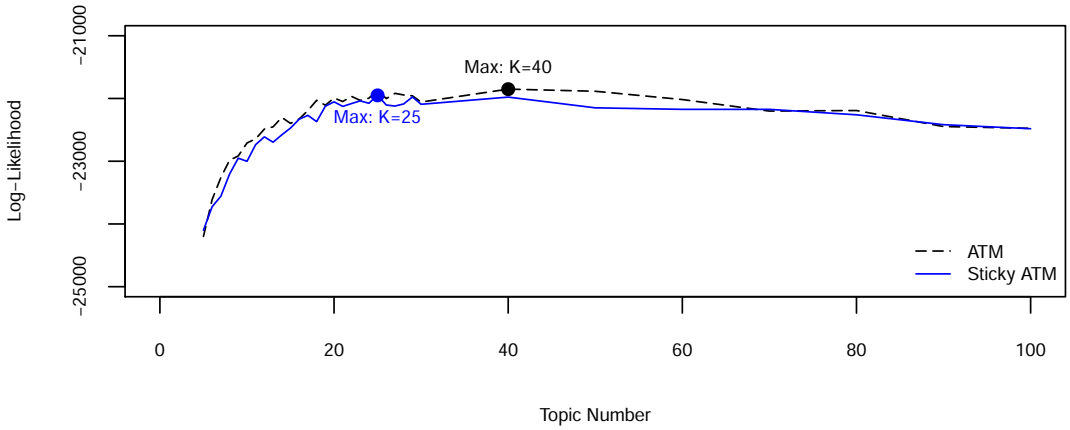


Figure 10: Predictive Fit of Topic Models for Data Preparation 11USC

7.1.2. Item and Topic Distribution

Topic models yield an individual *item distribution* for each topic (ϕ_k); that is, each item appears in a topic with a certain probability, and all those probabilities add up to 1. The high-probability items in the identified topics form meaningful topics. Table 7 displays eight example topics generated with the ATM model, where each topic shows the 10 most likely items for this topic. The topics were labeled by a veterinarian who is familiar with the individual products and for which purposes they are purchased. Items in italics were used to define the topic. The labels indicate that there are general topics such as *Old Cat*, *Old Dog*, or *Medicine Cabinet*. The topic models, however, also identified a collection of items that address specific pet problems, e.g., *Hot Spots*, *Bladder Stone*,

Bloated Stomach, *Joint Complaints*, and *Diarrhea*.

Topic models can also determine the most likely topics a customer is interested in by using the customer's *topic distribution* (θ_c); that is, a customer is represented in each topic with a certain probability, and all those probabilities add up to 1. Figure 11 shows four example customers and their topic distributions generated with the ATM model. The different examples indicate that customer preferences can be very different. In the upper right panel, the customer is likely to buy from multiple topics, whereas in the upper left panel, the customer is clearly interested in only a few topics. Some customers even have a strong preference for only one topic, as indicated in the lower left panel. The customer in the lower right panel appears in almost all topics with low probabilities and does not show a strong preference for one specific topic. The topic models were not only able to identify such heterogeneity between customers but also to reveal homogeneity between customers, which is expressed by similar topic distributions of customers. Such deep insights about customers' preferences help to make suitable and personalized recommendations.

	Old Cat	Old Dog	Hot Spots	Bladder Stone Cat
1	*Pfizer VMP Mobil 60 Kautabletten*	*Pfizer VMP 50 Tabletten*	*ecuphar OROZYME Zahnpflege-Gel 70 g*	*Vetoquinol Uro-Pet Paste 120 g*
2	Selectavet Gastrosel für Hunde und Katzen 30 Tabletten	*cp-pharma Relaxan forte 30 Tabletten*	Virbac C.E.T. Enzymatische Zahnpasta 70g	ecuphar OROZYME Zahnpflege-Gel 70 g
3	DERMAVET Lavapirox Shampoo 250 ml	*aniMedica Clorexyderm oto 150ml Ohrreiniger*	*Albrecht Nutri-Cal Paste 120,5 g*	*Hill's Prescription Diet c/d Multicare Chicken feline Frischebeutel – 12 x 85 g Nassfutter*
4	DERMAVET Lavapirox Fluid 250 ml	*Selectavet LEGAVIT KOMPLEX 30 Tabletten*	*Virbac Dermacool 50ml*	*Royal Canin Urinary s/o feline – Huhn – 12 x 100 g Frischebeutel*
5	*Hill's Prescription Diet k/d feline – gemischte Geschmacksrichtungen – 12 x 85 g Frischebeutel*	Oxbow Healthy Rabbit Pro 2kg	*Virbac EFA-Z 250 ml Flasche*	NutriLabs Canicox GR 100 Tabletten
6	*Pfizer VMP Mobil 60 Kautabletten*	cp-pharma Urocid Paste 120 g	*Coflex kohäsive Bandage 5,0 cm x 4,5 m blau*	almapharm astoral ImmuStim H 60 Tabletten
7	*Vetoquinol Enisyl-F Paste100ml Pumpflasche*	*Flexivet Go Tabletten*	Selectavet Equi Selenosel 1000 g	Vetoquinol Ipakitine Pulver 300g Dose
8	almapharm astoral MultiVital h.a. 500g für Hunde	*Virbac Pronefra 180 ml für Hunde und Katzen*	Vetoquinol Caniviton forte 30 - 1 kg - 1000 g	*cp-pharma Mielosan Salbe 100g*
9	Virbac Allerderm Spot-on 6 x 2 ml	Oxbow Healthy Rabbit Pro 2kg	Vetoquinol Ipakitine Pulver 180g Dose	*Royal Canin Urinary s/o feline – Rind – 12 x 100 g Frischebeutel*
10	*Royal Canin Calm feline – 4 kg Trockenfutter*	almapharm astoral Almazyme Pulver 120g	Virbac C.E.T. Enzymatische Zahnpasta 70g	Virbac Nutri-plus Gel 120,5g Tube Nutriplus
	Bloated Stomach Pets	**Dog with Joint Complaints**	**Medicine Cabinet**	**Chronic Diarrhea & Emergency Pharmacy**
1	*WDT Herbi Care Plus 200g*	Boehringer Ingelheim Canosan Katze 30 Kautabletten à 600 mg	Boehringer Ingelheim Canosan Kautabletten für Hunde 30 Stück à 2 g	*Albrecht Bene-Bac Gel 4 Tuben zu je 1 g*
2	NutriLabs Canicox GR 100 Tabletten	Bactisel HK 100 g Dose für Hunde und Katzen	Virbac EpiOtic Ohrreiniger 125 ml	*Oxbow Critical Care Beutel á 141g*
3	*Albrecht ProPre-Bac 50ml*	*cp-pharma Relaxan 60 Tabletten*	*WDT Korvimin ZVT + Reptil 200 g Pulver*	*Selectavet Bactisel H Gel für Hunde 30ml*
4	Canosan 1,3 kg Konzentrat Pellets 4%	Virbac Nutri-plus Gel 120,5g Tube Nutriplus	Albrecht Kamillenohrreiniger 200 ml	*aniMedica Clorexyderm Shampoo 250ml*
5	*Albrecht Dimeticon 50ml*	Oxbow Critical Care Beutel á 454g	*Selectavet B-Vitamin-Tabletten 20 Tabletten*	alfavet RodiCare akut 15 ml
6	Vetoquinol Bezo-Pet 120 g Paste	cp-pharma Herbax Paste 70g	Oxbow Critical Care Fine Grind 100g Beutel	*Selectavet Enterosel HK Gel 30 ml*
7	*Bunny goVet Gastroplex 300g*	*cp-pharma Relaxan forte 30 Tabletten*	*aniMedica curasal Dermafit forte 100ml*	alfavet RodiCare basic 1000g
8	Oxbow Critical Care Beutel á 36g	*fidavet Perloquan (Pillquan Pillis) 800 g Dose*	*BIODOR Pet ANIMAL1000 ml - mikrobiologischer Geruchsentferner, Reiniger*	alfavet RodiCare instant 170g
9	cp-pharma FiberFerm canine OBESITY-SATIETY 800g	*aniMedica Hyalutidin DC 2 x 125 ml*	*Virbac EpiOtic Ohrreiniger 125 ml*	alfavet DermaSterol Spender 125ml
10	almapharm astoral Almazyme 125 Tabs	*CEVA ADAPTIL Zerstäuber inkl. Flacon 48ml*	CEVA Feliway Zerstäuber Set incl. Flacon 48 ml	Albrecht Denticur RF^2 Kaustix S für Hunde bis 10kg 4x6 Stix

Table 7: Example Topics Revealed by ATM

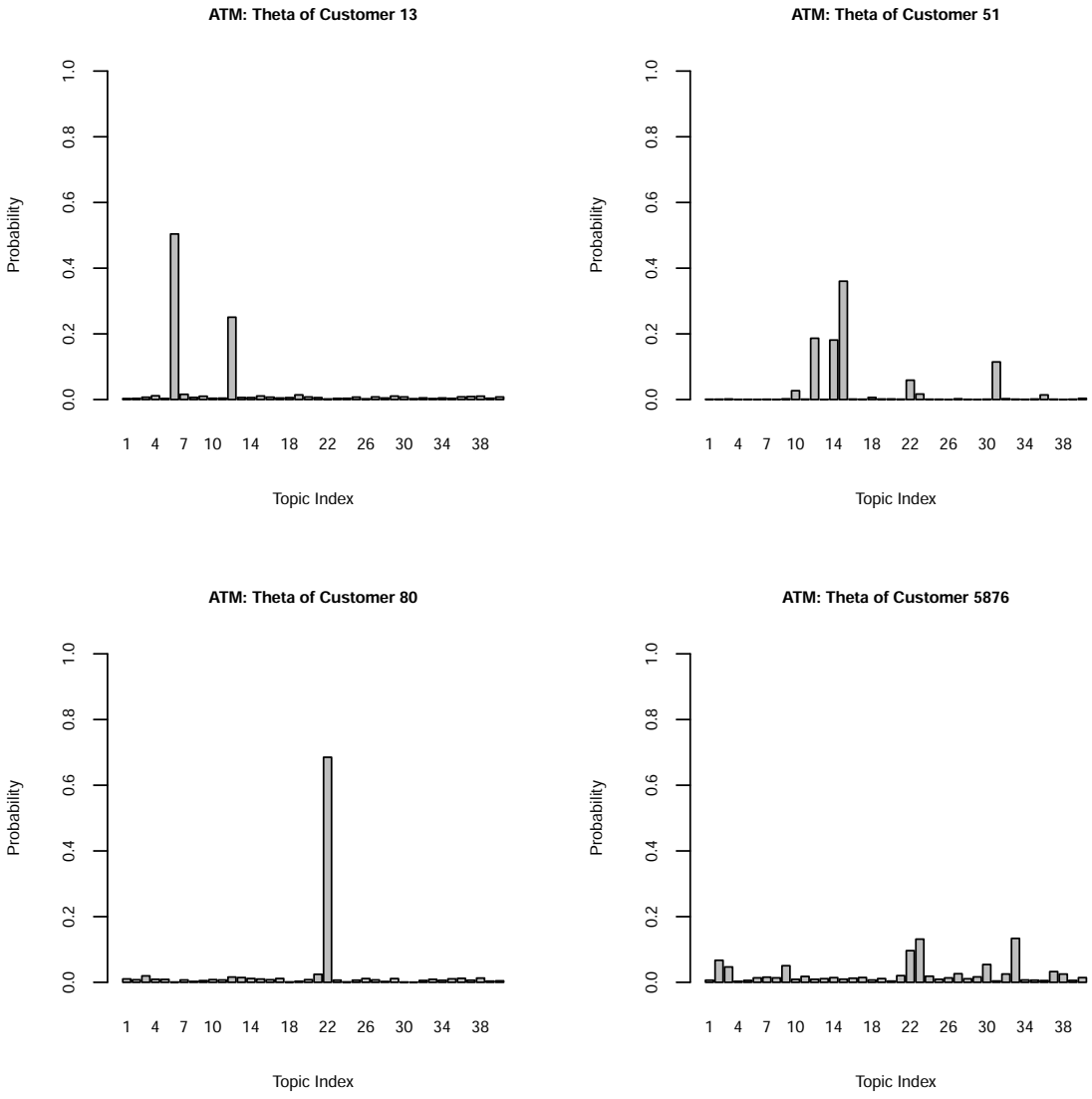


Figure 11: Example Topic Distributions for Some Selected Customers

7.2. Model Comparison

In this section, the recommender models are compared to one another by assessing their hit rates. For the topic models, two different algorithms for generating recommendations were tested (see Chapter 6.1.5). The approach of making recommendations based on *probability* $p_{c,p}$ always showed higher hit rates. Therefore, only those results are exhibited here. In the following sections, hit rates are reported for *all holdout customers* (7.2.1) and the various *subgroups* (7.2.2), as defined in Chapter 5.3. The exact hit rate values, according to which the figures in this chapter were created, are displayed in Appendix B. 2.

7.2.1. Hit Rates for All Holdout Customers

Figure 12 presents the hit rates for *all holdout customers*. Each of the four panels refers to different recommender set sizes, as indicated in the main header of each panel. The x-axis specifies the different data preparation variants and the y-axis the hit rates.[11] A look at the performance of the different models shows that CF and Bigram mostly outperformed the topic models except for data preparation 11USC. For this data preparation variant topic models were clearly better; their superiority became even more visible with a higher recommender set size. Hit rates of ATM and Sticky ATM were almost equivalent; neither model was clearly dominant. CF and Bigram also showed almost identical values. The Unigram model always yielded the lowest performance.

It was difficult to compare hit rates among the data variants because the test data was composed of different subgroups in different proportions. For example, the test data for evaluating the data preparation variants ALL, USO, and USC comprised many instances of *Cold Customer*, whereas the test data for evaluating the data preparation variant

[11]The hit rates were calculated as in Equation (16) in Chapter 6.3.

11USC solely contained *Old Customer, New SKU*.[12] For this reason, the next section shows hit rates only for the individual subgroups.

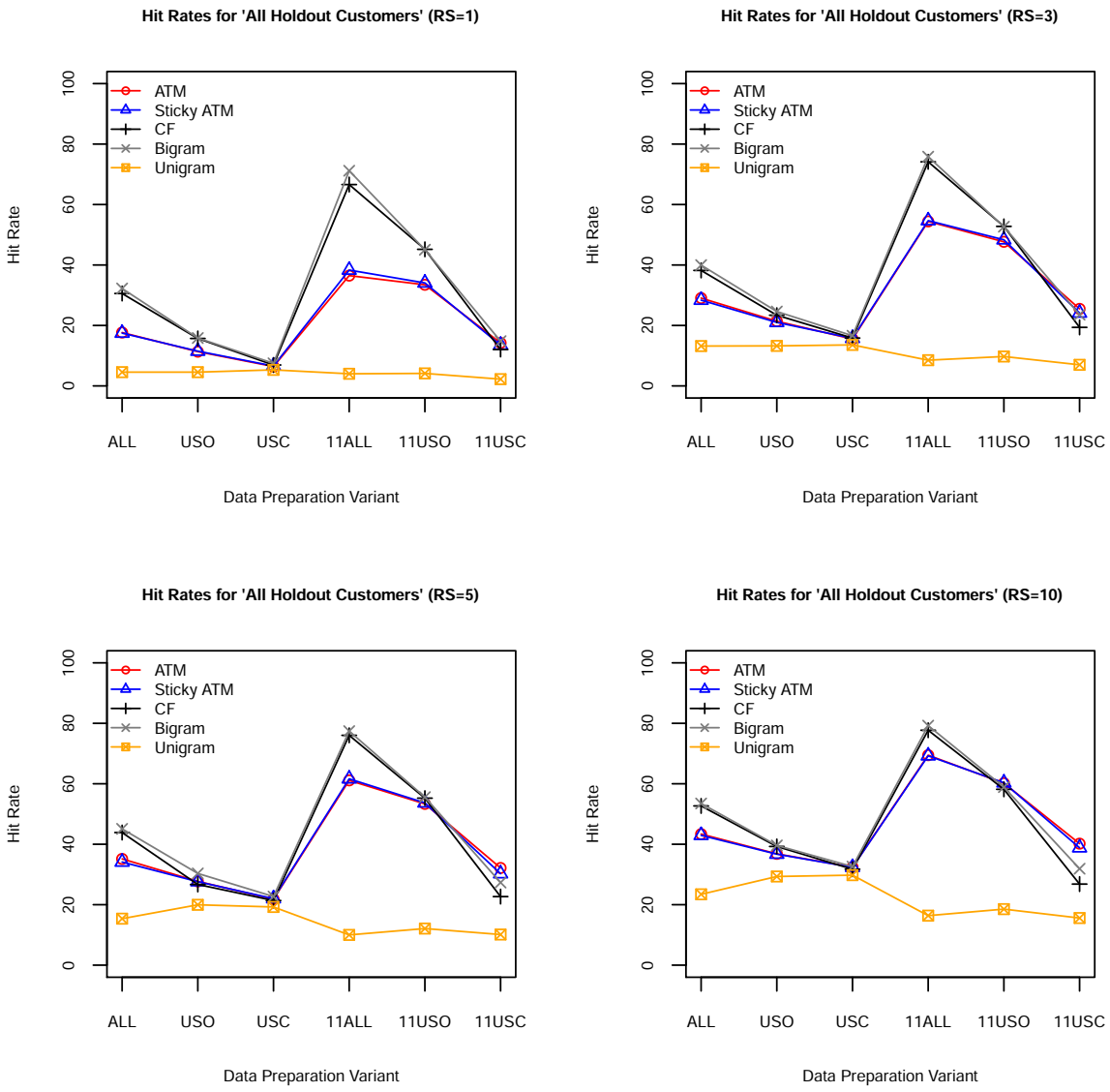


Figure 12: Hit Rates for All Holdout Customers

[12] Table 11, 14, 17, 20, 23 and 26 in Appendix B. 1 show the composition of the test data regarding the different subgroups for each data preparation variant.

7.2.2. Hit Rates for Subgroups

The predictive performance was studied in greater detail by separately considering different subgroups. Figure 13 and 14 show hit rates for the subgroup *Old Customer, Old SKU* and *Old Customer, New SKU*. Hit rates for the subgroup *Cold Customer* are not displayed because no considerable differences in the models' performance were found. The reason for this is that all models had to regress to marginal counts of products because no information was available for those customers. A comparison of Figure 13 and 14 reveals that hit rates differed strongly between the two subgroups. Hit rates for *Old Customer, Old SKU* were much higher than hit rates for *Old Customer, New SKU*. This implies that the models can more easily predict non-novel items than novel items.

With respect to predicting non-novel items, Figure 13 shows that CF and Bigram were extremely good, reaching hit rates of up to 93%. The topic models were not able to yield such high performances. In particular, for data preparation ALL and 11ALL, the performance difference between CF and Bigram versus topic models was the greatest. The Unigram model displayed the lowest hit rates. With data preparations USC and 11USC, no *Old Customer, Old SKU* was available in the data; hence, no hit rates can be shown.

With regard to the prediction of novel items, Figure 14 shows that topic models performed best with data preparation 11USC. Their performance superiority increased with a larger recommender set size. The results further reveal that topic models were much more sensitive to data preparation than their benchmark models. Whereas CF and Bigram display nearly vertical lines, meaning that hit rates did not change much between the data preparation variants, major variations were observed for the topic models. For them, hit rates increased most with the elimination of repeat items. The Unigram model showed fluctuations similar to those of the topic models, but its hit rates were by far the

lowest of all models.

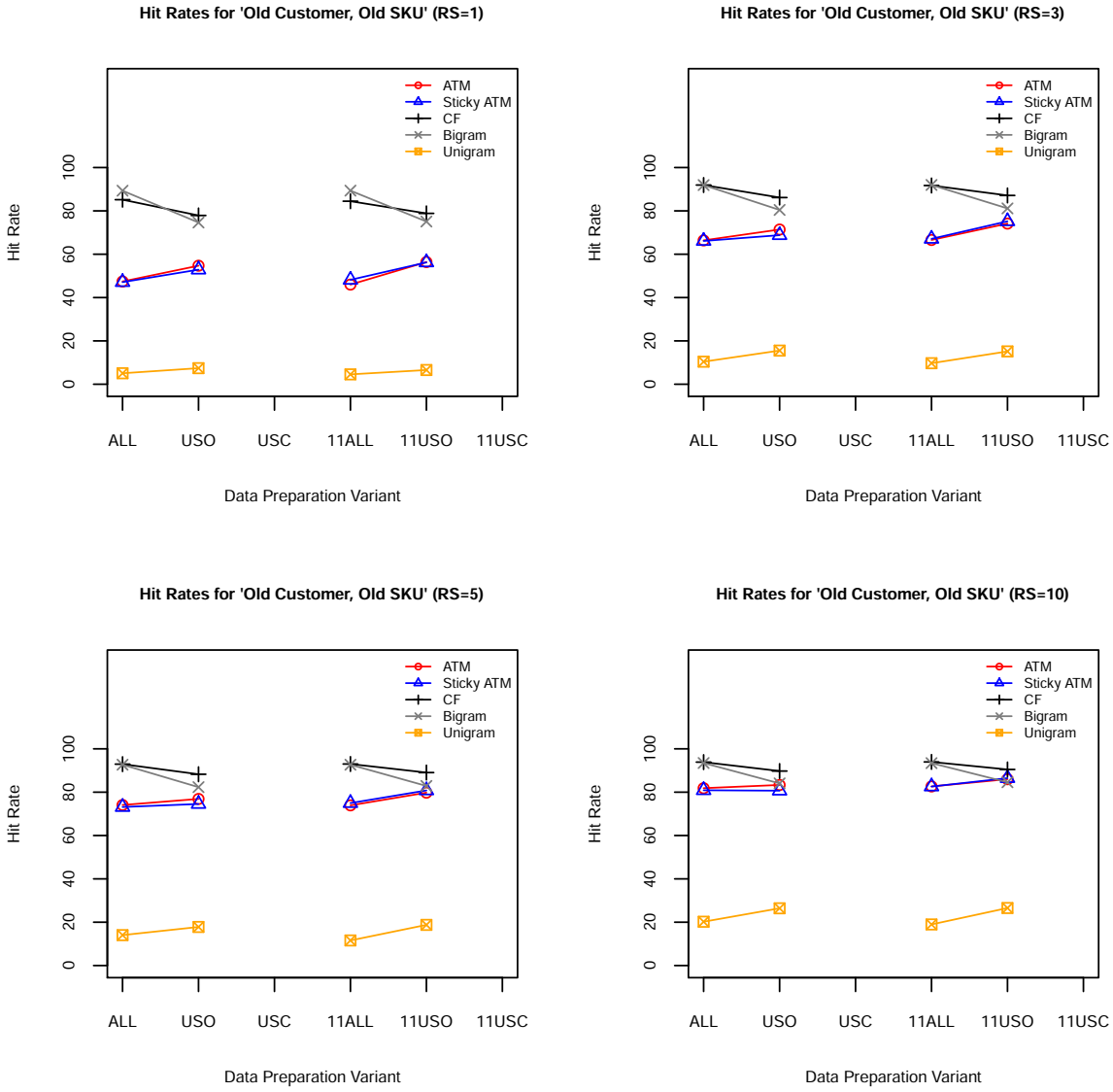


Figure 13: Hit Rates for the Subgroup *Old Customer, Old SKU*

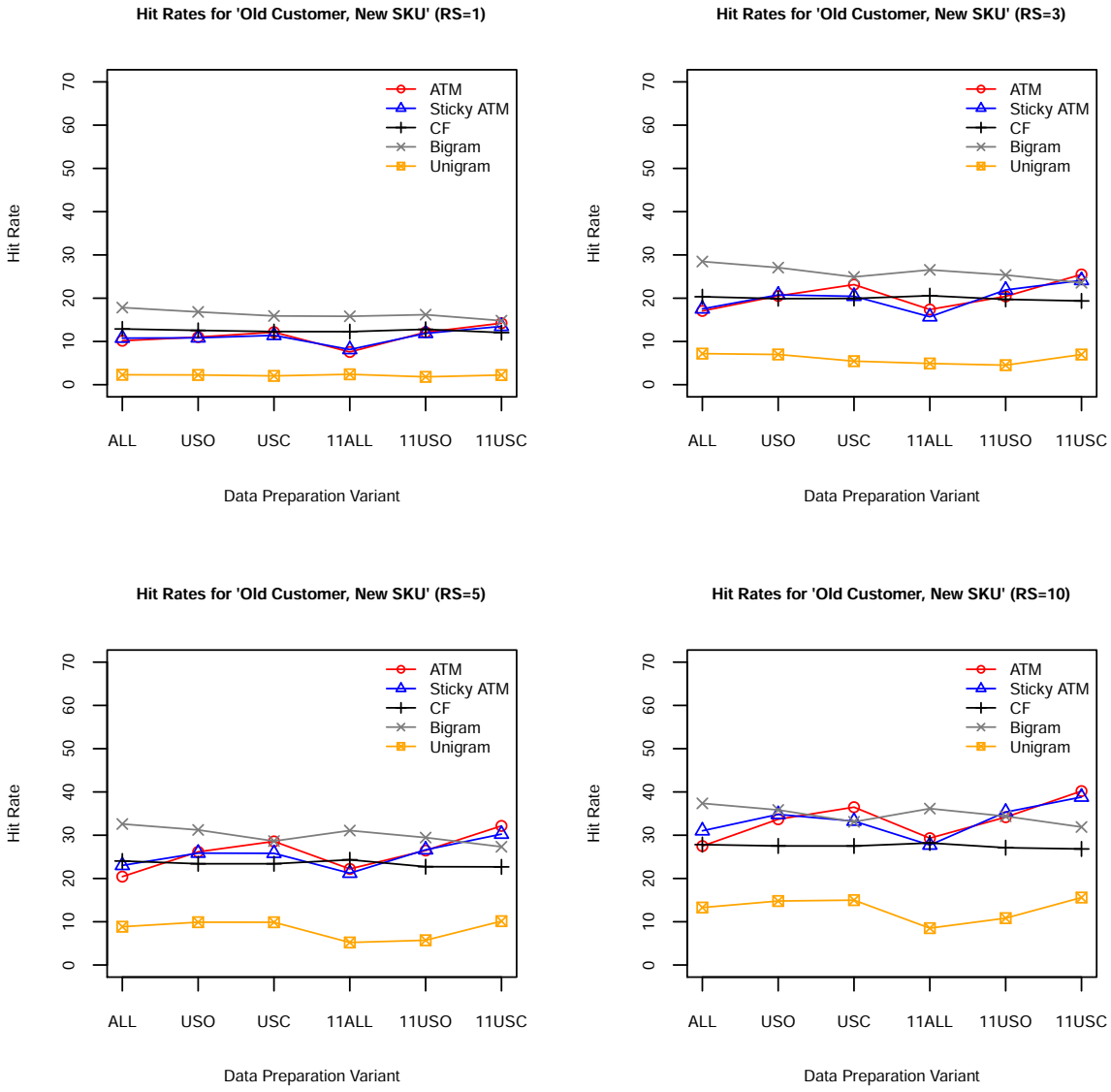


Figure 14: Hit Rates for the Subgroup *Old Customer, New SKU*

8. Discussion

The model comparison results showed that for recommending new items to customers, topic models performed better than the benchmark models with data preparation 11USC. This section examines this finding more closely and explains why this is the case. The discussion first looks at which *data properties* (8.1) are changed by the different data preparation variants. Subsequently, differences in the model results *phi* and *theta* (8.2) are shown for the different data preparation variants. Lastly, it is demonstrated how the data preparation changes the *ranking* (8.3) of items within the topics. This has considerable ramifications on which items are finally recommended to customers.

8.1. Differences in Data Properties

All six data preparation variants led to certain modifications in the data. The concrete data properties that were changed are listed and explained below:

1. *General size of the data set:* Through the elimination of *single-item customers* and/ or *repeat items*, the general size of the original data set was changed. As a result, some data scenarios had more customers than others. In addition, the number of items (total and unique) differed among the data scenarios. Appendix B. 1 shows descriptive statistics for each of the data preparation variants.

2. *Item distribution overall:* A less obvious property that was altered by the different data preparation variants is the concentration of the "item distribution overall." This distribution displays how often each unique item is purchased across *all* customers. If some items are purchased much more often than others, then the concentration would be very high. Otherwise, if the item purchases are more equally distributed across the entire product assortment, the concentration would be low. The concentration of this distribution is known to have an influence on which items are recommended by an algorithm. For instance, data sets with a high concentra-

tion lead to more recommendations of popular items, whereas data sets with a low concentration include more items from the long tail in the recommendations. Already Herlocker et al. (2004) found that the concentration of this distribution is relevant in a recommender context. Figure 15, in which the 200 most frequent items are plotted for each of the different data scenarios, shows that the concentration differed considerably. The frequency distribution became less concentrated with a more selective data preparation. For example, for data preparation ALL, frequencies of up to 8,000 were observed for the most popular items. These frequencies, however, were much lower for data preparation 11USC, where the maximum count was around 1,000.

3. *Item distribution per customer:* This distribution displays how often each unique item is purchased by *one* customer. Item purchases are not aggregated across customers (as in "item distribution overall"), but it is remained at the customer level. The distribution was least concentrated for data preparations USC and 11USC because all repeat items at the customer level were eliminated from the data. As a result, item counts could only take values of 1.

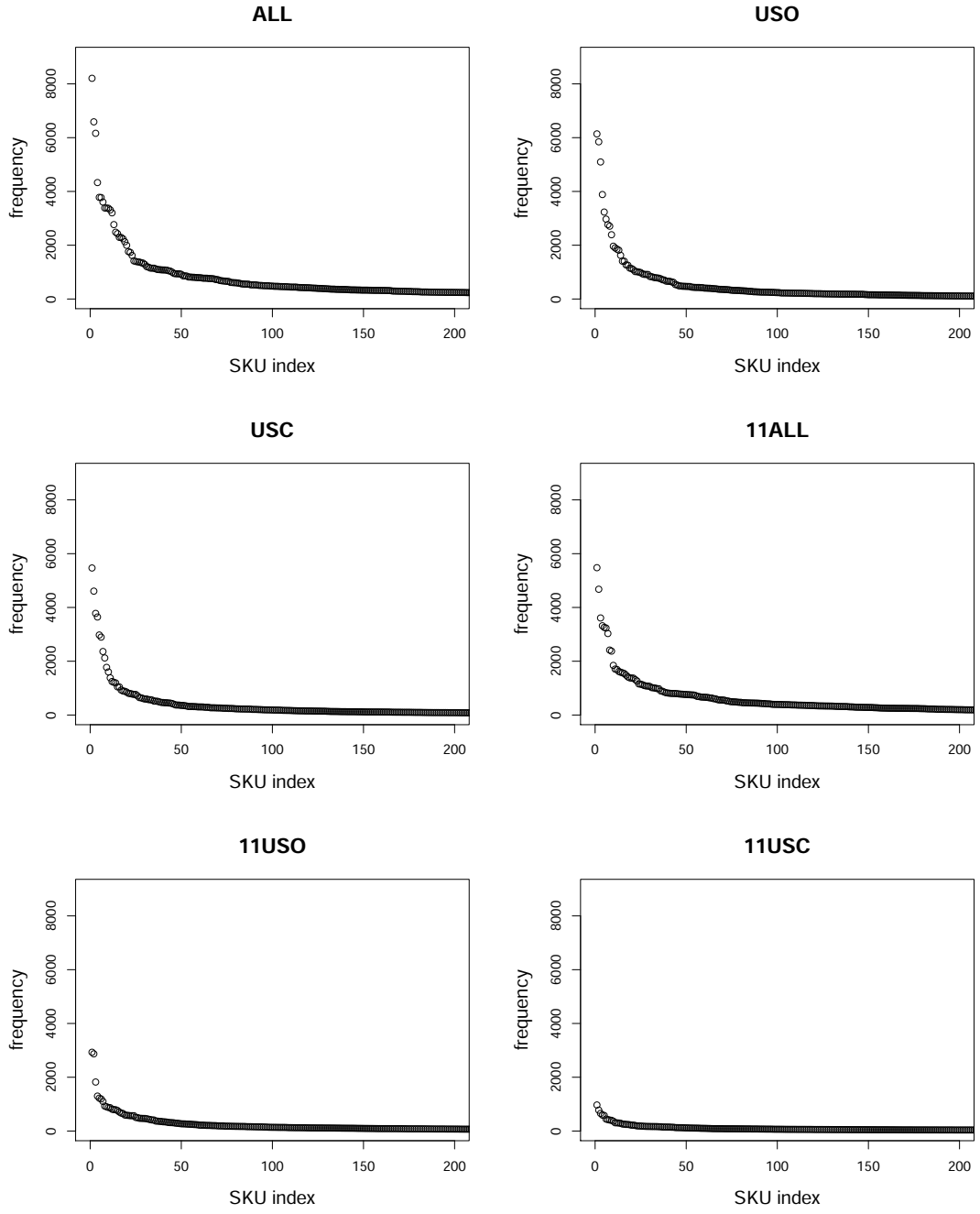


Figure 15: Concentration of *Item Distribution Overall*

8.2. Differences in Phi and Theta

To generate recommendations with topic models, the phis and thetas are needed. The phis represent the item probabilities per topic and provide information on how important an item is for a topic. The thetas represent the topic probabilities per customer and provide information on how much a customer is interested in a topic. Figures 16 and 17 show both variables as cumulative probabilities for each of the data preparation variants.

From the *phi* plots in Figure 16, it can be observed that the distributions differed greatly between the single data preparation variants. The curves are more concentrated when repeat items remained in the data, and they are less concentrated when more repeat items were removed from the data. For example, for data preparation ALL, it took far fewer SKUs to obtain a cumulative probability of 80% (less than 100 SKUs) than for data preparation USC (up to around 200 SKUs). The elimination of single-item customers further reduced the concentration. For data preparation 11USC, more than 500 SKUs were required to reach the same cumulative probability.

Also, differences in *theta* are apparent among the data preparation variants. This is illustrated in Figure 17. With data preparations ALL, USO, and USC, some customers with almost equal topic probabilities are observed. Those customers represent *Cold Customer*. They had not yet conducted any item purchases, i.e., for them, no information was available on their topics of interest. Their topic probabilities were therefore spread almost equally across all topics by the model. The lines on the diagonal represent their thetas. Another striking feature of the plots is that the theta curves are less concentrated when repeat items were deleted. For example, data preparation USC and 11USC had more customers with lower probabilities (around 0.2) and fewer customers with high probabilities (above 0.8) for the first few topics. Those lower probabilities imply that customers were generally interested in more different topics.

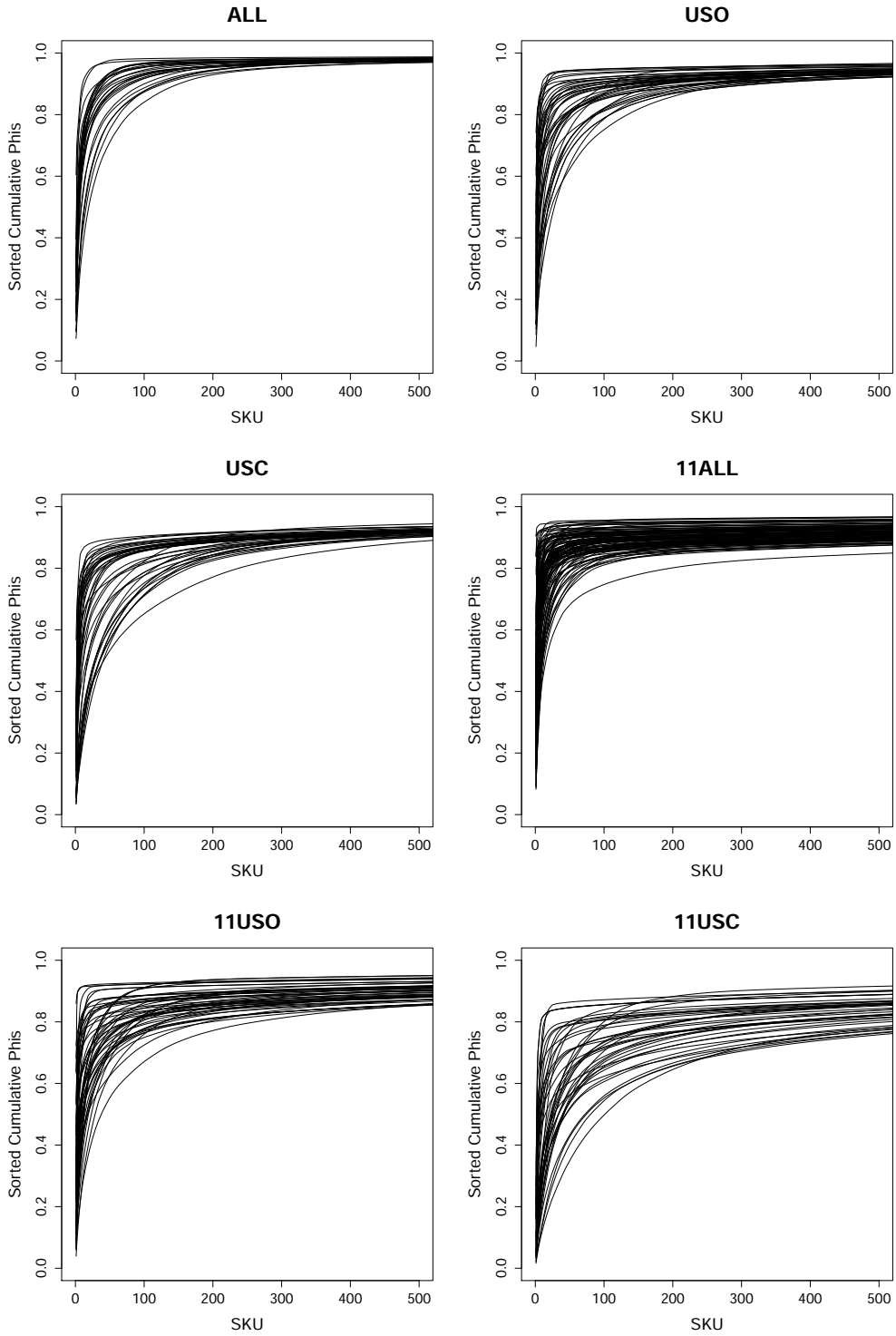


Figure 16: Cumulative Probabilities for Phi

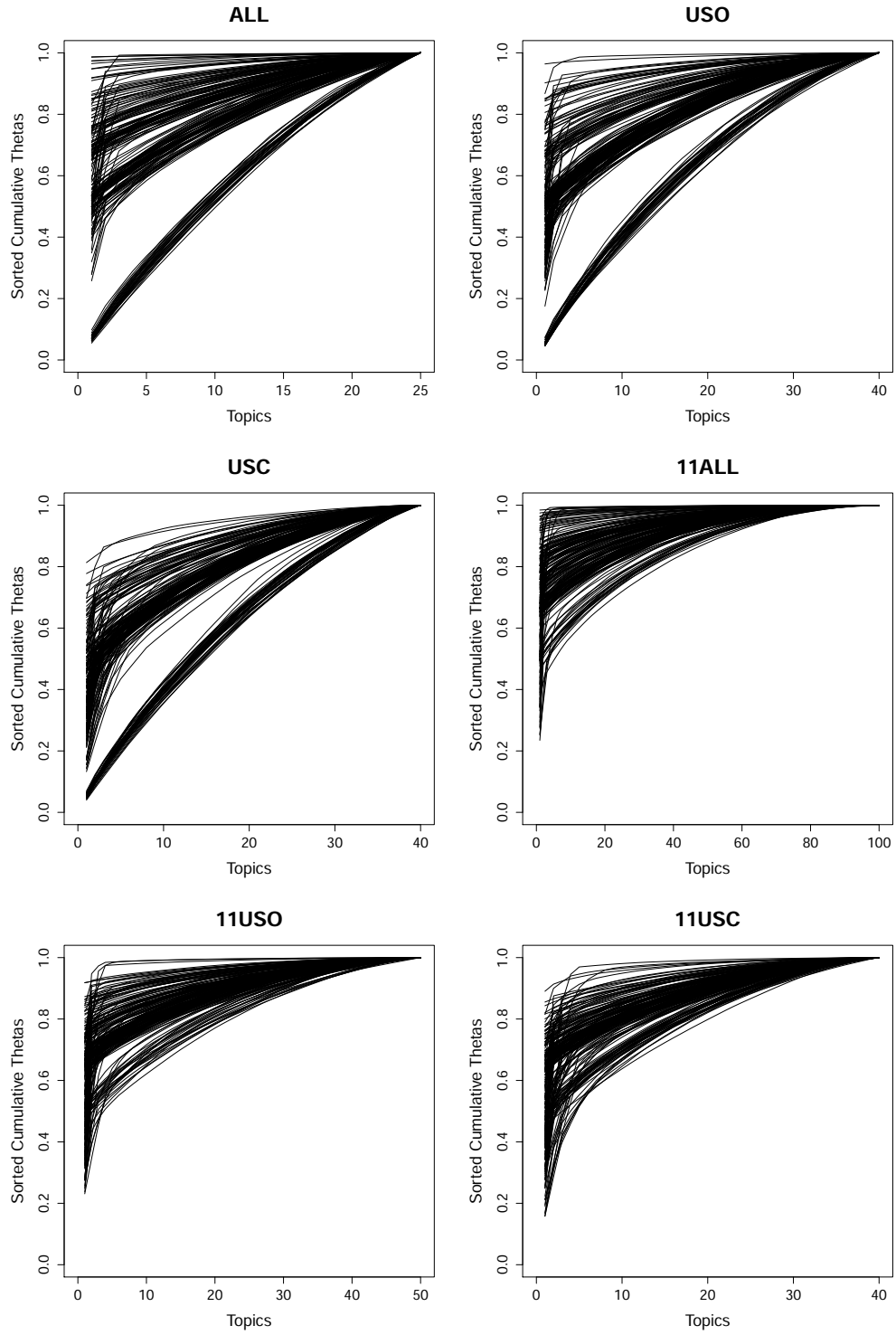


Figure 17: Cumulative Probabilities for Theta

8.3. Differences in Ranking

Central for creating the final recommendation list that is displayed to customers is the ranking of items within the topics. Items that appear higher in the topics are more likely to be recommended and items that appear lower in the topics are less likely to be recommended. Since the goal of this dissertation is to recommend *new* products to customers, novel products should therefore also appear in the upper ranks of the topics, and popular items, commonly known as less novel, should appear in the lower ranks of the topics. The following table shows that such a ranking can be achieved with data preparation 11USC.

Rank	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ALL	1	1	1	1	1	1	1	1
11USC	55	105	39	653	58	84	86	7
Rank	**S9**	**S10**	**S11**	**S12**	**S13**	**S14**	**S15**	
ALL	2	1	1	1	1	1	1	
11USC	55	1	7	3	90	68	37	

Table 8: Highest Ranking across Topics for the 15 Most Frequently Deleted Items between Data Preparation ALL and 11USC

For this table, the 15 most frequent items (S1-S15) deleted between data preparation ALL and 11USC were determined and their highest rankings across topics were identified. For data preparation ALL, all 15 items except S9 appeared in first place on one of the topics. This implies that topics generated with data preparation ALL are clearly dominated by frequently purchased items. For data preparation 11USC, the same items moved down in the ranking, which decreased their chance of appearing in the recommender set. Only S10 kept the same ranking, while S8, S11, and S12 remained at comparable rankings. As popular items go down in the ranking, less popular items from the long tail must move up. This may finally lead to recommendations that are perceived as more useful by customers.

Part IV.

Conclusion

In this dissertation, several recommender models were compared to each other by assessing their predictive performance for online purchase data. Influenced by the topic modeling literature, the topic models ATM and Sticky ATM were considered for analysis and contrasted against the benchmark models Unigram, Bigram, and CF. Moreover, this work set its focus on generating new product recommendations. For this purpose, it moved beyond the evaluation of recommender models based on accuracy alone. A new pre-processing approach was presented in which the data is pre-processed prior to model running in such a way to include more novel items from the long tail in the recommendations. This thesis closes with answering the research questions raised in Chapter 2 and giving a brief outlook for future research.

1. *Are topic models effective in recommending novel items to customers? How do they perform when predicting non-novel items? Are there recommender models that are better suited for the one task or the other?*

   All recommender models were evaluated based on their ability to predict novel and non-novel items to customers. With respect to the prediction of novel items, topic models outperformed all other models with data preparation variant 11USC. Regarding the prediction of non-novel items, CF and Bigram always performed better than topic models, especially with the data preparation variants ALL and 11ALL. The implication is that different models should be used, depending on the company's goals. If the objective is to remind a customer of certain products to repurchase, CF and Bigram work well. To recommend new products that customers may not have been aware of, topic models are more suitable.

2. *How do the two topic models perform? Can the Sticky ATM model outperform the ATM model?*

   It was expected that the Sticky ATM would perform better than the ATM model because it additionally considers stickiness at the item level. The results, however,

showed almost identical hit rates for both topic models. A possible explanation for this could be the composition of the data. The data was made up of many customers who only bought very few items. In the case of few item purchases, stickiness can hardly come to bear. For the model to show its true strength, longer buying histories are likely to be necessary. This would have to be tested with an appropriate data set.

3. *Which data preparation is most suitable for generating new product recommendations? And why? Are there differences between the recommender models?*

   The recommender models reacted with different sensitivities to the different data preparation variants. Topic models were most sensitive to pre-processing and displayed the largest differences in hit rates among the data preparation variants. They performed best when repeat items and single-item customers were eliminated from the data (data preparation variant 11USC). This is because neither data element is very helpful for topic models in generating new product recommendations. Repeat items drive up the counts of an item disproportionally to items ordered in smaller quantities. The implication is that repeat items dominate the revealed topics and end up in the recommendation lists. Less popular items, however, which often indicate new and emerging customer preferences, do not make it into the recommender set. The problem with single-item customers is that they do not provide information about item interaction and thus which other new items should be recommended. The discussion section detailed, why this data preparation variant worked so well for topic models. The hit rates of CF were mainly unaffected due to the different pre-processing variants. The performance of the Bigram model slightly deteriorated when repeat items were deleted from the data. As input, the model requires items to remain in the sequence in which they were ordered. Since this sequence is partly no longer maintained after repeat items were removed, the performance may have decreased. The Unigram model showed a slight improvement

in hit rates when repeat items were eliminated.

Based on the results of this work, five avenues for future research are apparent. First, for further validation, topic models should be applied to additional data sets. The data set used for this dissertation comprises very short purchase histories and many repeat items. However, these characteristics are not representative of the data of all online shops. There may be online shops with much longer buying histories (e.g., online food retailers), lower purchase frequencies and quantities (e.g., high-end and luxury goods), or even much smaller or larger product assortments (e.g., very specialized shops vs. Amazon). Instead of using real data sets, a simulation could also be carried out in which data sets of different structures are simulated. Appendix C shows an example of how this can be done. Second, future research efforts could extend topic models by incorporating additional purchase information, such as price, purchase quantity, expenditure, or product category. For example, Iwata and Sawada (2013) developed a topic model that clusters related items with their prices. Such an approach could help to generate product recommendations that align with customers' willingness to pay. Third, the use of a dynamic topic model (Blei and Lafferty 2006) for purchase data would be another interesting path of research. The ATM and Sticky ATM models assume that the topics are stable over time. However, this assumption is not always borne out by reality. New trends, technologies, and product introductions are continually changing customer preferences and buying behavior. A dynamic topic model would account for this. Fourth, further research could also focus on the development of a hybrid recommender system that uses topic models together with other models. For example, the model comparison could be extended by a Bigram Topic Model (Wallach 2006) or Collaborative Topic Model (Wang and Blei 2011; Wilson, Chaudhury, and Lall 2014). Lastly, the novelty of product recommendations could be improved by combining various beyond accuracy measures. For instance, the new pre-processing approach could be used together with some of the post-processing approaches discussed in the literature.

To conclude, topic models are a promising approach for generating new product recommendations in online shops. The present analysis has shown the suitability of those models based on offline evaluation. To put topic models into practice, they should be tested against the benchmark models in some live user experiments. This would finally allow measurement of customers' actual reaction to recommendations and whether those recommendations are really perceived as more novel.

Appendix

A. Appendix for

Theoretical Foundation

A. 1. Generative Process of LDA in R

This appendix demonstrates how the generative process of LDA can be implemented in R and what the resulting data looks like. A. 1.1 shows the example data that is generated, and A. 1.2 shows the corresponding R code.

A. 1.1. Example

Imagine a text corpus has $D = 5$ documents with each document comprising $N_d = 4$ words, $K = 3$ topics, and a vocabulary of $V = 6$ unique words.[13] For the hyper-parameters, symmetric priors are selected: $\alpha = 0.5$ and $\beta = 1$. Running the generative process with those parameters yields the following data:

Word	Topic	Document
1	3	1
5	3	1
3	2	1
3	3	1
2	3	2
1	2	2
4	2	2
3	3	2
6	1	3
4	1	3
3	1	3
6	1	3
4	1	4
2	1	4
4	1	4
6	1	4
4	2	5
4	1	5
3	1	5
4	2	5

Figure 18: Data Matrix

Each row in the matrix of Figure 18 corresponds to one word in the text corpus. The first column contains the words, the second column contains the topic assignment for

[13] Note that for demonstration purposes, small numbers were selected so that the results can be fully displayed here. In reality, the number of documents and words as well as the size of the vocabulary are typically much larger. For simplicity's sake, the same fixed number of words was selected for each document. To get a varying number of words for each document, the Poisson distribution, for instance, could be used (compare comments in R code).

each word, and the third column contains the document each word belongs to.

The generative process also provides all per-topic word distributions (ϕs) and per-document topic distributions (θs), which are summarized in the following matrices:

ϕ		Word 1	Word 2	Word 3	Word 4	Word 5	Word 6
Topic 1	$\phi_{k=1}$	0.05	0.05	0.25	0.35	0.01	0.29
Topic 2	$\phi_{k=2}$	0.19	0.01	0.17	0.49	0.04	0.10
Topic 3	$\phi_{k=3}$	0.02	0.29	0.28	0.17	0.15	0.09

θ		Topic 1	Topic 2	Topic 3
Document 1	$\theta_{d=1}$	0.24	0.10	0.66
Document 2	$\theta_{d=2}$	0.07	0.61	0.32
Document 3	$\theta_{d=3}$	1.00	0.00	0.00
Document 4	$\theta_{d=4}$	0.99	0.00	0.01
Document 5	$\theta_{d=5}$	0.28	0.72	0.00

Figure 19: Phi and Theta Matrix

It should be noted that for the ϕs (upper matrix of Figure 19), the word probabilities in each topic always need to sum to one ($\sum \phi_k = 1$), e.g., for topic 2:

$$\sum \phi_{k=2} = 0.19 + 0.01 + 0.17 + 0.49 + 0.04 + 0.10 = 1$$

The same is true for the θs (lower matrix of Figure 19). Here the topic shares within a document must sum to one ($\sum \theta_d = 1$), e.g., for document 1:

$$\sum \theta_{d=1} = 0.24 + 0.10 + 0.66 = 1$$

Based on the generated data, the count matrices N^{KV} and N^{DK} and the decremented count matrices N^{KV}_{-i} and N^{DK}_{-i}, reduced by the current word entry i, can be created. Those matrices are necessary to reversing the generative process and estimating the LDA model. The count matrices are displayed in Figure 20 and Figure 21. For the decremented count

matrices, it is assumed that the fifth word in the text corpus is excluded. A look at the data matrix gives word = 2, topic = 3, and document = 2 for the fifth word. Comparing the decremented count matrices with the regular count matrices shows that only one entry (in blue) changes by "-1" and all other cells remain the same.

N^{KV}		Word 1	Word 2	Word 3	Word 4	Word 5	Word 6
Topic 1	$n_{k=1}$	0	1	2	4	0	3
Topic 2	$n_{k=2}$	1	0	1	3	0	0
Topic 3	$n_{k=3}$	1	1	2	0	1	0

N^{DK}		Topic 1	Topic 2	Topic 3
Document 1	$n_{d=1}$	0	1	3
Document 2	$n_{d=2}$	0	2	2
Document 3	$n_{d=3}$	4	0	0
Document 4	$n_{d=4}$	4	0	0
Document 5	$n_{d=5}$	2	2	0

Figure 20: Count Matrices N^{KV} and N^{DK}

N^{KV}_{-i}		Word 1	Word 2	Word 3	Word 4	Word 5	Word 6
Topic 1	$n_{k=1}$	0	1	2	4	0	3
Topic 2	$n_{k=2}$	1	0	1	3	0	0
Topic 3	$n_{k=3}$	1	**0**	2	0	1	0

N^{DK}_{-i}		Topic 1	Topic 2	Topic 3
Document 1	$n_{d=1}$	0	1	3
Document 2	$n_{d=2}$	0	2	**1**
Document 3	$n_{d=3}$	4	0	0
Document 4	$n_{d=4}$	4	0	0
Document 5	$n_{d=5}$	2	2	0

Figure 21: Decremented Count Matrices N^{KV}_{-i} and N^{DK}_{-i} for i=5

A. 1.2. R Code

```
######################################################################
## LATENT DIRICHLET ALLOCATION (LDA):
## Generative Process
######################################################################
######################################################################
##  This code comes from the PhD seminar "Topic Models" held
##  by Prof. Joachim Bueschken in the Summer Semester 2016
##  and is adapted to the purposes of this dissertation.
######################################################################

library(bayesm)
library(data.table)
rm(list=ls())
set.seed(91)

##-------------------------------------------------##
## Set Parameters:
K = 3               # number of topics
V = 6               # number of unique words (vocabulary)
D = 5               # number of documents
N_d = 4             # number of words per document

alpha = 0.5         # hyper-parameter for theta
beta = 1            # hyper-parameter for phi
##-------------------------------------------------##

######################################################################
## 1. DRAW OF PHI (FOR EACH TOPIC)
######################################################################
Phi = matrix(NA,V,K)
for(k in 1:K){Phi[,k] = rdirichlet(rep(beta,V))}
## --> each column corresponds to phi_k

######################################################################
## 2.A. DRAW OF THETA (FOR EACH DOCUMENT)
######################################################################
Theta = matrix(NA,K,D)
```

```
38 for(d in 1:D){Theta[,d] = rdirichlet(rep(alpha,K))}
39 ## --> each column corresponds to theta_d
40
41 ########################################################################
42 ## 2.B.i  DRAW OF Z (FOR EACH WORD IN DOCUMENT D)
43 ## 2.B.ii DRAW OF W (FOR EACH WORD IN DOCUMENT D)
44 ########################################################################
45 Data = NULL
46 data = NULL
47
48 for(d in 1:D){
49
50   ##-------------------------------------------------------------##
51   ## Determine number of words for document d:
52   ##-------------------------------------------------------------##
53   ## Fixed number of words:
54   N_d = 4
55
56   ## or:
57   ## Draw from Poisson distribution:
58   #lambda = 10
59   #N_d = rpois(1,lambda)
60
61   ##-------------------------------------------------------------##
62   ## Draw topic assignment z for each word in document d
63   ##-------------------------------------------------------------##
64   z = rep(NA,N_d)
65   for(n in 1:N_d){z[n] = which.max(rmultinom(1,1,Theta[,d]))}
66
67   ##-------------------------------------------------------------##
68   ## Draw word w for each word in document d
69   ##-------------------------------------------------------------##
70   w = rep(NA,N_d)
71   for(n in 1:N_d){w[n] = which.max(rmultinom(1,1,Phi[,z[n]]))}
72
73   ##-------------------------------------------------------------##
74   ## Store data for each document
75   ##-------------------------------------------------------------##
76   Data[[d]] = list(d = d, N_d = N_d, z = z, w = w)
```

```
77
78    ## Get data of current document:
79    data.new = cbind(Data[[d]]$w, Data[[d]]$z, Data[[d]]$d)
80
81    ## Stack data across all documents
82    data = rbind(data, data.new)
83    ## --> matrix: where each row corresponds to one word
84
85  }
86
87  ##########################################################################
88  ## ADDITIONAL ANALYSES
89  ##########################################################################
90  ##-------------------------------------------------------------##
91  ## Create Data Matrix (Figure 18)
92  ##-------------------------------------------------------------##
93  colnames(data) <- c("w","z","d")
94  data
95  w <- data[,"w"]
96  z <- data[,"z"]
97  d <- data[,"d"]
98
99  ##-------------------------------------------------------------##
100 ## Create Phi and Theta Matrix (Figure 19)
101 ##-------------------------------------------------------------##
102 round(t(Phi),2)
103 round(t(Theta),2)
104
105 ##-------------------------------------------------------------##
106 ## Create Count Matrices (Figure 20)
107 ##-------------------------------------------------------------##
108 ## N_KV COUNT MATRIX: containing all n_kv
109 N_kv = matrix(0,K,V)
110 rows = as.numeric(unlist(labels(table(z)))) ## (account for all "0"-rows)
111 cols = as.numeric(unlist(labels(table(w)))) ## (account for all "0"-cols)
112 N_kv[rows,cols] = table(data.table(cbind(z,w)))
113
114 ### N_DK COUNT MATRIX: containing all n_dk
115 N_dk = matrix(0,D,K)
```

```
116 rows = as.numeric(unlist(labels(table(d)))) ## (account for all "0"-rows)
117 cols = as.numeric(unlist(labels(table(z)))) ## (account for all "0"-columns)
118 N_dk[rows,cols] = table(data.table(cbind(d,z)))
119
120 ##-------------------------------------------------------##
121 ## Create Decremented Count Matrices (-i) (Figure 21)
122 ##-------------------------------------------------------##
123 ## Example: Let's assume, we exclude the 5th word from our data
124 ## --> see: data[5,]
125 data[5,]
126
127 N_kv_reduced <- N_kv
128 N_dk_reduced <- N_dk
129
130 N_kv_reduced[data[5,"z"],data[5,"w"]] = N_kv[data[5,"z"],data[5,"w"]]-1
131 N_dk_reduced[data[5,"d"],data[5,"z"]] = N_dk[data[5,"d"],data[5,"z"]]-1
```

A. 2. Derivation of Collapsed Gibbs Sampler

The derivation of the collapsed Gibbs sampler is based on the technical notes of Heinrich (2008), Büschken (2013), and Darling (2011). In this sampler, the parameters ϕ and θ are integrated out. Through this procedure, the posterior distribution in Equation (8) in Chapter 4.4 reduces to:

$$p(z|w, \alpha, \beta) = \frac{p(z, w|\alpha, \beta)}{p(w|\alpha, \beta)} \tag{17}$$

This new posterior distribution is also intractable to compute because it has the same denominator $p(w|\alpha, \beta)$, which involves the sum over an exponentially large number of combinations. Thus, approximation techniques need to come into play. Instead of going through the full sampling scheme for every latent variable (ϕ, θ, z) of the LDA, the collapsed Gibbs sampler simply requires the sampling of z because this is the sole unknown parameter left when collapsing ϕ and θ. The only full conditional is therefore $p(z_i|z_{-i}, w, \alpha, \beta)$, which makes the sampling scheme very easy. The probabilities ϕ and θ are calculated after Gibbs sampling with a Dirichlet update using the final count matrices N^{KV} and N^{DK} (and hyper-parameters α and β).

The derivation of the full conditional $p(z_i|z_{-i}, w, \alpha, \beta)$ requires formulation of the joint distribution with ϕ and θ integrated out. The following section therefore examines this joint distribution first (A. 2.1). It continues with the derivation of the full conditional (A. 2.2). Finally, it is explained how to obtain ϕ and θ (A. 2.3). An implementation of the collapsed Gibbs sampler in R is further added to this appendix (A. 2.4).

A. 2.1. Joint Distribution

The joint distribution and its factorization were specified in Chapter 4.3.3. When ϕ and θ are integrated out, the joint distribution becomes:

$$
\begin{aligned}
p(w, z|\alpha, \beta) &= \int \int p(\phi, \theta, z, w|\alpha, \beta)\, d\theta\, d\phi \\
&= \int \int p(\phi|\beta) \cdot p(\theta|\alpha) \cdot p(z|\theta) \cdot p(w|\phi, z)\, d\theta\, d\phi \\
&= \int p(w|\phi, z) \cdot p(\phi|\beta)\, d\phi \int p(z|\theta) \cdot p(\theta|\alpha)\, d\theta \\
&= p(w|z, \beta) \cdot p(z|\alpha) \qquad (18)
\end{aligned}
$$

The joint distribution results in the product of the two terms $p(w|z, \beta)$ and $p(z|\alpha)$. These terms are now examined separately.

Consider the *first term* $p(w|z, \beta)$. To obtain this distribution, ϕ needs to be integrated out. This entails the following calculations:

$$
\begin{aligned}
p(w|z, \beta) &= \int p(w|\phi, z) \cdot p(\phi|\beta)\, d\phi \\
&= \int \prod_{k=1}^{K} \prod_{v=1}^{V} \phi_{k,v}^{n_{k,v}} \cdot \prod_{k=1}^{K} \frac{1}{B(\beta)} \prod_{v=1}^{V} \phi_{k,v}^{\beta_v - 1}\, d\phi_k \\
&= \int \prod_{k=1}^{K} \frac{1}{B(\beta)} \prod_{v=1}^{V} \phi_{k,v}^{n_{k,v} + \beta_v - 1}\, d\phi_k \\
&= \prod_{k=1}^{K} \frac{1}{B(\beta)} \int \prod_{v=1}^{V} \phi_{k,v}^{n_{k,v} + \beta_v - 1}\, d\phi_k
\end{aligned}
$$

$$= \prod_{k=1}^{K} \frac{B(n_k + \beta)}{B(\beta)} \tag{19}$$

The expression $p(w|\phi, z)$ is a Multinomial distribution and $p(\phi|\beta)$ is a Dirichlet distribution. Both expressions were specified in Equation (7) and Equation (4) in Chapter 4.3.3 and are plugged in to the equation above. Subsequently, the exponents are summarized and the constants are rearranged. According to Wang (2008, p. 5) and Büschken (2013, p. 10), the integral of a Dirichlet distribution equals its normalizing constant. So in the above calculation, it applies:

$$\int \prod_{v=1}^{V} \phi_{k,v}^{n_{k,v}+\beta v-1} \, d\phi_k = B(n_k + \beta) \tag{20}$$

The integrating out of ϕ finally results in the quotient of two beta functions, where $n_k = \{n_{k,v}\}_{v=1}^{V}$ is the k^{th} row of the N^{KV} count matrix and β is the pseudo counts.

The *second term* $p(z|\alpha)$ can be derived analogously to the first term by integrating out θ and plugging in the Multinomial distribution specified in Equation (6) for $p(z|\theta)$ and the Dirichlet distribution specified in Equation (5) for $p(\theta|\alpha)$ in Chapter 4.3.3. This yields:

$$\begin{aligned}
p(z|\alpha) &= \int p(z|\theta) \cdot p(\theta|\alpha) \, d\theta \\
&= \int \prod_{d=1}^{D} \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}} \cdot \prod_{d=1}^{D} \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1} \, d\theta_d \\
&= \int \prod_{d=1}^{D} \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}+\alpha_k - 1} \, d\theta_d \\
&= \prod_{d=1}^{D} \frac{1}{B(\alpha)} \int \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}+\alpha_k - 1} \, d\theta_d
\end{aligned}$$

$$= \prod_{d=1}^{D} \frac{B(n_d + \alpha)}{B(\alpha)} \tag{21}$$

where $n_d = \{n_{d,k}\}_{k=1}^{K}$ is the d^{th} row of the N^{DK} count matrix and α is the pseudo counts.

Finally, *combining both terms* in Equation (19) and Equation (21), results in the joint distribution:

$$p(w, z|\alpha, \beta) = p(w|z, \beta) \cdot p(z|\alpha)$$

$$= \prod_{k=1}^{K} \frac{B(n_k + \beta)}{B(\beta)} \cdot \prod_{d=1}^{D} \frac{B(n_d + \alpha)}{B(\alpha)} \tag{22}$$

A. 2.2. Full Conditional

Now that the joint distribution is defined, the full conditional $p(z_i|z_{-i}, w, \alpha, \beta)$ can be derived. z_i represents the topic assignment of word i and z_{-i} refers to the topic assignments of all other words (*except* for z_i). The index "$-i$" thus always means that the i^{th} word is omitted from the calculation. Making use of the Bayes rule and $w = \{w_i, w_{-i}\}$ and $z = \{z_i, z_{-i}\}$, the full conditional becomes:

$$p(z_i|z_{-i}, w, \alpha, \beta) = \frac{p(w, z_i, z_{-i}|\alpha, \beta)}{p(w, z_{-i}|\alpha, \beta)} = \frac{p(w, z|\alpha, \beta)}{p(w, z_{-i}|\alpha, \beta)}$$

$$= \frac{p(w, z|\alpha, \beta)}{p(w_i, w_{-i}, z_{-i}|\alpha, \beta)}$$

$$= \frac{p(w, z|\alpha, \beta)}{p(w_{-i}, z_{-i}|\alpha, \beta)p(w_i|\alpha, \beta)}$$

$$\propto \frac{p(w, z|\alpha, \beta)}{p(w_{-i}, z_{-i}|\alpha, \beta)} \tag{23}$$

The result displays the quotient of two joint distributions, where the numerator $p(w, z|\alpha, \beta) = p(w|z, \beta) \cdot p(z|\alpha)$ is defined in Equation (22) and the denominator $p(w_{-i}, z_{-i}|\alpha, \beta) = p(w_{-i}|z_{-i}, \beta) \cdot p(z_{-i}|\alpha)$ has a similar form but uses the reduced counts $n_k^{(-i)}$ and $n_d^{(-i)}$. Thus, $p(w_{-i}, z_{-i}|\alpha, \beta) = \prod_{k=1}^{K} \frac{B(n_k^{(-i)}+\beta)}{B(\beta)} \cdot \prod_{d=1}^{D} \frac{B(n_d^{(-i)}+\alpha)}{B(\alpha)}$. Plugging in those terms yields:

$$
\begin{aligned}
&= \frac{p(w|z, \beta) \cdot p(z|\alpha)}{p(w_{-i}|z_{-i}, \beta) \cdot p(z_{-i}|\alpha)} \\
&= \frac{\prod_{k=1}^{K} \frac{B(n_k+\beta)}{B(\beta)} \cdot \prod_{d=1}^{D} \frac{B(n_d+\alpha)}{B(\alpha)}}{\prod_{k=1}^{K} \frac{B(n_k^{(-i)}+\beta)}{B(\beta)} \cdot \prod_{d=1}^{D} \frac{B(n_d^{(-i)}+\alpha)}{B(\alpha)}} \\
&= \prod_{k=1}^{K} \frac{B(n_k+\beta)}{B(n_k^{(-i)}+\beta)} \cdot \prod_{d=1}^{D} \frac{B(n_d+\alpha)}{B(n_d^{(-i)}+\alpha)} \\
&= \frac{B(n_{k=1}+\beta) \cdot B(n_{k=2}+\beta) \cdot ... \cdot B(n_{k=K}+\beta)}{B(n_{k=1}^{(-i)}+\beta) \cdot B(n_{k=2}^{(-i)}+\beta) \cdot ... \cdot B(n_{k=K}^{(-i)}+\beta)} \\
&\quad \cdot \quad \frac{B(n_{d=1}+\alpha) \cdot B(n_{d=2}+\alpha) \cdot ... \cdot B(n_{d=D}+\alpha)}{B(n_{d=1}^{(-i)}+\alpha) \cdot B(n_{d=2}^{(-i)}+\alpha) \cdot ... \cdot B(n_{d=D}^{(-i)}+\alpha)} \\
&= \frac{B(n_k+\beta)}{B(n_k^{(-i)}+\beta)} \cdot \frac{B(n_d+\alpha)}{B(n_d^{(-i)}+\alpha)} \qquad (24)
\end{aligned}
$$

The beta function can be expressed in terms of a gamma function (compare Appendix A. 3). Hence, $B(n_k+\beta) = \frac{\prod_{v=1}^{V} \Gamma(n_{k,v}+\beta_v)}{\Gamma(\sum_{v=1}^{V} n_{k,v}+\beta_v)}$ and $B(n_d+\alpha) = \frac{\prod_{k=1}^{K} \Gamma(n_{d,k}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} n_{d,k}+\alpha_k)}$. This works similarly for $B(n_k^{(-i)}+\beta) = \frac{\prod_{v=1}^{V} \Gamma(n_{k,v}^{(-i)}+\beta_v)}{\Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)}+\beta_v)}$ and $B(n_d^{(-i)}+\alpha) = \frac{\prod_{k=1}^{K} \Gamma(n_{d,k}^{(-i)}+\alpha_k)}{\Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)}+\alpha_k)}$, where only the index "$-i$" needs to be added. Replacing the beta functions by gamma functions results in:

$$= \frac{\frac{\prod_{v=1}^{V} \Gamma(n_{k,v} + \beta_v)}{\Gamma(\sum_{v=1}^{V} n_{k,v} + \beta_v)}}{\frac{\prod_{v=1}^{V} \Gamma(n_{k,v}^{(-i)} + \beta_v)}{\Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}} \cdot \frac{\frac{\prod_{k=1}^{K} \Gamma(n_{d,k} + \alpha_k)}{\Gamma(\sum_{k=1}^{K} n_{d,k} + \alpha_k)}}{\frac{\prod_{k=1}^{K} \Gamma(n_{d,k}^{(-i)} + \alpha_k)}{\Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}}$$

$$= \frac{\prod_{v=1}^{V} \Gamma(n_{k,v} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}{\Gamma(\sum_{v=1}^{V} n_{k,v} + \beta_v) \cdot \prod_{v=1}^{V} \Gamma(n_{k,v}^{(-i)} + \beta_v)} \cdot \frac{\prod_{k=1}^{K} \Gamma(n_{d,k} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}{\Gamma(\sum_{k=1}^{K} n_{d,k} + \alpha_k) \cdot \prod_{k=1}^{K} \Gamma(n_{d,k}^{(-i)} + \alpha_k)}$$

$$= \frac{\prod_{v=1}^{V} \Gamma(n_{k,v} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}{\prod_{v=1}^{V} \Gamma(n_{k,v}^{(-i)} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v} + \beta_v)} \cdot \frac{\prod_{k=1}^{K} \Gamma(n_{d,k} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}{\prod_{k=1}^{K} \Gamma(n_{d,k}^{(-i)} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k} + \alpha_k)}$$

$$= \frac{\Gamma(n_{k,v=1} + \beta_v) \cdot \Gamma(n_{k,v=2} + \beta_v) \cdot ... \cdot \Gamma(n_{k,v=V} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}{\Gamma(n_{k,v=1}^{(-i)} + \beta_v) \cdot \Gamma(n_{k,v=2}^{(-i)} + \beta_v) \cdot ... \cdot \Gamma(n_{k,v=V}^{(-i)} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v} + \beta_v)}$$

$$\cdot \quad \frac{\Gamma(n_{d,k=1} + \alpha_k) \cdot \Gamma(n_{d,k=2} + \alpha_k) \cdot ... \cdot \Gamma(n_{d,k=K} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}{\Gamma(n_{d,k=1}^{(-i)} + \alpha_k) \cdot \Gamma(n_{d,k=2}^{(-i)} + \alpha_k) \cdot ... \cdot \Gamma(n_{d,k=K}^{(-i)} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k} + \alpha_k)}$$

$$= \frac{\Gamma(n_{k,v} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}{\Gamma(n_{k,v}^{(-i)} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v} + \beta_v)} \cdot \frac{\Gamma(n_{d,k} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}{\Gamma(n_{d,k}^{(-i)} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k} + \alpha_k)} \tag{25}$$

To simplify the last expression above, the recursive property $\Gamma(x+1) = x\,\Gamma(x)$ of the

gamma function is used. Furthermore, note that $n_{k,v} = n_{k,v}^{(-i)} + 1$ and $n_{d,k} = n_{d,k}^{(-i)} + 1$. Exemplary for the first gamma term, it applies that $\Gamma(n_{k,v} + \beta_v) = \Gamma(n_{k,v}^{(-i)} + \beta_v + 1) = (n_{k,v}^{(-i)} + \beta_v)\Gamma(n_{k,v}^{(-i)} + \beta_v)$. Using the same logic for all other terms yields the final expression of the full conditional:

$$
\begin{aligned}
&= \frac{\Gamma(n_{k,v}^{(-i)} + \beta_v + 1) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}{\Gamma(n_{k,v}^{(-i)} + \beta_v) \cdot \Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v + 1)} \cdot \frac{\Gamma(n_{d,k}^{(-i)} + \alpha_k + 1) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}{\Gamma(n_{d,k}^{(-i)} + \alpha_k) \cdot \Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k + 1)} \\
&= \frac{(n_{k,v}^{(-i)} + \beta_v)\cancel{\Gamma(n_{k,v}^{(-i)} + \beta_v)} \cdot \cancel{\Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}}{\cancel{\Gamma(n_{k,v}^{(-i)} + \beta_v)} \cdot (\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)\cancel{\Gamma(\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v)}} \\
&\quad \cdot \frac{(n_{d,k}^{(-i)} + \alpha_k)\cancel{\Gamma(n_{d,k}^{(-i)} + \alpha_k)} \cdot \cancel{\Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}}{\cancel{\Gamma(n_{d,k}^{(-i)} + \alpha_k)} \cdot (\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)\cancel{\Gamma(\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k)}} \\
&= \frac{n_{k,v}^{(-i)} + \beta_v}{\sum_{v=1}^{V} n_{k,v}^{(-i)} + \beta_v} \cdot \frac{n_{d,k}^{(-i)} + \alpha_k}{\sum_{k=1}^{K} n_{d,k}^{(-i)} + \alpha_k} \\
&= \frac{n_{k,v}^{(-i)} + \beta_v}{\sum_{v=1}^{V} (n_{k,v} + \beta_v) - 1} \cdot \frac{n_{d,k}^{(-i)} + \alpha_k}{\sum_{k=1}^{K} (n_{d,k} + \alpha_k) - 1}
\end{aligned}
\tag{26}
$$

The final result in Equation (26) shows that the full conditional $p(z_i|z_{-i}, w, \alpha, \beta)$ only requires calculating the topic-word count matrix N^{KV} and the document-topic count matrix N^{DK}. The counts $n_{k,v}$ and $n_{d,k}$ come directly from N^{KV} and N^{DK} and the counts $n_{k,v}^{(-i)}$ and $n_{d,k}^{(-i)}$ come from the decremented count matrices N_{-i}^{KV} and N_{-i}^{DK}.

A. 2.3. Obtaining ϕ and θ

In a last step, the parameters ϕ and θ can be computed through a Dirichlet update. The following derivations illustrate this:

The *parameter* ϕ is calculated by $p(\phi|w,z,\beta) = \frac{p(w|\phi,z)\cdot p(\phi|\beta)}{p(w|z,\beta)}$ where the numerator is the product of a Multinomial distribution and a Dirichlet distribution as specified in Equation (7) $p(w|\phi,z) = \prod_{k=1}^{K}\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}}$ and Equation (4) $p(\phi|\beta) = \prod_{k=1}^{K}\frac{1}{B(\beta)}\prod_{v=1}^{V}\phi_{k,v}^{\beta_v-1}$ in Chapter 4.3.3. The denominator was derived in Equation (19) $p(w|z,\beta) = \prod_{k=1}^{K}\frac{B(n_k+\beta)}{B(\beta)}$. Plugging in all these terms leads to:

$$
\begin{aligned}
p(\phi|w,z,\beta) &= \frac{p(w|\phi,z)\cdot p(\phi|\beta)}{p(w|z,\beta)} \\
&= \frac{\prod_{k=1}^{K}\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}}\cdot\prod_{k=1}^{K}\frac{1}{B(\beta)}\prod_{v=1}^{V}\phi_{k,v}^{\beta_v-1}}{\prod_{k=1}^{K}\frac{B(n_k+\beta)}{B(\beta)}} \\
&= \prod_{k=1}^{K}\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}}\cdot\prod_{k=1}^{K}\frac{1}{B(\beta)}\prod_{v=1}^{V}\phi_{k,v}^{\beta_v-1}\cdot\prod_{k=1}^{K}\frac{B(\beta)}{B(n_k+\beta)} \\
&= \prod_{k=1}^{K}\frac{1}{B(\beta)}\frac{B(\beta)}{B(n_k+\beta)}\cdot\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}+\beta_v-1} \\
&= \prod_{k=1}^{K}\frac{1}{B(n_k+\beta)}\cdot\prod_{v=1}^{V}\phi_{k,v}^{n_{k,v}+\beta_v-1} \\
&= Dir(\phi_k|n_k+\beta)\ for\ each\ topic\ k=1,\ ...,\ K
\end{aligned}
\tag{27}
$$

The procedure is analogous for obtaining *parameter* θ. Therefore, $p(\theta|z,\alpha) = \frac{p(z|\theta)\cdot p(\theta|\alpha)}{p(z|\alpha)}$ needs to be calculated. The numerator is again the product of a Multinomial likelihood and a Dirichlet prior, as stated in Equation (6) $p(z|\theta) = \prod_{d=1}^{D}\prod_{k=1}^{K}\theta_{d,k}^{n_{d,k}}$ and Equation (5) $p(\theta|\alpha) = \prod_{d=1}^{D}\frac{1}{B(\alpha)}\prod_{k=1}^{K}\theta_{d,k}^{\alpha_{k-1}}$ in Chapter 4.3.3. The denominator was derived in

Equation (21) $p(z|\alpha) = \prod_{d=1}^{D} \frac{B(n_d+\alpha)}{B(\alpha)}$. Using all these terms results in:

$$
\begin{aligned}
p(\theta|z,\alpha) &= \frac{p(z|\theta) \cdot p(\theta|\alpha)}{p(z|\alpha)} \\
&= \frac{\prod_{d=1}^{D}\prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}} \cdot \prod_{d=1}^{D} \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1}}{\prod_{d=1}^{D} \frac{B(n_d+\alpha)}{B(\alpha)}} \\
&= \prod_{d=1}^{D}\prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}} \cdot \prod_{d=1}^{D} \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_{d,k}^{\alpha_k - 1} \cdot \prod_{d=1}^{D} \frac{B(\alpha)}{B(n_d+\alpha)} \\
&= \prod_{d=1}^{D} \frac{1}{B(\alpha)} \frac{B(\alpha)}{B(n_d+\alpha)} \cdot \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}+\alpha_k - 1} \\
&= \prod_{d=1}^{D} \frac{1}{B(n_d+\alpha)} \cdot \prod_{k=1}^{K} \theta_{d,k}^{n_{d,k}+\alpha_k - 1} \\
&= Dir(\theta_d|n_d+\alpha) \; for \; each \; document \; d = 1, \; ..., \; D
\end{aligned} \tag{28}
$$

As an alternative, the expectation (=mean) of the Dirichlet distribution $\mathbb{E}[x_k] = \frac{\alpha_k}{\sum_{k=1}^{K} \alpha_k}$ can be used to obtain ϕ and θ. This yields the following expressions:

$$\phi_{k,v} = \frac{n_{k,v} + \beta_v}{\sum_{v=1}^{V} (n_{k,v} + \beta_v)} \tag{29}$$

$$\theta_{d,k} = \frac{n_{d,k} + \alpha_k}{\sum_{k=1}^{K} (n_{d,k} + \alpha_k)} \tag{30}$$

A. 2.4. R Code

```
#######################################################################
## LATENT DIRICHLET ALLOCATION (LDA):
## Collapsed Gibbs Sampling
#######################################################################
#######################################################################
##  This code comes from the PhD seminar "Topic models" held
##  by Prof. Joachim Bueschken in the Summer Semester 2016
##  and is adapted to the purposes of this dissertation.
#######################################################################
##-----------------------------------------
###Get Data
###--> here: generated data from A. 1. is used
##-----------------------------------------
rm(list=ls(1)[-c(which(ls(1)=="data"))])

w <- data[,"w"]          ## all words
d <- data[,"d"]          ## document that each words belongs to
W <- length(w)           ## number of all words
V <- length(unique(w))   ## number of unique words
D <- length(unique(d))   ## number of documents

##-----------------------------------------
###1.Initialization
##-----------------------------------------
##Initialize number of topics:
K <- 3

##Initialize number of iterations:
R <- 500

##Initialize hyper-parameters:
alpha = 0.5    ## for theta
beta = 1       ## for phi

##Initialize z:
z = apply(rmultinom(W,1,rep(1/K,K)),2,which.max)

```

```
##Initialize count matrices N_kv and N_dk
##--> given the initialized topic indicators z:
##N_kv (word-topic counts)
N_kv = matrix(0,V,K)
rows = as.numeric(unlist(labels(table(w)))) ## account for all-0 rows
cols <- as.numeric(unlist(labels(table(z)))) ## account for all-0 columns
N_kv[rows,cols] = table(data.table(cbind(w,z)))
##N_dk (topic-document counts)
N_dk = matrix(0,K,D)
rows <- as.numeric(unlist(labels(table(z)))) ## account for all-0 rows
cols = as.numeric(unlist(labels(table(d)))) ## account for all-0 columns
N_dk[rows,cols] = table(data.table(cbind(z,d)))

##Initialize matrices for saving phi and theta (for all iterations):
Phi_draw = array(NA,c(V,K,R))
Theta_draw = array(NA,c(K,D,R))

##Time tracking
starttime = proc.time()[3]

##-----------------------------------------
###2.Gibbs Sampling
##-----------------------------------------
for (rep in 1:R){

  for(i in 1:W){

    ##-----------------------------------------
    ###2.a Decrement count matrices N_kv and N_dk by 1 for the entries of the
    ###    current topic assignment z[i]
    ##-----------------------------------------
    N_kv[w[i],z[i]] = N_kv[w[i],z[i]] - 1
    N_dk[z[i],d[i]] = N_dk[z[i],d[i]] - 1

    ##-----------------------------------------
    ###2.b Sample new topic
    ##-----------------------------------------
    pt = (N_kv[w[i],]+beta)/colSums(N_kv+beta) * (N_dk[,d[i]]+alpha)/rowSums(N_dk+
        alpha)
```

```
    z[i] = which.max(rmultinom(1,1,pt))

    ##----------------------------------------
    ###2.c Increment count matrices N_kv and N_dk by 1 for the entries of the
    ###    new topic assignment z[i]
    ##----------------------------------------
    N_kv[w[i],z[i]] = N_kv[w[i],z[i]] + 1
    N_dk[z[i],d[i]] = N_dk[z[i],d[i]] + 1

  } ## end of W-loop (words)

  ##Output in console
  print(paste("Topic:", K, ";", "Iteration:", rep))

  ##----------------------------------------
  ###3.Burn-in
  ##----------------------------------------
  ###Only calculate parameters for last 100 draws:
  if(rep>400){

  ##----------------------------------------
  ###4.Obtaining phi and theta
  ##----------------------------------------
    ##----------------------------------------
    ###4.a ... with Dirichlet draws
    ##----------------------------------------
    phi = matrix(NA,V,K)
    theta = matrix(NA,K,D)
    ##Draw of phi_k
    for(k in 1:K){phi[,k]=rdirichlet(N_kv[,k]+rep(beta,V))}
    ##Draw of theta_a
    for(d in 1:D){theta[,d] = rdirichlet(N_dk[,d]+rep(alpha,K))}

    ##----------------------------------------
    ###4.b ... with the expectation of the Dirichlet distribution
    ##----------------------------------------
    #exp.phi = matrix(NA,V,K)
    #exp.theta = matrix(NA,K,D)
    #for(k in 1:K){exp.phi[,k] = (N_kv[,k]+beta)/sum(N_kv[,k]+beta)}
```

```
115     #for(d in 1:D){exp.theta[,d] = (N_dk[,d]+alpha)/sum(N_dk[,d]+alpha)}
116
117     ##-----------------------------------------
118     ###4.c ... saving phi and theta
119     ##-----------------------------------------
120     Phi_draw[,,rep] <- phi
121     Theta_draw[,,rep] <- theta
122     #Phi_draw[,,rep] <- exp.phi
123     #Theta_draw[,,rep] <- exp.theta
124
125   } ## end of if(rep>400) (burn-in)
126
127   ##Time tracking and output in console
128   if(rep%%10 == 0){ ## time and output are updated after every 10^th iteration
129     endtime = proc.time()[3]
130     timetoend = (endtime-starttime)
131     print(paste(paste("Topic:",K, sep=""), ";", paste("Iteration:",rep, sep=""),
132                 ";", paste("Time:", round((timetoend)/60,2), "min", sep="")))
133   }
134
135 } ## end of R-loop (iterations)
```

A. 3. Excursus: Dirichlet and Multinomial Distribution

This excursus has been added to provide a better understanding of the LDA model, which makes use of the Dirichlet distribution and the Multinomial distribution. The properties of both distributions are explained in detail (A. 3.1 and A. 3.2) as well as how both distributions combine in the Dirichlet-Multinomial model (A. 3.3). At the end, the R code (A. 3.4) used for creating the graphics in this excursus is shown.

A. 3.1. The Dirichlet Distribution

The Dirichlet distribution is the multivariate generalization of the beta distribution. It is commonly used in Bayesian statistics as a prior distribution for the Multinomial distribution. The probability density function (PDF) of the Dirichlet distribution is defined as (Murphy 2012, pp. 47-49; Bishop 2006, pp. 76-78):

$$p(\theta|\alpha) = \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \tag{31}$$

$$where \;\; \alpha = (\alpha_1, ..., \alpha_K) \;\; and \;\; \alpha_k > 0$$

$$where \;\; K \geq 2 \;\; number\, of\, categories$$

$$where \;\; B(\alpha) = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}$$

$$where \;\; \theta = (\theta_1, ..., \theta_K) \;\; and \;\; \theta_k \in [0, 1] \;\; , \;\; \sum_{k=1}^{K} \theta_k = 1$$

The parameters of the model are the *concentration parameters* $\alpha = (\alpha_1, ..., \alpha_K)$, defined for positive values only, and the *number of categories* K, with a minimum of two categories. In the case of $K = 2$ categories, the Dirichlet distribution equals a beta distribution. The *multivariate beta function* $B(\alpha)$ is the normalizing constant and can be expressed in terms of a gamma function as shown in the equation above. The entries

of the *vector* $\theta = (\theta_1, ..., \theta_K)$ are real numbers in the interval [0,1] and need to sum to 1. They are the probabilities for each category K.

Figure 22 displays several Dirichlet density plots with different concentration parameters α for $K = 3$ categories. Two aspects are worth mentioning. First, there are asymmetric and symmetric distributions. An *asymmetric Dirichlet* is a Dirichlet where the parameter components of α have different values, e.g., $\alpha_1 = 2$, $\alpha_2 = 5$, $\alpha_3 = 4$ in the upper-left panel. The distribution is asymmetric because it is not centered in the middle of the simplex. A *symmetric Dirichlet* is a Dirichlet where each parameter component of α has the same values, e.g., $\alpha_1 = 2$, $\alpha_2 = 2$, $\alpha_3 = 2$ in the middle-left panel. Those distributions are always centered in the middle of the simplex. A special case comprises $\alpha_1 = 1$, $\alpha_2 = 1$, $\alpha_3 = 1$ in the upper-right panel; it is called a *flat (symmetric) Dirichlet*. The Dirichlet distribution is then equivalent to a uniform distribution in the simplex. Second, values of α higher and lower to 1 lead to different shapes of the Dirichlet distribution. When $\alpha > 1$ (see middle panels), the distribution is bent upward and the density is concentrated in the center. With increasing α (moving from the middle-left to the middle-right panel), more density is pushed to the center, which makes the distribution peakier. When $0 < \alpha < 1$ (see lower panels), the distribution is bent downward and the density is crammed up against the edges of the simplex. The closer α gets to zero (moving from the lower-left to the lower-right panel), the more density is pushed to the corners.

Figure 23 shows sample draws from different symmetric Dirichlet distributions with $K = 10$ categories. Each row corresponds to another Dirichlet distribution with varying concentration parameter α. The selected value is indicated in the labels above the panels. From each of the distinct Dirichlet distributions 5 samples are drawn. The results show that when sampling from Dirichlets with α much greater than 1, it yields almost uniform distributions, i.e., all the values within a single sample are similar to each other

(see upper two panels). Samples from Dirichlets with α less than 1 (see lower two panels) generate sparse distributions ("sparsity"). This means that very few categories have extremely high probabilities and the remaining categories' probabilities are close to zero.

The LDA model has two Dirichlet variables: the topic distribution θ_d and the word distribution ϕ_k, which are drawn from a Dirichlet distribution with concentration parameter α respectively β. Steyvers and Griffiths (2007) and Griffiths and Steyvers (2004) discuss the selection of those concentration parameters more in detail:

- The *choice of* α determines the concentration of topics in a document. If a document should be dominated by very few topics, a low value for α needs to be chosen. In return, a high value for α leads to more equal topic probabilities and thus more topics for each document.

- The *choice of* β determines the concentration of words in a topic and the required number of topics. A low β value results in topics that are strongly determined by very few words. This means that topics are more unique and address specific topics. To cover all themes present in large text collections, the number of topics is therefore typically higher. Conversely, higher values for β yield more equal word distributions for a topic and make the topics more generic and similar to each other. The required number of topics decreases.

In the LDA model, it is most common to use symmetric Dirichlet priors. Steyvers and Griffiths (2007, pp. 431-433) have had good experiences with the symmetric priors $\alpha = 50/K$ and $\beta = 0.01$ for many different text collections. However, for superior performance, some literature (Wallach, Mimno, and McCallum 2009; Syed and Spruit 2018) also suggests asymmetric priors, especially over the per-document topic distributions θ_d. All in all, appropriate concentration parameters should be selected with care as their choice can have tremendous implications on the model results.

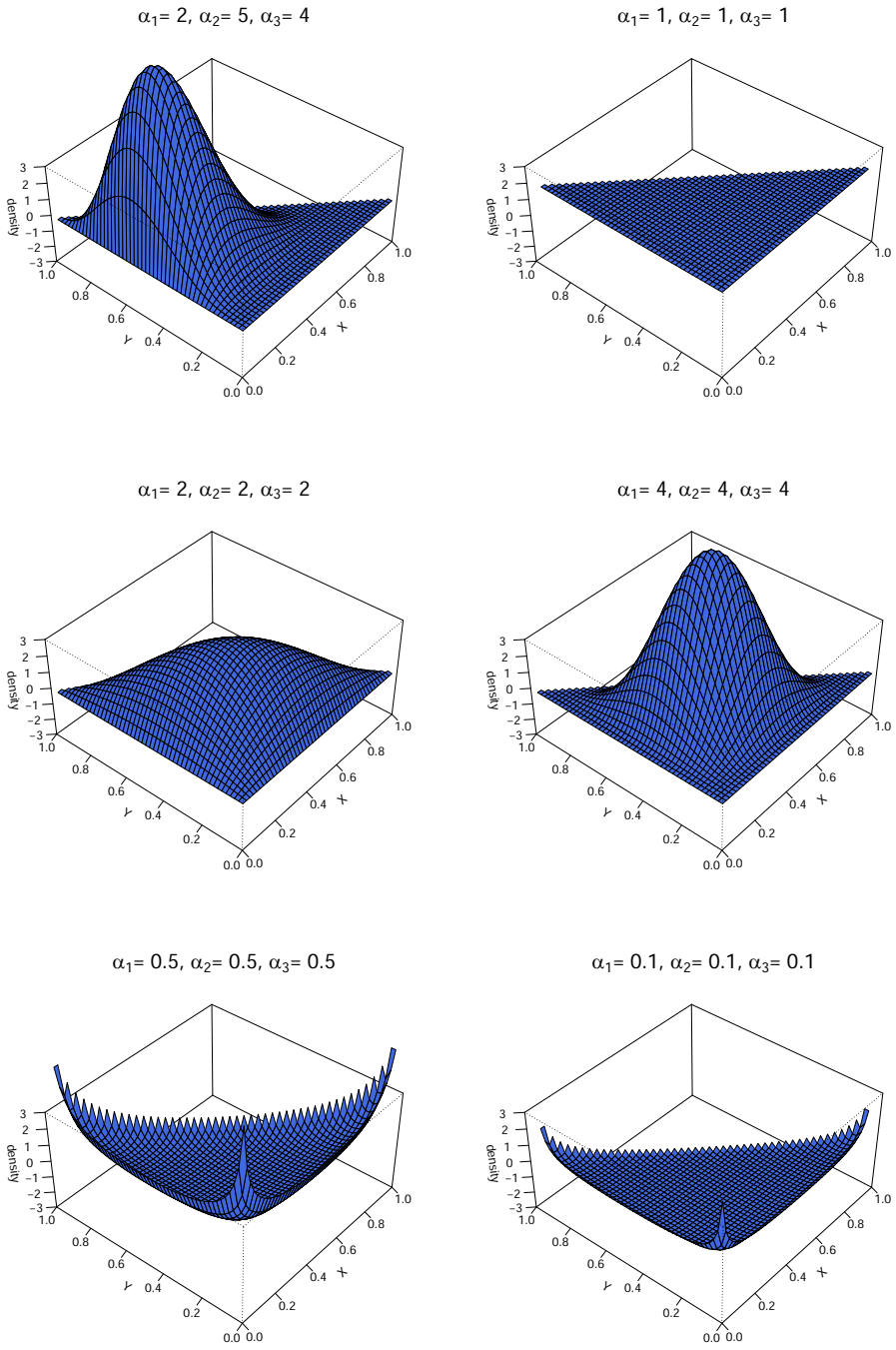


Figure 22: Dirichlet PDFs with Different Concentation Parameters[14]

[14]The plots were created in R. The code that generated the graphics is displayed in Appendix A. 3.4.1.

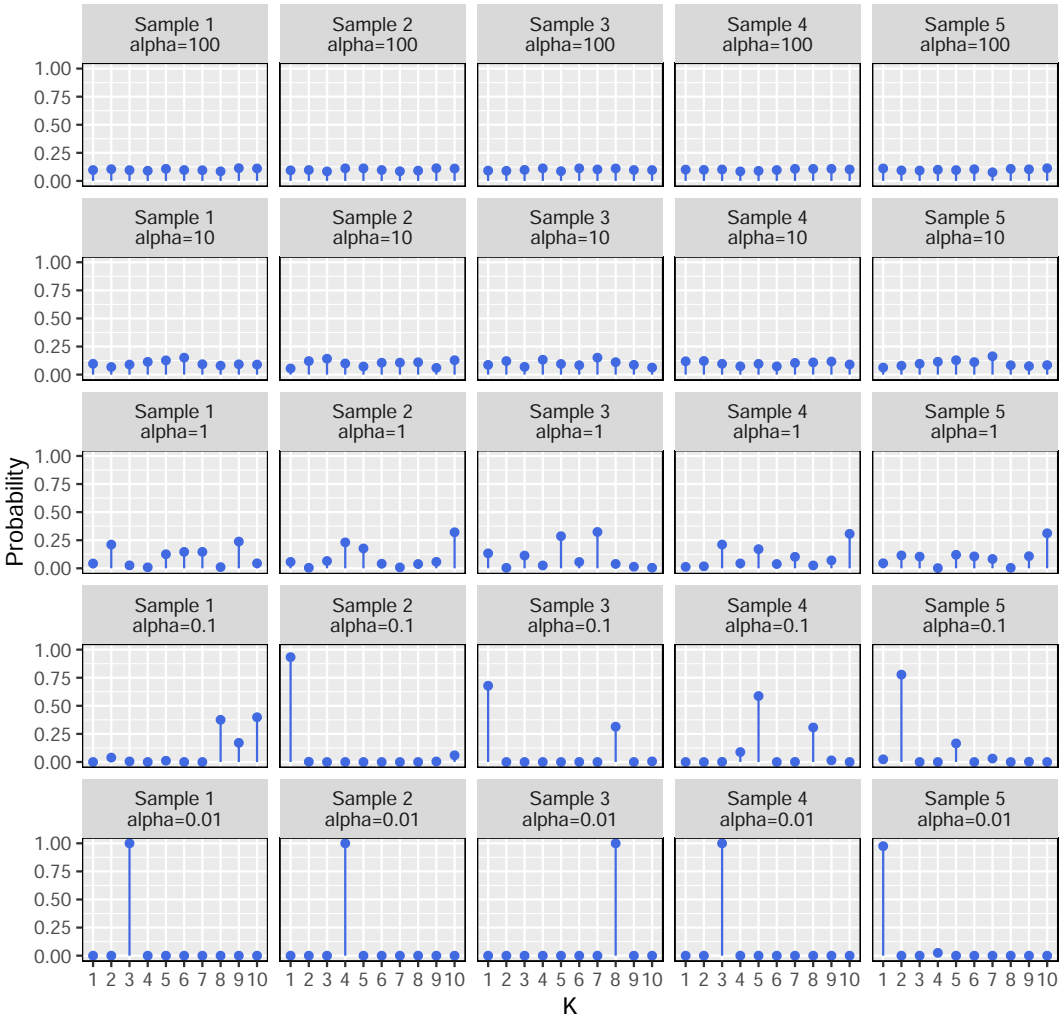


Figure 23: Draws from Different Dirichlet Distributions[15]

[15]The plots were created in R following Blei's Topic Models tutorial at the Machine Learning Summer School (MLSS) in Cambridge 2009 (`http://videolectures.net/mlss09uk_blei_tm/`). The code that generated the graphics is displayed in Appendix A. 3.4.2.

A. 3.2. The Multinomial Distribution

The Multinomial distribution is the multivariate generalization of the binomial distribution. It is used to model K outcomes such as the tossing of a *K-sided* die n times or the extraction of n balls from an urn containing balls of k different colors. The n trials are independent, i.e., a new trial does not depend on the previous trial. For the urn example, this means that the extracted ball is replaced after each draw ("drawing with replacement"). The Multinomial distribution is commonly used as the likelihood in Bayesian statistics. Its probability mass function (PMF) is defined as (Murphy 2012, pp. 35-36; Bishop 2006, pp. 74-76):

$$p(x|n,\theta) = \binom{n}{x_1 \ ... \ x_K} \prod_{k=1}^{K} \theta_k^{x_k} \tag{32}$$

$$\begin{aligned} &where \quad n > 0 \ \ number\ of\ trials \\ &where \quad \theta = (\theta_1, ..., \theta_K) \ \ and \ \ \theta_k \in [0,1] \ \ , \ \ \sum_{k=1}^{K} \theta_k = 1 \end{aligned}$$

$$\begin{aligned} &where \quad x = (x_1, ..., x_K) \ \ and \ \ x_k \in \{0, ..., n\} \ \ , \ \ \sum_{k=1}^{K} x_k = n \\ &where \quad \binom{n}{x_1 \ ... \ x_K} = \frac{n!}{x_1! \cdots x_K!} \end{aligned}$$

The parameters of the Multinomial distribution are θ and n. The *vector* $\theta = (\theta_1, ..., \theta_K)$ contains the *event probabilities*, i.e., the probabilities that each outcome K occurs. They take values in the interval [0,1] and sum to 1. n is the *number of trials*, which needs to be a minimum of 1. The *vector* $x = (x_1, ..., x_K)$ counts the *number of times an outcome k occurs*. Its entries are integer values in the range of 0 *to* n and their total sum equals the number of trials n. $\binom{n}{x_1 \ ... \ x_K}$ is the *multinomial coefficient*, which gives the number of distinguishable permutations for the n trials. The *number of possible outcomes* is denoted by K.

A special case of the Multinomial distribution is when there is only *one* single trial (n=1). This would be the toss of a *K-sided* die *once* or the extraction of *one* ball from an urn containing balls of k different colors. The distribution is then called the discrete or categorical distribution. In analogy to the Bernoulli distribution, which also comprises only one single trial, some literature also suggests the name Multinoulli distribution. The PMF simplifies to:

$$p(x|1,\theta) = \prod_{k=1}^{K} \theta_k^{x_k} \tag{33}$$

which is the Multinomial distribution without the multinomial coefficient.

A. 3.3. The Dirichlet-Multinomial Model

The Dirichlet-Multinomial model combines the Dirichlet distribution and the Multinomial distribution from the two previous chapters and is frequently used in Bayesian statistics for posterior computation. The model greatly simplifies calculation because of the conjugacy between the both distributions. In a Bayesian setting where the Dirichlet-Multinomial model is used, the likelihood $p(D|\theta)$ always has the form of a Multinomial distribution and the prior $p(\theta)$ the form of a Dirichlet distribution. When multiplying the likelihood with the prior, both distributions combine so that the posterior $p(\theta|D)$ has the same form as the prior, namely a Dirichlet distribution. This is called *conjugacy*.

The following calculations illustrate the Dirichlet-Multinomial model by plugging in the formula for the Dirichlet distribution, Equation (31), and Multinomial distribution, Equation (32), (Murphy 2012, pp. 78-82):

$$p(\theta|D) = \frac{p(D|\theta) \cdot p(\theta)}{p(D)}$$

$$\propto p(D|\theta) \cdot p(\theta)$$

$$= \binom{n}{x_1 \; ... \; x_K} \prod_{k=1}^{K} \theta_k^{x_k} \cdot \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}$$

$$\propto \prod_{k=1}^{K} \theta_k^{x_k} \cdot \theta_k^{\alpha_k - 1} = \prod_{k=1}^{K} \theta_k^{x_k + \alpha_k - 1}$$

$$= Dir(\theta|x_1 + \alpha_1, ..., x_K + \alpha_K) = Dir(\theta|x + \alpha) \tag{34}$$

A. 3.4. R Code

A. 3.4.1. R Code for Dirichlet PDFs

```
######################################################################
##  DIRICHLET DISTRIBUTION:
##  Plotting different Probability Density Functions (PDFs)
######################################################################
######################################################################
##  This code comes from
##  https://www.math.wustl.edu/~victor/classes/ma322/r-eg-35.txt
##  and has been modified for the purposes of this dissertation
##
##  Other interesting resources:
##  https://cswiki.cs.byu.edu/cs-401r/plotting-a-dirichlet
######################################################################
rm(list=ls())
##-----------------------------------------------------------##
## EXPLANATION
##-----------------------------------------------------------##
# The Dirichlet density is not implemented in base R.
# It is necessary to get a contributed package.
# Several are available, here are two:

#install.packages('gtools')
require('gtools')

#install.packages('MCMCpack')
require('MCMCpack')

# ...provides functions
# ddirichlet(x,alpha)   # Density function at (x1,...,xk)
# rdirichlet(n,alpha)   # n random samples

###### Dirichlet density on 3 variables x=(x1,x2,x3),
######  with x1+x2+x3=1, and shape parameters a=(a1,a2,a3)
#
# Generate a grid of x1,x2 values:
x1 <- seq(0,1, by=0.01)
```

```
x2 <- seq(0,1, by=0.01)

# Initialize a matrix to hold the pdf value:
z <- matrix(0, nrow=length(x1), ncol=length(x2))

# Choose some reasonable shape parameters:
alpha <- c(3,5,12)

# Fill z by looping over all valid (x1,x2) pairs, putting in
# x3=1-x1-x2 as the third variable in ddirichlet():
for(i in 1:length(x1)) {
  for(j in 1:length(x2) ) {
    if( x1[i]+x2[j] < 1) {
      x <- c(x1[i], x2[j], 1-x1[i]-x2[j])  # so x1+x2+x3=1
      z[i,j] <- ddirichlet(x,alpha);
    } else {
      z[i,j] <- NA
    }
  }
}
# Comment: It is the property of the Dirichlet distribution,
# that the probabilities in the simplex always sum up to one.
# That's also why x1+x2+x3=1.

# View the results as a perpective plot:
persp(x1,x2,z)

## To learn more about the different arguments of the function 'persp()', go to:
## https://www.rdocumentation.org/packages/graphics/versions/3.5.2/topics/persp

# View the results as a contour plot:
# contour(x1,x2,z)

##-----------------------------------------------------------##
## FUNCTION
##-----------------------------------------------------------##
dirichletPDF_3D <- function(alpha1=2, alpha2=2, alpha3=2, granuarity=0.02) {

  alpha <- c(alpha1,alpha2,alpha3)
```

```
  x1 <- seq(0,1, by=granuarity)
  x2 <- seq(0,1, by=granuarity)
  z <- matrix(0, nrow=length(x1), ncol=length(x2))
  # Fill z by looping over all valid (x1,x2) pairs, putting in
  # x3=1-x1-x2 as the third variable in ddirichlet():
  for(i in 1:length(x1)) {
    for(j in 1:length(x2) ) {
      if( x1[i]+x2[j] < 1) {
        x <- c(x1[i], x2[j], 1-x1[i]-x2[j])  # so x1+x2+x3=1
        z[i,j] <- ddirichlet(x,alpha);
      } else {
        z[i,j] <- NA
      }
    }
  }

  persp(x1,x2,z, col = "royalblue",
        main=substitute(
          paste(alpha[1], "= ", n1, ", ", alpha[2], "= ", n2, ", ",
                alpha[3], "= ", n3), list(n1=alpha[1],n2=alpha[2],n3=alpha[3])),
        cex.main=1.8,
        zlim=c(-3,3), theta=310,phi=40, expand=0.5,
        axes=TRUE, ticktype="detailed",d=4,
        cex.axis=1.1,cex.lab=1.1,shade=0.01,
        xlab="X", ylab="Y", zlab="density", border="black")

}

##-------------------------------------------------------##
## GET PLOTS
##-------------------------------------------------------##
dirichletPDF_3D(2,5,4)
dirichletPDF_3D(1,1,1)

dirichletPDF_3D(2,2,2)
dirichletPDF_3D(4,4,4)

dirichletPDF_3D(0.5,0.5,0.5)
dirichletPDF_3D(0.1,0.1,0.1)
```

```
114
115 ##-------------------------------------------------------------##
116 ## SAVE PLOTS
117 ##-------------------------------------------------------------##
118 setwd("C:/Users/wwa732/Desktop/Diss_R-Code")
119 pdf("dirichletPDF1.pdf")
120 par(mfrow=c(1,1))
121 dirichletPDF_3D(2,5,4)
122 dev.off()
123 pdf("dirichletPDF2.pdf")
124 par(mfrow=c(1,1))
125 dirichletPDF_3D(1,1,1)
126 dev.off()
127 pdf("dirichletPDF3.pdf")
128 par(mfrow=c(1,1))
129 dirichletPDF_3D(2,2,2)
130 dev.off()
131 pdf("dirichletPDF4.pdf")
132 par(mfrow=c(1,1))
133 dirichletPDF_3D(4,4,4)
134 dev.off()
135 pdf("dirichletPDF5.pdf")
136 par(mfrow=c(1,1))
137 dirichletPDF_3D(0.5,0.5,0.5)
138 dev.off()
139 pdf("dirichletPDF6.pdf")
140 par(mfrow=c(1,1))
141 dirichletPDF_3D(0.1,0.1,0.1)
142 dev.off()
```

A. 3.4.2. R Code for Dirichlet Draws

```
######################################################################
##  DIRICHLET DISTRIBUTION:
##  Plotting Draws from a Dirichlet with rdirichlet() and ggplot()
######################################################################
######################################################################
##  This code comes from
##  https://stats.stackexchange.com/questions/15198/dirichlet-distribution-plot-in
    -r
##  and has been modified for the purposes of this dissertation
######################################################################
rm(list=ls())
set.seed(9989)
##-------------------------------------------------------##
## Getting Draws from a Dirichlet using rdirichlet()
##-------------------------------------------------------##
# For this task exists the function 'rdirichlet', which is
# available in three different packages:

# 1. 'bayesm':
# install.packages('bayesm')
# library('bayesm')
# require('bayesm')
# --> rdirichlet(alpha)

# Example:
# K <- 5
# alpha <- 1
# rdirichlet(alpha=rep(alpha,K))

# 2. 'gtools'
# install.packages('gtools')
library('gtools')
require('gtools')
# --> ddirichlet(x, alpha)
# --> rdirichlet(n, alpha)

# 3. 'MCMCpack'
```

```
library('MCMCpack')
require('MCMCpack')
# --> ddirichlet(x, alpha)
# --> rdirichlet(n, alpha)

# Example:
# K <- 5
# alpha <- 1
# rdirichlet(n=3, alpha=rep(alpha,K))
# rdirichlet(n=2, alpha=c(1,1,1))

# Caution: be aware that, depending on which package is used,
# the arguments are different. The following code requires the 'rdirichlet'
# function from the 'gtools' and 'MCMCpack' package.

#######  Draws from a Dirichlet distribution
#######  with different concentration parameters:
# alpha=100
x100 <- rdirichlet(n=5, alpha=rep(100, 10))
# alpha=10
x10 <- rdirichlet(n=5, alpha=rep(10, 10))
# alpha=1
x1 <- rdirichlet(n=5, alpha=rep(1, 10))
# alpha=0.1
x0.1 <- rdirichlet(n=5, alpha=rep(0.1, 10))
# alpha=0.01
x0.01 <- rdirichlet(n=5, alpha=rep(0.01, 10))

# Stack all draws:
x <- rbind(x100,x10,x1,x0.1,x0.01)
# --> This is the data for plotting

##-------------------------------------------------------##
## Plotting Draws with ggplot()
##-------------------------------------------------------##
# install.packages('ggplot2')
library('ggplot2')

# The input data needs to be a 'data.frame':
```

```
dat <- data.frame(item=factor(rep(1:10,25)),
                  draw=factor(rep(1:25,each=10)),
                  value=as.vector(t(x)))

# To define own labels in ggplot() a labeller function with
# two arguments (variable, value) needs to be specified (see below):
# https://stackoverflow.com/questions/3472980/ggplot-how-to-change-facet-labels
# https://github.com/tidyverse/ggplot2/wiki/labeller

LABEL_names <- list(
  '1'="Sample 1\nalpha=100",
  '2'="Sample 2\nalpha=100",
  '3'="Sample 3\nalpha=100",
  '4'="Sample 4\nalpha=100",
  '5'="Sample 5\nalpha=100",
  '6'="Sample 1\nalpha=10",
  '7'="Sample 2\nalpha=10",
  '8'="Sample 3\nalpha=10",
  '9'="Sample 4\nalpha=10",
  '10'="Sample 5\nalpha=10",
  '11'="Sample 1\nalpha=1",
  '12'="Sample 2\nalpha=1",
  '13'="Sample 3\nalpha=1",
  '14'="Sample 4\nalpha=1",
  '15'="Sample 5\nalpha=1",
  '16'="Sample 1\nalpha=0.1",
  '17'="Sample 2\nalpha=0.1",
  '18'="Sample 3\nalpha=0.1",
  '19'="Sample 4\nalpha=0.1",
  '20'="Sample 5\nalpha=0.1",
  '21'="Sample 1\nalpha=0.01",
  '22'="Sample 2\nalpha=0.01",
  '23'="Sample 3\nalpha=0.01",
  '24'="Sample 4\nalpha=0.01",
  '25'="Sample 5\nalpha=0.01"
)

# Labeller Function:
LABEL_labeller <- function(variable,value){
```

```
  return(LABEL_names[value])
}

# To learn more about the different arguments of the function 'ggplot', go to:
# https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
# --> ggplot2 Cheatsheet!

#######  Plot Dirichelt draws with ggplot()
#######  and save as pdf file:
#
setwd("C:/Users/wwa732/Desktop/Diss_R-Code")
pdf("dirichletdraws_plot.pdf")
par(mfrow=c(1,1))

ggplot(dat,aes(x=item,y=value,ymin=0,ymax=value)) +
  geom_point(colour=I("royalblue"))       +
  geom_linerange(colour=I("royalblue"))   +
  facet_wrap(~draw,ncol=5,labeller=LABEL_labeller)          +
  scale_y_continuous(lim=c(0,1))     +
  theme(panel.border = element_rect(fill=0, colour="black")) +
  labs(title="", x="K", y="Probability")

dev.off()
```

B. Appendix for

Empirical Analysis

B. 1. Data

B. 1.1. Data Preparation ALL

	Total	Median	Mean	SD	Min	Max
No. of customers	96,239	-	-	-	-	-
No. of SKUs (total)	264,717	-	-	-	-	-
No. of SKUs (unique)	4,758	-	-	-	-	-
No. of orders	135,456	-	-	-	-	-
No. of orders per customer	-	1	1.41	2.63	1	603
No. of SKUs per customer	-	1	2.75	10.82	1	1,347
No. of SKUs per order	-	1	1.95	3.40	1	148
No. of orders per SKU	-	4	33.37	206.53	1	6,140

Table 9: Descriptive Statistics (ALL)

	Full Data	Training Data	Test Data
No. of customers	96,239	81,531	24,060
No. of SKUs (total)	264,717	240,657	24,060
No. of SKUs (unique)	4,758	4,634	2,315
No. of orders	135,456	118,254	24,060

Table 10: Overview Splitted Data Set (ALL)

Holdout Customers	Number
All Customers	24,060
Cold Customer	14,708
Old Customer, Old SKU	7,002
Old Customer, New SKU	2,350

Table 11: Holdout Customers (ALL)

B. 1.2. Data Preparation USO

	Total	Median	Mean	SD	Min	Max
No. of customers	96,239	-	-	-	-	-
No. of SKUs (total)	158,788	-	-	-	-	-
No. of SKUs (unique)	4,758	-	-	-	-	-
No. of orders	135,456	-	-	-	-	-
No. of orders per customer	-	1	1.41	2.63	1	603
No. of SKUs per customer	-	1	1.65	3.66	1	671
No. of SKUs per order	-	1	1.17	0.63	1	21
No. of orders per SKU	-	4	33.37	206.53	1	6,140

Table 12: Descriptive Statistics (USO)

	Full Data	Training Data	Test Data
No. of customers	96,239	78,057	24,060
No. of SKUs (total)	158,788	134,728	24,060
No. of SKUs (unique)	4,758	4,591	2,315
No. of orders	135,456	113,463	24,060

Table 13: Overview Splitted Data Set (USO)

Holdout Customers	Number
All Customers	24,060
Cold Customer	18,182
Old Customer, Old SKU	2,916
Old Customer, New SKU	2,962

Table 14: Holdout Customers (USO)

B. 1.3. Data Preparation USC

	Total	Median	Mean	SD	Min	Max
No. of customers	96,239	-	-	-	-	-
No. of SKUs (total)	124,043	-	-	-	-	-
No. of SKUs (unique)	4,758	-	-	-	-	-
No. of SKUs per customer	-	1	1.29	1.33	1	178
No. of customers per SKU	-	4	26.07	169.06	1	5,468

Table 15: Descriptive Statistics (USC)

	Full Data	Training Data	Test Data
No. of customers	96,239	75,956	24,060
No. of SKUs (total)	124,043	99,983	24,060
No. of SKUs (unique)	4,758	4,576	2,348

Table 16: Overview Splitted Data Set (USC)

Holdout Customers	Number
All Customers	24,060
Cold Customer	20,283
Old Customer, Old SKU	-
Old Customer, New SKU	3,777

Table 17: Holdout Customers (USC)

B. 1.4. Data Preparation 11ALL

	Total	Median	Mean	SD	Min	Max
No. of customers	37,583	-	-	-	-	-
No. of SKUs (total)	206,061	-	-	-	-	-
No. of SKUs (unique)	4,321	-	-	-	-	-
No. of orders	76,800	-	-	-	-	-
No. of orders per customer	-	1	2.04	4.13	1	603
No. of SKUs per customer	-	3	5.48	16.96	2	1,347
No. of SKUs per order	-	2	2.68	4.37	1	148
No. of orders per SKU	-	4	23.17	112.01	1	3,412

Table 18: Descriptive Statistics (11ALL)

	Full Data	Training Data	Test Data
No. of customers	37,583	37,583	9,396
No. of SKUs (total)	206,061	196,665	9,396
No. of SKUs (unique)	4,321	4,255	1,765
No. of orders	76,800	74,295	9,396

Table 19: Overview Splitted Data Set (11ALL)

Holdout Customers	Number
All Customers	9,396
Cold Customer	-
Old Customer, Old SKU	7,073
Old Customer, New SKU	2,323

Table 20: Holdout Customers (11ALL)

B. 1.5. Data Preparation 11USO

	Total	Median	Mean	SD	Min	Max
No. of customers	24,136	-	-	-	-	-
No. of SKUs (total)	86,685	-	-	-	-	-
No. of SKUs (unique)	4,173	-	-	-	-	-
No. of orders	63,353	-	-	-	-	-
No. of orders per customer	-	2	2.63	5.06	1	603
No. of SKUs per customer	-	2	3.60	6.96	2	671
No. of SKUs per order	-	1	1.37	0.87	1	21
No. of orders per SKU	-	3	20.77	99.03	1	2,930

Table 21: Descriptive Statistics (11USO)

	Full Data	Training Data	Test Data
No. of customers	24,136	24,136	6,034
No. of SKUs (total)	86,685	80,651	6,034
No. of SKUs (unique)	4,173	4,083	1,423
No. of orders	63,353	59,464	6,034

Table 22: Overview Splitted Data Set (11USO)

Holdout Customers	Number
All Customers	6,034
Cold Customer	-
Old Customer, Old SKU	2,958
Old Customer, New SKU	3,076

Table 23: Holdout Customers (11USO)

B. 1.6. Data Preparation 11USC

	Total	Median	Mean	SD	Min	Max
No. of customers	15,365	-	-	-	-	-
No. of SKUs (total)	43,169	-	-	-	-	-
No. of SKUs (unique)	4,102	-	-	-	-	-
No. of SKUs per customer	-	2	2.81	2.88	2	178
No. of customers per SKU	-	3	10.52	36.53	1	969

Table 24: Descriptive Statistics (11USC)

	Full Data	Training Data	Test Data
No. of customers	15,365	15,365	3,841
No. of SKUs (total)	43,169	39,328	3,841
No. of SKUs (unique)	4,102	3,975	1,366

Table 25: Overview Splitted Data Set (11USC)

Holdout Customers	Number
All Customers	3,841
Cold Customer	-
Old Customer, Old SKU	-
Old Customer, New SKU	3,841

Table 26: Holdout Customers (11USC)

B. 2. Hit Rates

B. 2.1. Data Preparation ALL

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	17.66	29.05	35.15	43.36
Old Customer, Old SKU	47.42	66.38	74.08	81.85
Old Customer, New SKU	10.17	17.06	20.43	27.49
Sticky ATM				
All Customers	17.47	28.37	34.11	42.94
Old Customer, Old SKU	47.12	66.12	73.15	80.89
Old Customer, New SKU	10.77	17.53	23.02	31.02
CF				
All Customers	30.57	38.20	43.87	52.67
Old Customer, Old SKU	85.16	91.92	92.93	93.87
Old Customer, New SKU	12.89	20.34	24.04	27.83
Bigram				
All Customers	32.23	39.97	45.08	53.48
Old Customer, Old SKU	89.31	91.91	92.68	93.46
Old Customer, New SKU	17.83	28.48	32.59	37.36
Unigram				
All Customers	4.56	13.19	15.38	23.44
Old Customer, Old SKU	5.06	10.40	14.04	20.22
Old Customer, New SKU	2.30	7.19	8.85	13.28

Table 27: Hit Rates in Percent (ALL)

B. 2.2. Data Preparation USO

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	11.33	21.40	27.72	36.82
Old Customer, Old SKU	54.70	71.40	76.89	83.33
Old Customer, New SKU	11.01	20.53	26.16	33.69
Sticky ATM				
All Customers	11.50	21.01	27.61	36.72
Old Customer, Old SKU	52.88	68.79	74.55	80.69
Old Customer, New SKU	10.84	20.76	25.86	34.81
CF				
All Customers	15.70	23.39	29.69	39.26
Old Customer, Old SKU	77.88	86.15	88.27	89.75
Old Customer, New SKU	12.56	19.89	23.40	27.52
Bigram				
All Customers	15.84	24.61	30.34	39.60
Old Customer, Old SKU	74.62	80.43	82.32	84.05
Old Customer, New SKU	16.83	27.07	31.22	35.84
Unigram				
All Customers	4.56	13.25	19.96	29.33
Old Customer, Old SKU	7.41	15.53	17.80	26.41
Old Customer, New SKU	2.26	6.99	9.89	14.79

Table 28: Hit Rates in Percent (USO)

B. 2.3. Data Preparation USC

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	6.42	15.51	21.83	32.43
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	12.15	23.14	28.57	36.48
Sticky ATM				
All Customers	6.58	15.74	22.00	32.37
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	11.41	20.44	25.81	33.25
CF				
All Customers	6.92	15.84	21.38	31.73
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	12.28	19.94	23.40	27.51
Bigram				
All Customers	7.48	16.60	22.70	32.60
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	15.90	24.92	28.64	33.09
Unigram				
All Customers	5.31	13.57	19.25	29.77
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	2.04	5.45	9.88	14.99

Table 29: Hit Rates in Percent (USC)

B. 2.4. Data Preparation 11ALL

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	36.49	54.42	61.13	69.46
Old Customer, Old SKU	45.99	66.58	73.91	82.64
Old Customer, New SKU	7.58	17.39	22.21	29.32
Sticky ATM				
All Customers	38.32	54.67	61.63	69.25
Old Customer, Old SKU	48.17	67.10	74.96	82.60
Old Customer, New SKU	8.14	15.71	21.22	27.72
CF				
All Customers	66.61	74.13	76.02	77.70
Old Customer, Old SKU	84.46	91.71	93.00	93.96
Old Customer, New SKU	12.27	20.58	24.32	28.20
Bigram				
All Customers	71.18	75.82	77.41	79.23
Old Customer, Old SKU	89.34	91.96	92.67	93.41
Old Customer, New SKU	15.82	26.55	31.08	36.15
Unigram				
All Customers	4.02	8.51	10.00	16.38
Old Customer, Old SKU	4.55	9.70	11.58	18.96
Old Customer, New SKU	2.41	4.91	5.21	8.52

Table 30: Hit Rates in Percent (11ALL)

B. 2.5. Data Preparation 11USO

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	33.46	47.68	53.35	60.18
Old Customer, Old SKU	56.39	74.14	79.78	86.00
Old Customer, New SKU	12.13	20.42	26.46	34.17
Sticky ATM				
All Customers	34.07	48.34	53.68	60.32
Old Customer, Old SKU	56.22	75.19	80.80	86.48
Old Customer, New SKU	11.87	21.91	26.63	35.34
CF				
All Customers	45.18	52.77	55.24	58.17
Old Customer, Old SKU	78.84	87.12	89.05	90.47
Old Customer, New SKU	12.81	19.73	22.72	27.11
Bigram				
All Customers	45.04	52.67	55.58	58.90
Old Customer, Old SKU	75.09	81.14	82.87	84.58
Old Customer, New SKU	16.19	25.36	29.45	34.38
Unigram				
All Customers	4.14	9.73	12.11	18.53
Old Customer, Old SKU	6.56	15.15	18.76	26.54
Old Customer, New SKU	1.82	4.52	5.72	10.83

Table 31: Hit Rates in Percent (11USO)

B. 2.6. Data Preparation 11USC

	|RS|=1	**|RS|=3**	**|RS|=5**	**|RS|=10**
ATM				
All Customers	14.22	25.49	32.15	40.20
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	14.22	25.49	32.15	40.20
Sticky ATM				
All Customers	13.49	24.11	30.25	38.82
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	13.49	24.11	30.25	38.82
CF				
All Customers	12.05	19.37	22.68	26.82
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	12.05	19.37	22.68	26.82
Bigram				
All Customers	14.81	23.57	27.33	31.90
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	14.81	23.57	27.33	31.90
Unigram				
All Customers	2.24	6.98	10.13	15.59
Old Customer, Old SKU	-	-	-	-
Old Customer, New SKU	2.24	6.98	10.13	15.59

Table 32: Hit Rates in Percent (11USC)

C. Appendix for *Simulation*

This appendix demonstrates how purchase data of different structures can be generated and additional information on the performance of recommender models can be gained. In the following, the set-up (C. 1) of the simulation, some descriptive statistics (C. 2) of the simulated data sets, and the hit rate results (C. 3) for three recommender models using the simulation data are presented. At the end of this appendix, the R code (C. 4) of the simulation is displayed.

C. 1. Set-Up

In this simulation, 18 different data sets of purchase data were simulated and used to fit some recommender models. To generate the data, the *generative process of the ATM model* was used and different values for beta, alpha, and number of topics (K) were selected. For *beta*, the six values 1, 0.5, 0.1, 0.05, 0.01, and 0.001 were chosen, and for *alpha*, the three values 1, 0.5, and 0.1 were chosen. The *number of topics* was determined to be $K = 20$. With each possible combination of alpha, beta, and number of topics, one data set was generated. Each data set contained 2,000 *customers* with a potential *product assortment* of 1,000 items. The *number of items per customer* was simulated from a truncated log-normal distribution using mean = 1.5 and sd = 4.5. The drawn values were rounded up or down to the nearest whole number. The log-normal distribution has two major advantages. First, it only provides positive values. This is important since the number of items per customer can never be negative. Second, the log-normal distribution controls better for the standard deviation than other distributions. This allows varying the number of products per customer more strongly. Such data is more likely to correspond to real purchase data. Table 33 summarizes all parameter settings considered in this simulation.

Beta	1, 0.5, 0.1, 0.05, 0.01, 0.001
Alpha	1, 0.5, 0.1
Number of topics (K)	20
Customers	2,000
Product assortment	1,000
Number of items per customer	truncated log-normal: mean = 1.5; sd = 4.5

Table 33: Parameter Settings for Simulation

C. 2. Descriptive Statistics

Descriptive statistics of all simulated data sets are displayed in Table 34. Each row represents one data set-up. Information is given on the number of items unique (*n.SKU.uniq*), in total (*n.SKU.total*), and per customer (*n.SKU.cust*). The number of items unique varied strongly between the different data set-ups. In particular, it decreased when a lower beta value was chosen. The total number of products was almost equal for all data set-ups, at about 17,000 items. The statistics on the number of SKUs per customer did not differ much between the different data set-ups. On average, each customer purchased around 8 to 9 items. The median was below the mean and equaled 4 items. This implies that the distribution is skewed to the right, showing a long tail toward higher values. The minimum number of items per customer was 2 and the maximum was 475. The last two columns provide information on the percentage of repeat items (*perc.rep*) and the concentration of the item distribution overall (*concentr*). Both variables varied between the single data set-ups. They increased with a smaller beta value. While *concentr* was largely unaffected by the value of alpha, *perc.rep* additionally increased with a lower alpha.

Each of the 18 different data set-ups was replicated 60 times. This was done because, when data is simulated from distributions, each draw generates slightly different results.

Hit rate results are therefore displayed below as box plots to show these variations. On the simulated data sets, the ATM model using recommendation generation approach 1 (ATM_1)[16], the ATM model using recommendation generation approach 2 (ATM_2)[17], and the most basic benchmark model Unigram are applied as examples. The models are compared based on their hit rates for predicting novel items.

C. 3. Results

Figure 24 displays the models' hit rates for all simulated data sets. The results reveal that recommender models were not equally well suited for all types of data sets. For example, there were data set-ups where the ATM model was much better than the Unigram model. In other data set-ups, though, the performance of both models was almost identical. Here, it makes little sense to implement the much more complex topic model. The simulation also allows comparing a model's performance using distinct recommendation generation approaches. The plots show that ATM_2 is either equivalent to or superior to ATM_1 for all simulated data sets. The same result was found for the data used in the empirical part of this dissertation.

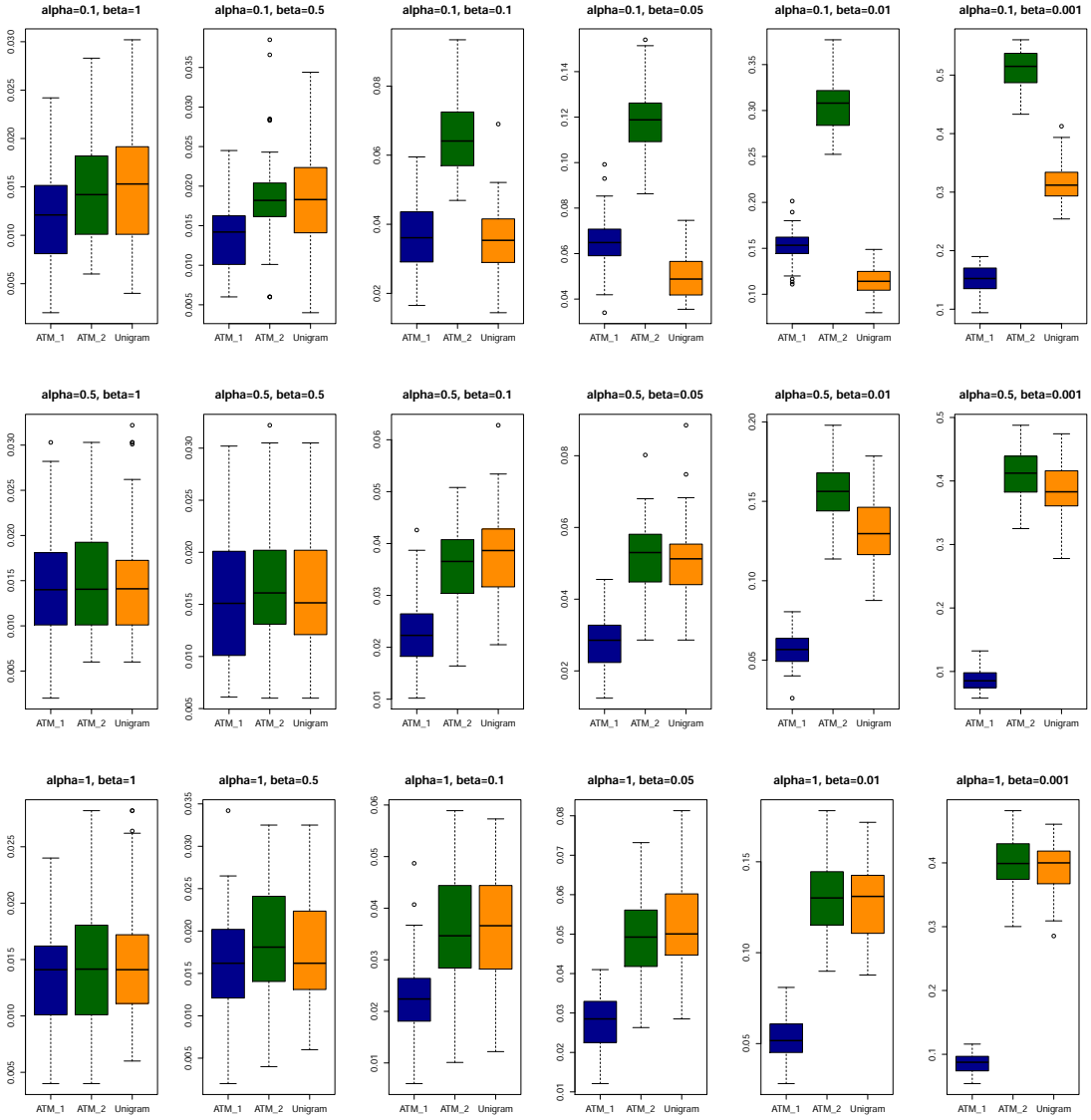
This and additional insights into the applicability of topic models as recommender models can be gained through simulating data. Ideally, a simulation would help to set up boundary conditions under which topic models work best in comparison with other recommender models. The big advantage of a simulation is that it can be conducted very quickly and with minimal effort. For instance, it does not require data collection from different online shops or the cumbersome cleaning of data. Instead, the simulated data can be used right away.

[16]The recommendation generation approach 1 was explained in Chapter 6.1.5.
[17]The recommendation generation approach 2 was explained in Chapter 6.1.5.

Set-up			n.SKU.uniq	n.SKU.total	n.SKU.cust							perc.rep	concentr
beta	alpha	K			mean	q25	q50	q75	sd	min	max		
1	1	20	1000	17258	8.63	2	4	8	16.51	2	345	0.0203	0.2977
0.5	1	20	1000	17376	8.69	2	4	8	17.81	2	423	0.0245	0.3195
0.1	1	20	989	17194	8.60	2	4	8	16.32	2	331	0.0329	0.4324
0.05	1	20	947	17389	8.69	2	4	8	16.99	2	368	0.0472	0.5117
0.01	1	20	596	17309	8.65	2	4	8	17.22	2	395	0.1126	0.6784
0.001	1	20	136	17240	8.62	2	4	8	16.98	2	363	0.3048	0.7552
1	0.5	20	1000	17387	8.69	2	4	8	17.65	2	402	0.0232	0.2972
0.5	0.5	20	1000	17469	8.73	2	4	8	18.26	2	411	0.0271	0.3197
0.1	0.5	20	990	17355	8.68	2	4	8	17.11	2	381	0.0400	0.4340
0.05	0.5	20	945	17323	8.66	2	4	8	17.07	2	392	0.0564	0.5117
0.01	0.5	20	594	17585	8.79	2	4	9	18.40	2	436	0.1430	0.6815
0.001	0.5	20	135	17286	8.64	2	4	8	17.00	2	362	0.3532	0.7563
1	0.1	20	1000	17471	8.74	2	4	8	17.19	2	377	0.0258	0.2968
0.5	0.1	20	1000	17426	8.71	2	4	8	16.87	2	352	0.0312	0.3204
0.1	0.1	20	989	17444	8.72	2	4	8	19.00	2	475	0.0721	0.4359
0.05	0.1	20	947	17309	8.65	2	4	8	17.60	2	410	0.1001	0.5122
0.01	0.1	20	596	17304	8.65	2	4	8	17.20	2	367	0.2366	0.6837
0.001	0.1	20	138	17441	8.72	2	4	8	18.18	2	428	0.5136	0.7573

The first column on the left shows the different data set-ups comprising beta, alpha, and K. In the subsequent columns, descriptive statistics for each set-up are displayed. The shown values are average values over the 60 replicates. *n.SKU.uniq* is the unique number of items and *n.SKU.total* is the total number of items in a data set. *n.SKU.cust* stands for the number of items per customer for which mean, first quartile (q25), median (q50), third quartile (q75), standard deviation (sd), minimum (min), and maximum (max) are calculated. *perc.rep* displays the percentage of repeat items in the data. It was calculated by using repeat items at the customer level. *concentr* is the concentration of the item distribution overall in a data set. A lower value stands for lower concentration and means that product purchases are spread rather evenly across the product assortment. Contrarily, a higher value stands for higher concentration and means that a few products were bought disproportionately often compared to others. In detail, the values describe the proportion of data that is covered by 20% of the product assortment. For example, the first value in the column reads "with 20% of the product assortment, 29.77% of the data can be generated."

Table 34: Descriptive Statistics of Simulated Data Sets



The upper row of plots shows the hit rates for all data set-ups using alpha = 0.1, the middle row for all data set-ups using alpha = 0.5, and the lower row for all data set-ups using alpha = 1. Moving from left to right, the data set-ups have lower beta values. Hit rates for each model are displayed as box plots. Each model has its own color. Blue represents the ATM model using recommendation generation approach 1, green represents the ATM model using recommendation generation approach 2 and orange represents the Unigram model.

Figure 24: Hit Rate Results

C. 4. R Code

```
## SIMULATION
## --> with generative process of ATM

library(bayesm)
library(data.table)
rm(list=ls())

##-------------------------------------------------##
## Set Parameters:
K = 20              # number of topics
V = 1000            # number of unique items (product assortment)
C = 2000            # number of customers
N_c = NULL          # number of items per customer
alpha = 1           # hyper-parameter for theta
beta = 0.1          # hyper-parameter for phi
##-------------------------------------------------##

#########################################################################
## 1. DRAW OF PHI (FOR EACH TOPIC)
#########################################################################
Phi = matrix(NA,V,K)
for(k in 1:K){Phi[,k] = rdirichlet(rep(beta,V))}
## --> each column corresponds to phi_k

#########################################################################
## 2.A. DRAW OF THETA (FOR EACH CUSTOMER)
#########################################################################
Theta = matrix(NA,K,C)
for(c in 1:C){Theta[,c] = rdirichlet(rep(alpha,K))}
## --> each column corresponds to theta_c

#########################################################################
## 2.B.i  DRAW OF Z (FOR EACH ITEM IN CUSTOMER C)
## 2.B.ii DRAW OF X (FOR EACH ITEM IN CUSTOMER C)
#########################################################################
Data = NULL
data = NULL
```

```
38
39 for(c in 1:C){
40
41   ##-----------------------------------------------------------##
42   ## Determine number of items for customer c:
43   ##-----------------------------------------------------------##
44   ## Draw from truncated log-normal distribution
45   m <- log(0.5); s <- log(3.0); n <- 1
46   Y <- round(rlnorm(n, meanlog = m, sdlog = s),0)
47   Y0 <- Y[Y>1]; r <- (n - length(Y0))
48
49   while(r>0){
50     Y <- round(rlnorm(r, meanlog = m, sdlog = s),0)
51     Y0 <- c(Y0,Y[Y>1]); r <- (n - length(Y0))
52   }
53
54   N_c <- Y0
55
56   ##-----------------------------------------------------------##
57   ## Draw topic assignment z for each item in customer c
58   ##-----------------------------------------------------------##
59   z = rep(NA,N_c)
60   for(n in 1:N_c){z[n] = which.max(rmultinom(1,1,Theta[,c]))}
61
62   ##-----------------------------------------------------------##
63   ## Draw item x for each item in customer c
64   ##-----------------------------------------------------------##
65   x = rep(NA,N_c)
66   for(n in 1:N_c){x[n] = which.max(rmultinom(1,1,Phi[,z[n]]))}
67
68   ##-----------------------------------------------------------##
69   ## Store data for each author
70   ##-----------------------------------------------------------##
71   Data[[c]] = list(c = c, N_c = N_c, z = z, x = x)
72
73   ## Get data of current customer:
74   data.new = cbind(Data[[a]]$x, Data[[a]]$z, Data[[a]]$c)
75
76   ## Stack data across all customers:
```

```
  data = rbind(data, data.new)
  ## --> matrix: where each row corresponds to one word

}

colnames(data) <- c("x","z","c")
data
x <- data[,"x"]
z <- data[,"z"]
c <- data[,"c"]
mycustomer <- split(x,c)

rm(data.new, C, K, k, m, n, N_c, r, s, V, Y, Y0)

#########################################################################
## SIMULATED DATA SET
#########################################################################
mycustomer
```

References

Abdollahpouri, Himan, Robin Burke, and Bamshad Mobasher (2019). "Managing Popularity Bias in Recommender Systems with Personalized Re-Ranking." *Proceedings of the 32nd International Florida Artificial Intelligence Research Society Conference (FLAIRS'19)*, 413–418.

Abinaya, G. and S. Godfrey Winster (2014). "Event Identification in Social Media through Latent Dirichlet Allocation and Named Entity Recognition." *Proceedings of IEEE International Conference on Computer Communication and Systems (ICCCS'14)*, 142–146.

Adamopoulos, Panagiotis (2013). "Beyond Rating Prediction Accuracy: On New Perspectives in Recommender Systems." *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys'13)*, 459–462.

Adamopulos, Panagiotis and Alexander Tuzhilin (2014). "On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected." *ACM Transactions on Intelligent Systems and Technology* 5.4, 1–32.

Aggarwal, Charu C. (2016). "Recommender Systems: The Textbook." First edition. Springer.

Airoldi, Edoardo M., David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg (2015). "Introduction to Mixed Membership Models and Methods." *Handbook of Mixed Membership Models and Their Applications*. Ed. by Edoardo M. Airoldi, David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg. Boca Raton, FL, USA: CRC Press, 3–14.

Alsumait, Loulwah, Pu Wang, Carlotta Domeniconi, and Daniel Barbará (2010). "Embedding Semantics in LDA Topic Models." *Text Mining: Applications and Theory*. Ed. by Michael W. Berry and Jacob Kogan. Chichester, U.K: Wiley, 183–204.

Ansari, Asim, Yang Li, and Jonathan Z. Zhang (2018). "Probabilistic Topic Model for Hybrid Recommender Systems: A Stochastic Variational Bayesian Approach." *Marketing Science* 37.6, 987–1008.

Asuncion, Arthur, Max Welling, Padhraic Smyth, and Yee W. Teh (2009). "On Smoothing and Inference for Topic Models." *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*, 27–34.

Bishop, Christopher M. (2006). "Pattern Recognition and Machine Learning." New York, NY, USA: Springer Science+Business Media, LLC.

Blei, David M. (2009). "Topic Models." *Machine Learning Summer School (MLSS'09)*.

Blei, David M. (2012). "Probabilistic Topic Models." *Communications of the ACM* 55.4, 77–84.

Blei, David M. (2014). "Build, Compute, Critique, Repeat: Data Analysis with Latent Variable Models." *Annual Review of Statistics and Its Application* 1, 203–232.

Blei, David M. and Michael I. Jordan (2003). "Modeling Annotated Data." *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'03)*, 127–134.

Blei, David M. and John D. Lafferty (2006). "Dynamic Topic Models." *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, 113–120.

Blei, David M. and John D. Lafferty (2007). "A Correlated Topic Model of Science." *The Annals of Applied Statistics* 1.1, 17–35.

Blei, David M. and John D. Lafferty (2009). "Topic Models." *Text Mining*. Ed. by Ashok Srivastava and Mehran Sahami. Boca Raton, FL, USA: CRC Press, 71–93.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). "Latent Dirichlet Allocation." *Journal of Machine Learning Research* 3, 993–1022.

Bodapati, Anand V. (2008). "Recommendation Systems with Purchase Data." *Journal of Marketing Research* 45, 77–93.

Boyd-Graber, Jordan, David Mimno, and David Newman (2015). "Care and Feeding of Topic Models: Problems, Diagnostics, and Improvementes." *Handbook of Mixed Membership Models and Their Applications*. Ed. by Edoardo M. Airoldi, David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg. Boca Raton, FL, USA: CRC Press, 225–254.

Burke, Robin (2007). "Hybrid Web Recommender Systems." *The Adaptive Web*. Ed. by Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl. Berlin, Heidelberg: Springer, 377–408.

Burke, Robin, Alexander Felfernig, and Mehmet H. Göker (2011). "Recommender Systems: An Overview." *AI Magazine* 32.3, 13–18.

Büschken, Joachim (2013). "Modeling Discrete Data via Latent Dirichlet Allocation: Technical Note." *Catholic University Eichstätt-Ingolstadt, Germany*, 1–22.

Büschken, Joachim and Greg M. Allenby (2016). "Sentence-Based Text Analysis for Customer Reviews." *Marketing Science* 35.6, 953–975.

Carpenter, Bob (2010). "Integrating Out Multinomial Paramenters in Latent Dirichlet Allocation and Naive Bayes for Collapsed Gibbs Sampling." *LingPipe, Inc* 1.4, 1–10.

Casella, George and Edward I. George (1992). "Explaining the Gibbs Sampler." *The American Statistician* 46.3, 167–174.

Chang, Jonathan, Jordan Boyd-Graber, Sean Gerrish, Chong Wang, and David M. Blei (2009). "Reading Tea Leaves: How Humans Interpret Topic Models." *Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09)*, 288–296.

Chen, Si and Yufei Wang (2014). "Latent Dirichlet Allocation." *University of California, San Diego, USA*, 1–12.

Christidis, Konstantinos, Dimitris Apostolou, and Gregoris Mentzas (2010). "Exploring Customer Preferences with Probabilistic Topics Models." *National Technical University of Athens, Greece*, 1–13.

Christidis, Konstantinos and Gregoris Mentzas (2013). "A Topic-based Recommender System for Electronic Marketplace Platforms." *Expert Systems with Applications* 40, 4370–4379.

Darling, William M. (2011). "A Theoretical and Practical Implementation Tutorial on Topic Modeling and Gibbs Sampling." *University of Guelph, Ontario, Canada*, 1–10.

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman (1990). "Indexing by Latent Semantic Analysis." *Journal of the American Society for Information Science* 41.6, 391–407.

del Olmo, Félix Hernández and Elena Gaudioso (2008). "Evaluation of Recommender Systems: A New Approach." *Expert Systems with Applications* 35, 790–804.

Erosheva, Elena A. (2002). "Grade of Membership and Latent Structure Models with Application to Disability Survey Data." Pittsburgh, PA, USA: Department of Statistics, Carnegie Mellon University.

Erosheva, Elena A., Stephen E. Fienberg, and John D. Lafferty (2004). "Mixed-Membership Models of Scientific Publications." *Proceedings of the National Academy of Sciences (PNAS)* 101 (suppl 1), 5220–5227.

Fleder, Daniel and Kartik Hosanagar (2009). "Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity." *Management Science* 55.5, 697–712.

Galyardt, April (2015). "Interpreting Mixed Membership Models: Implications of Erosheva's Representation Theorem." *Handbook of Mixed Membership Models and Their Applications*. Ed. by Edoardo M. Airoldi, David M. Blei, Elena A. Erosheva, and Stephen E. Fienberg. Boca Raton, FL, USA: CRC Press, 39–66.

Ge, Mouzhi, Carla Delgado-Battenfeld, and Dieter Jannach (2010). "Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity." *Proceedings of the 2010 ACM Conference on Recommender Systems (RecSys'10)*, 257–260.

Geigle, Chase (2016). "Inference Methods for Latent Dirichlet Allocation." *University of Illinois at Urbana-Champaign, USA*, 1–29.

Gelfand, Alan E. and Adrian F. M. Smith (1990). "Sampling-Based Approaches to Calculating Marginal Densities." *Journal of the American Statistical Association* 85.410, 398–409.

Geman, Stuart and Donald Geman (1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6.6, 721–741.

Gilks, W. R. and P. Wild (1992). "Adaptive Rejection Sampling for Gibbs Sampling." *Applied Statistics* 41.2, 337–348.

Griffiths, Thomas L. and Mark Steyvers (2004). "Finding Scientific Topics." *Proceedings of the National Academy of Sciences (PNAS)* 101 (suppl 1), 5228–5235.

Grün, Bettina and Kurt Hornik (2011). "topicmodels : An R Package for Fitting Topic Models." *Journal of Statistical Software* 40.13, 1–30.

Gunawardana, Asela and Guy Shani (2009). "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks." *Journal of Machine Learning Research* 10, 2935–2962.

Heinrich, Gregor (2008). "Parameter Estimation for Text Analysis: Technical Note." *vsonix GmbH and University of Leipzig, Germany*, 1–31.

Hendel, Avishai, Daphna Weinshall, and Shmuel Peleg (2010). "Identifying Surprising Events in Videos Using Bayesian Topic Models." *10th Asian Conference on Computer Vision (ACCV'10)*, 448–459.

Herlocker, Jonathan L., Konstan A. Konstan, Loren G. Terveen, and John T. Riedl (2004). "Evaluating Collaborative Filtering Recommender Systems." *ACM Transactions on Information Systems* 22.1, 5–53.

Hofmann, Thomas (1999). "Probabilistic Latent Semantic Indexing." *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, 50–57.

Hruschka, Harald (2014). "Linking Multi-Category Purchases to Latent Activities of Shoppers: Analysing Market Baskets by Topic Models." *Marketing ZFP - Journal of Research and Management* 36.4, 267–273.

Hruschka, Harald (2016). "Hidden Variable Models for Market Basket Data. Statistical Performance and Managerial Implications." *University of Regensburg, Germany*, 1–17.

Hu, Diane J. (2009). "Latent Dirichlet Allocation for Text, Images, and Music." *University of California, San Diego, USA*, 1–19.

Hu, Diane J. and Lawrence K. Saul (2009). "A Probabilistic Topic Model for Unsupervised Learning of Musical Key-Profiles." *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR'09)*, 1–6.

Hu, Yifan, Yehuda Koren, and Chris Volinsky (2008). "Collaborative Filtering for Implicit Feedback Datasets." *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM'08)*, 263–272.

Ishigaki, Tsukasa, Nobuhiko Terui, Tadahiko Sato, and Greg M. Allenby (2015). "Topic Modeling of Market Responses for Large-Scale Transaction Data." *Tohoku University, Japan*, 1–41.

Iwata, Tomoharu and Hiroshi Sawada (2013). "Topic Model for Analyzing Purchase Data with Price Information." *Data Mining and Knowledge Discovery* 26.3, 559–573.

Jacobs, Bruno J.D., Bas Donkers, and Dennis Fok (2016). "Model-Based Purchase Predictions for Large Assortments." *Marketing Science* 35.3, 389–404.

Jannach, Dietmar, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac (2015). "What Recommenders Recommend: An Analysis of Recommendation Biases and Possible Countermeasures." *User Modeling and User-Adapted Interaction* 25.5, 427–491.

Jannach, Dietmar, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich (2011). "Recommender Systems: An Introduction." 1st ed. New York, NY, USA: Cambridge University Press.

Jaradat, Shatha and Mihhail Matskin (2019). "On Dynamic Topic Models for Mining Social Media." *Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining*. Ed. by Nitin Agarwal, Nima Dokoohaki, and Serpil Tokdemir. Cham, Switzerland: Springer, 209–230.

Jordan, Franz (2017). "150 Mio. Produkte auf Amazon.de, 280 Mio. in USA." Ed. by Sellics: https://sellics.com/de/blog-amazon-sortimentgroesse-laender-vergleich/. accessed: 09-09-2019.

Kaminskas, Marius and Derek Bridge (2016). "Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems." *ACM Transactions on Interactive Intelligent Systems* 7.1, 2:1–2:42.

Kapoor, K., V. Kumar, L. Terveen, J. A. Konstan, and P. Schrater (2015). “"I Like to Explore Sometimes": Adapting to Dynamic User Novelty Preferences.” *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys'15)*, 19–26.

Kotkov, Denis, Jari Veijalainen, and Shuaiqiang Wang (2016). “Challenges of Serendipity in Recommender Systems.” *Proceedings of the 12th International Conference on Web Information Systems and Technologies (WEBIST'16)* 2, 251–256.

Kotkov, Denis, Shuaiqiang Wang, and Jari Veijalainen (2016). “A Survey of Serendipity in Recommender Systems.” *Knowledge-Based Systems* 111, 180–192.

Lee, Dokyun and Kartik Hosanagar (2014). “Impact of Recommender Systems on Sales Volume and Diversity.” *Proceedings of the International Conference on Information Systems (ICIS'14)*, 1–15.

Lee, Sangno, Jeff Baker, Jaeki Song, and James C. Wetherbe (2010). “An Empirical Comparison of Four Text Mining Methods.” *Journal of Computer Information Systems* 51.1, 1–10.

Liu, Duen-Ren, Chin-Hui Lai, and Wang-Jung Lee (2009). “A Hybrid of Sequential Rules and Collaborative Filtering for Product Recommendation.” *Information Sciences* 179, 3505–3519.

Maksai, Andrii, Florent Garcin, and Boi Faltings (2015). “Predicting Online Performance of News Recommender Systems through Richer Evaluation Metrics.” *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys'15)*, 179–186.

Mazarura, Jocelyn and Alta de Waal (2016). “A Comparison of the Performance of Latent Dirichlet Allocation and the Dirichlet Multinomial Mixture Model on Short Text.” *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech'16)*, 1–6.

McNee, Sean M., John Riedl, and Joseph A. Konstan (2006). "Being Accurate Is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems." *Proceedings of the 2006 Conference on Human Factors in Computing Systems (CHI'06)*, 1097–1101.

Mehrotra, Rishabh, Scott Sanner, Wray Buntine, and Lexing Xie (2013). "Improving LDA Topic Models for Microblogs via Tweet Pooling and Automatic Labeling." *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'13)*, 889–892.

Mimno, David M., Hanna M. Wallach, Edmund M. Talley, Miriam Leenders, and Andrew McCallum (2011). "Optimizing Semantic Coherence in Topic Models." *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*, 262–272.

Murphy, Kevin P. (2012). "Machine Learning: A Probabilistic Perspective." Massachusetts: The MIT Press.

Newman, David, Timothy Baldwin, Lawrence Cavedon, Eric Huang, Sarvnaz Karimi, David Martinez, Falk Scholer, and Justin Zobel (2010). "Visualizing Search Results and Document Collections Using Topic Maps." *Web Semantics: Science, Services and Agents on the World Wide Web* 8.2-3, 169–175.

Newman, David, Jey Lau, Karl Grieser, and Timothy Baldwin (2010). "Automatic Evaluation of Topic Coherence." *Proceedings of the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT'10)*, 100–108.

Niebles, Juan Carlos, Hongcheng Wang, and Li Fei-Fei (2008). "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words." *International Journal of Computer Vision* 79, 1249–1258.

Pathak, Bhavik, Robert Garfinkel, Ram D. Gopal, Rajkumar Venkatesan, and Fang Yin (2010). "Empirical Analysis of the Impact of Recommender Systems on Sales." *Journal of Management Information Systems* 27.2, 159–188.

Postma, O. J. and M. Brokke (2002). "Personalisation in Practice: The Proven Effects of Personalisation." *Journal of Database Marketing* 9.2, 137–142.

Pritchard, J. K., M. Stephens, and P. Donnelly (2000). "Inference of Population Structure Using Multilocus Genotype Data." *Genetics* 155.2, 945–959.

Rabe, L. (2019). "Umsatz von Amazon weltweit bis zum 2. Quartal 2019." Ed. by Statista: https://de.statista.com/statistik/daten/studie/197099/umfrage/. accessed 09-09-2019.

Rasiwasia, Nikhil and Nuno Vasconcelos (2013). "Latent Dirichlet Allocation Models for Image Classification." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.11, 2665–2679.

Reisenbichler, Martin and Thomas Reutterer (2019). "Topic Modeling in Marketing: Recent Advances and Research Opportunities." *Journal of Business Economics* 89, 327–356.

Ricci, Francesco, Lior Rokach, Bracha Shapira, and Paul B. Kantor, eds. (2011). "Recommender Systems Handbook." New York, NY, USA: Springer.

Rossi, Peter Eric, Greg Martin Allenby, and Robert Edward McCulloch (2006). "Bayesian Statistics and Marketing." Wiley Series in Probability and Statistics. Hoboken, NJ: John Wiley & Sons, Ltd.

Schröder, Nadine (2017). "Using Multidimensional Item Response Theory Models to Explain Multi-Category Purchases." *Marketing ZFP - Journal of Research and Management* 39.2, 27–37.

Schröder, Nadine, Andreas Falke, Harald Hruschka, and Thomas Reutterer (2017). "Analyzing Browsing and Purchasing Across Multiple Websites Based on Latent Dirichlet Allocation." *Proceedings of the 3rd International Conference on Big Data, Small Data, Linked Data and Open Data (ALLDATA'17)*, 40–44.

Senecal, Sylvain and Jacques Nantel (2004). "The Influence of Online Product Recommendations on Consumers' Online Choices." *Journal of Retailing* 80, 159–169.

Shani, Guy and Asela Gunawardana (2011). "Evaluating Recommendation Systems." *Recommender Systems Handbook*. Ed. by Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. New York, NY, USA: Springer, 257–297.

Smith, Brent and Greg Linden (2017). "Two Decades of Recommender Systems at Amazon.com." *IEEE Internet Computing* 21.3, 12–18.

Srinivasan, Srini S., Rolph Anderson, and Kishore Ponnavolu (2002). "Customer Loyalty in E-Commerce: An Exploration of Its Antecedents and Consequences." *Journal of Retailing* 78, 41–50.

Steyvers, Mark and Tom Griffiths (2007). "Probabilistic Topic Models." *Handbook of Latent Semantic Analysis*. Ed. by Thomas K. Landauer, Danielle S. McNamara, Simon Dennis, and Walter Kintsch. Mahwah, NJ, USA: Lawrence Erlbaum Associates Publishers, 427–448.

Sun, Feng-Tso, Martin Griss, Ole Mengshoel, and Yi-Ting Yeh (2013). "Latent Topic Analysis for Predicting Purchasing Behavior on the Social Web." *Proceedings of the 2013 Uncertainty in Artificial Intelligence Application Workshops (UAI'13)*, 67–76.

Syed, Shaheen and Marco Spruit (2018). "Selecting Priors for Latent Dirichlet Allocation." *Proceedings of the 2018 IEEE 12th International Conference on Semantic Computing (ICSC'18)*, 194–202.

Tirunillai, Seshadri and Gerard J. Tellis (2014). "Mining Marketing Meaning from Online Chatter: Strategic Brand Analysis of Big Data Using Latent Dirichlet Allocation." *Journal of Marketing Research* 51, 463–479.

Vargas, Saúl and Pablo Castells (2011). "Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems." *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys'11)*, 109–116.

Wallach, Hanna M. (2006). "Topic Modeling: Beyond Bag-of-Words." *Proceedings of the 23rd International Conference on Machine Learning (ICML'06)*, 977–984.

Wallach, Hanna M., David Mimno, and Andrew McCallum (2009). "Rethinking LDA: Why Priors Matter." *Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09)*, 1973–1981.

Wallach, Hanna M., Iain Murray, Ruslan Salakhutdinov, and David Mimno (2009). "Evaluation Methods for Topic Models." *Proceedings of the 26th International Conference on Machine Learning (ICML'09)*, 1105–1112.

Walsh, B. (2002). "Markov Chain Monte Carlo and Gibbs Sampling: Lecture Notes." *University of Arizona, USA*, 1–24.

Wang, Chong, D. Blei, and Fei-Fei Li (2009). "Simultaneous Image Classification and Annotation." *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, 1903–1910.

Wang, Chong and David M. Blei (2011). "Collaborative Topic Modeling for Recommending Scientific Articles." *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*, 448–456.

Wang, Xiaogang, Xiaoxu Ma, and Eric Grimson (2007). "Unsupervised Activity Perception by Hierarchical Bayesian Models." *Proceedings of IEEE Computer Society Conference on Computer Vision and Patter Recognition (CVPR'07)*, 1–8.

Wang, Yi (2008). "Distributed Gibbs Sampling of Latent Topic Models: The Gritty Details.", 1–16.

Wilson, Jobin, Santanu Chaudhury, and Brejesh Lall (2014). "Improving Collaborative Filtering Based Recommders Using Topic Modelling." *Proceedings of the 2014 IEEE/ WIC/ ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies (WI-IAT'14)*, 340–346.

Woodbury, Max A., Jonathan Clive, and Arthur Garson (1978). "Mathematical Typology: A Grade of Membership Technique for Obtaining Disease Definition." *Computers and Biomedical Research* 11.3, 277–298.

Yano, Tae, William W. Cohen, and Noah A. Smith (2009). "Predicting Response to Political Blog Posts with Topic Models." *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL'09)*, 477–485.

Zhang, Tong, Zhe-Ming Lu, Kap Luk Chan, and Zhen Li (2011). "Automatic Image Annotation and Retrieval Using the Latent Dirichlet Allocation Model." *IJCSES International Journal of Computer Sciences and Engineering Systems* 5.1, 1–4.

Zhang, Yuan Cao, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor (2012). "Auralist: Introducing Serendipity into Music Recommendation." *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM'12)*, 13–22.