Bayesian Cognitive Ranking Methods and Their Applications to Consumer Data Analysis and Molecular Bioinformatics

March ２０１９

Luo Song

Bayesian Cognitive Ranking Methods and Their Applications to Consumer Data Analysis and Molecular Bioinformatics

Graduate School of Systems and Information Engineering

University of Tsukuba

March ２０１９

Luo Song

UNIVERSITY OF TSUKUBA

Abstract

Doctoral Program in Social Systems and Management
Graduate School of Systems and Information Engineering

Doctor of Philosophy

Bayesian Cognitive Ranking Methods and Their Applications to Consumer Data Analysis and Molecular Bioinformatics

by Song LUO

This dissertation discusses cognitive ranking methods and their machine learning algorithms from a Bayesian perspective. Ranking methods play a fundamentally important role in a variety of modern information retrieval systems and can oftentimes be powered by machine learning algorithms. Machine learning is a data-driven method that addresses the question of how to build computer programs that improve their performance at some task through experience and has become central to many areas of interest in computer science, engineering, social sciences and other human endeavors involving information processing domains. The ranking methods discussed in this dissertation are cognitive, in the sense that knowledge representation regarding domain-specific ranking rationales is highly emphasized and mathematically formulated. Knowledge representation enriches the application of machine learning in the construction of agent-based ranking models for information retrieval systems. As a methodology of statistical inference, Bayesian approach has its own theoretical and practical advantages, especially when domain-specific knowledge is available and needs to be incorporated into a learning process.

Chapter 1 first explains the reasons why this dissertation prefers a ranking method that is Bayesian, cognitive and machine-learning driven. Then, Chapter 1 introduces the problems of our interest regarding consumer data analysis and molecular bioinformatics.

Chapter 2 analyzes a real-world dataset containing millions of smartphone users' behavior records. Two ranking methods are proposed to rank smartphone apps according to their quality assessment from a user's perspective, one for extracting superior apps and the other for arranging the extracted apps into a linear order by learning users' preferences from users' behavioral information. The former model is constructed by introducing a cognitive process of pointwise comparison, while the latter is a pairwise-comparison-based method that can refine the ranking results of the former. In both methods, we transform context-dependent information and domain-specific knowledge regarding ranking rationales into representational structures and construct computational procedures that operate on those structures. The resulting knowledge representation plays an important role in the construction of app quality assessment measures that can be used to mine useful knowledge from users' behavioral information. The computational procedures are simple and suitable for big data processing.

Chapter 3 presents a method for discovering the knowledge of item rank from consumer reviews. The basic idea of the method is to construct a cognitive ranking model and then to build it in the framework of computational learning theory in order to estimate its parameters. This idea formulates the questions of interest as a single biconvex optimization problem which has a relationship with SVM(Support Vector Machines). To facilitate the process of knowledge discovery, we develop a two-stage learning algorithm. In the first stage, we extract as much information as possible from the observed data, while in the second stage, the extracted information is further summarized into knowledge. Particularly, the learning algorithm is essential Bayesian, combined with Frequentist learning theory. The Frequentist approach is adopted to deal with the difficulty that the likelihood function cannot be explicitly formulated, while the Bayesian approach is used to incorporate scientific hypotheses into the ranking model.

Chapter 4 details a Bayesian framework for the problem of detecting consecutive positives in pooling experiments. More concretely, a Bayesian machine learning algorithm is proposed to decode the multi-leveled pooling results given by random designs. The choice of important parameters is also discussed within the framework. It shows some happy coincidences between theoretical computation results

and previously known simulation results. Numerical simulation results show that the Bayesian framework is promising to deal with the uncertainty in real settings and that the prior information is helpful to reduce the number of pools needed to detect the positives. One final point to make here is that the framework can be used as a component of an expert system for pooling experiments, so as to making the screening procedure automatic and interactive.

Acknowledgements

The dissertation has been written with the direct and indirect support and help of many individuals to whom I am grateful very much. First of all, I would like to thank Professor Maiko Shigeno, Professor Ying Miao, Associate Professor Hiroyasu Ando, Associate Professor Masahiro Hachimori, and Associate Professor Yuichi Takano for thoroughly reading the manuscript and giving me many constructive comments that are helpful to improve the clarity of the content of the dissertation.

In particular, I would like to express my special thanks to my supervisor, Professor Maiko Shigeno, for her everlasting support and encouragement, especially during the very challenging time of my academic life. She always held the highest consideration and enthusiasm toward my research and gave high priority to funding my participation in conferences to present my research results while I was a Ph.D. student. Over these years, I have benefited much from her valuable comments which have influenced this dissertation directly or indirectly. This dissertation and some of the ideas it contains would not have been achieved without her professional guidance and persistent help. I am always feeling lucky and happy to be a Ph.D. student of hers.

In addition, I would like to thank Professor Ying Miao for introducing me the topic of combinatorial group testing and motivating me to consider the problem of probabilistic group testing for screening a clone map. During these years, I also benefited much from valuable suggestions given by Professor Ushio Sumita, Professor Yoshitsugu Yamamoto, and Professor Yuichiro Kanazawa. I thank them for their helpful instructions and comments. I also owe many thanks to my co-author Mingchao Zhang, to my teammates in the FULLER project, Wenbo Ma, Megumi Fujimoto, Mutiara Dian Sari and Miki Zago, and to FULLER Inc. for providing the dataset of users' behavior records and various technical support in the project.

Finally, I deeply thank my parents, Jing Yu and Naiping Luo, for their continuous love, patience, support, and enthusiasm, without which this dissertation would have never been completed. I also thank my closest friends, Ting Qiu, Mingchao Zhang, Mingxin Zhang, Chengpeng Tan, Yi Wang, and Weiwei Yin for their everlasting encouragement, consideration, trust, and support, without which the journey of my doctoral program would have been even tougher.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Ranking items is a fundamental task in a variety of applications where information of priority relationship between a set of items is required for comparison purposes. For example, in politics, rankings focus on the comparison of economic, social, environmental and governance performance of countries [126]; in sports, individuals or teams are ranked for comparison of relative chance to win a game [23]; in document retrieval systems, documents are ranked in agreement with user preferences and processed queries [16]. In addition, ranking is also pivotal for many other industrial and commercial systems, such as question answering in Q&A systems [143], multimedia retrieval in multimedia search systems [188], text summarization in natural language understanding systems [119], online advertising in modern marketing systems [186] and product recommendation in e-Commerce systems [108].

Because of its central role, great attention has been paid to the research and development of ranking technologies and a variety of ranking models are developed to express application-specific ranking rationales and produce rankings of items in an efficient and effective way. In general, a ranking model of a ranking process gives a mathematical description of the operating behavior of a ranker, as well as its qualitative and quantitative operating conditions, by specifying the principal quantities of the process, namely: input, system parameters and output. The analysis of a given process via the corresponding mathematical model may be divided into three distinct types of problems [17]:

- **Direct problem:** Given the input and the system parameters, find out the output of the model.
- **Reconstruction problem:** Given the system parameters and the output, find out which input has led to this output.
- **Identification problem:** Given the input and the output, determine the system parameters which are in agreement with the relation between input and output.

Reconstruction problem and identification problem are called inverse problems, because they consist of finding out unknown factors of known consequences.

Machine learning is a useful tool for studying those problems, and successful algorithms have been invented that are effective for certain types of inverse problems. Then, what is machine learning? Machine learning is a data-driven method that addresses the question of how to build computer programs that improve their performance at some task through experience and has been central to many areas of interest in computer science, engineering, social sciences and other human endeavors involving information processing domains. An overview of machine learning

disciplines can be found in the standard textbooks [15, 54, 67, 86, 102, 122, 127]. Driven by information revolution, the recent advent of increasingly fast computing devices with large-scale storage capacity has resulted in the availability of unprecedented amounts of data. The availability of cost effective computing power and massive databases creates both opportunities and challenges, which has been motivating software researchers and practitioners to solve large-scale problems in the data-driven approach by developing efficient and effective machine learning algorithms. Some machine learning algorithms have proven to be of great practical value in a variety of modern applications [51, 115, 120]. They are especially useful in (1) data mining problems where massive databases from heterogeneous information sources may contain valuable implicit regularities that can be discovered through an automatic process of hypothesis formulation and hypothesis testing (e.g., to analyze outcomes of medical treatments from patient databases [91, 103] or to learn general rules for credit worthiness from financial databases [104, 150]); (2) less well understood or highly empirical domains where humans might not have enough well-formulated knowledge that is needed to develop effective algorithms (e.g., spam email detection [110], handwriting recognition [138], speech recognition [140], face recognition from images [162]) or highly specialized domains where it is extremely tedious or infeasible to directly translate domain-specific human knowledge into explicit algorithms with good performance (e.g., transaction fraud detection [4], strategic investment decision [169] or new drug discovery [2]); and (3) domains where the program must dynamically adapt to changing conditions (e.g., controlling manufacturing processes under changing conditions [123], online learning to rank[153] or adapting to the changing of reading interests of individuals [66]).

Without knowing the ranking process of the ranker, a successful machine learning algorithm is able to produce a permutation of items in a way that mimics the behavior of the ranker. This application of machine learning technologies to ranking problems has led to many innovative and effective ranking models, and also has led to the emerging of a new research area named learning to rank [22, 31, 32, 33, 109, 183, 191]. However, in majority of the methods of learning to rank, prediction is the focus, and as a consequence the ranking process is approximated by a class of simplified hypotheses and then evaluated by measures such as Spearman's and Kendall's rank correlation coefficients [96], with both hypotheses and evaluation measures independent of prior knowledge of the ranking rationale behind the ranking process. So the resulting model, though perhaps good at mimicking for prediction tasks, may not be helpful to further explain or understand the behavior of the ranker.

In real-world applications, there will present a large quantity of information regarding experimental uncertainty, domain-specific prior knowledge and experimental observations. Therefore, scientific discovery is commonly conducted in an interactive way and involves the task of experiment design again and again by taking into consideration a variety of domain-specific knowledge back and forth. This oftentimes requires an automatic knowledge discovery system to process the domain-specific structures in the mind and the computational procedures that operate on those structures. In this dissertation, we are concerned with the cognitive aspect of ranking models and emphasize on domain knowledge representation. More specifically, we show (1) how domain-specific prior knowledge can be represented and incorporated into a ranking model, (2) how this knowledge representation can be used for automatic knowledge discovery, (3) why and to what extent this knowledge representation will work. Unlike the techniques such as learning to rank that fit data with commonly used loss functions that stress more on the mathematical or statistical senses, we are more interested to construct application-specific ranking models,

serving as rational decision-making agents for ranking tasks. To this end, the ranking methods discussed in this dissertation will borrow conceptual ideas and analytical tools from a diverse set of fields, to name a few but not limited to, probability and statistics [148], Bayesian methods[8, 53, 166], artificial intelligence [117], mathematical programming [152, 159, 167], information theory [111], economics [24], psychology [151] and cognitive science [43]. These ideas and tools are employed, on the one hand, to construct learning algorithms for searching through a space of possible hypotheses to find the hypothesis that best fits the available observed data and other prior constraints or knowledge; and on the other hand, to establish theoretical results of why and how these learning algorithms might work.

Specifically, this dissertation discusses ranking methods and their machine learning algorithms from a Bayesian perspective. Then, why the Bayesian is preferred? As a methodology of statistical inference, Bayesian approach has its own theoretical and practical advantages. The pioneering works of Cox [37] and Pólya [139] opened up new worlds of thought, whose exploration laid a theoretical foundation for explaining plausible reasoning and its relationship with Bayesian system of probability theory [90]. In particular, the Bayesian system of probability theory is consistent with that of Kolmogorov. The standard mathematical rules of probability theory can be seen as uniquely valid principles of logic for plausible reasoning in general, so their range of applications can be vastly extended beyond conventional usage for calculating frequencies of "random variables". As a result, the imaginary distinction between "probability theory" and "statistical inference" disappears. In this sense, if well-designed, a learning machine implementing Bayesian inference is to use probability theory as the extended logic that is in accordance with the logic of science. This would give a learning machine great technical power and flexibility when applied in modern applications, not only capable of dealing with uncertainty in a variety of situations but also transparent for human to explain or assess its inference result.

Then, how does a Bayesian approach learn? The essence of applying Bayesian approach to a problem is to express all form of uncertainty in terms of probability. More concretely, beliefs about the unknowns of events or outcomes of interest are quantified by a probability measure conditional on the knowns including observed data and prior information. There are mainly three technical ingredients needed to implement Bayesian inference:

- prior probability distribution
- likelihood function
- decision making strategy

Bayesian inference starts with converting available prior information into a quantitative expression of prior probability distribution. Prior information may include pre-determined hypotheses to be tested, domain-specific information of the problem at hand, and accumulated beliefs derived from past experience. Hence, the selection of prior distribution usually requires technical and application-specific knowledge, making it somewhat case-dependent. However, the selection is not arbitrary, especially when one would construct a prior in some sense of optimality that incorporates and only incorporates the prior beliefs but nothing else. Next, a likelihood function is required to be constructed for describing the uncertainty of the observed data when a specific instance of the prior is fixed and known. A posterior distribution for these unknowns can be obtained by applying Bayes' rule, which takes account of both the prior and the training data. From the posterior distribution, a

predictive distribution can be obtained by, for example, sampling methods or approximate inference methods. Finally, a decision making strategy is constructed for translating the information of the predictive distribution into the prediction of the future data. More detailed discussions of Bayesian methods can be found in the books [13, 18, 19, 38, 59, 114].

In summary, designing a Bayesian machine learning algorithm involves a number of design choices, including translating subjective prior beliefs into a mathematically formulated model and prior, constructing posterior probability distribution as the model of performance metric to be learned, and an algorithm for learning the model from observed data. Although Bayesian approach has simple process in theory, technically, it is nontrivial to design an effective Bayesian machine learning algorithm. This may be due to the following difficulties: (1) the selection of the prior in the context of complex prior information would be challenging; (2) in some case, it is impossible to construct an explicit likelihood function of the observed data conditional on the parameters of the model of prior beliefs; and (3) when likelihood function cannot be explicitly formulated, the computational difficulties with Bayesian approach may not be overcome with sampling methods or approximate inference methods. One may face these difficulties when applying Bayesian approach to real world problems. Methodological solutions to ease these difficulties proposed in this work have been applied in the field of smartphone app quality assessment by analyzing users' behavioral data, in the field of customer opinion mining by analyzing textual data obtained from consumer reviews, and in the field of DNA library screening with non-adaptive group testing techniques.

The rest of the first chapter is organized as follows. Section 1.2 discusses the problem of ranking smartphone apps from a user's perspective. In the consumer data analysis project, smartphone users' behavioral records are used for predicting the rankings of smartphone apps. Chapter 2 details the descriptive process of data inspecting, cleansing and transforming processes, and then proposes two ranking algorithms, one for extracting superior apps and the other for arranging the extracted apps into a linear order. Section 1.3 presents the problem of discovering useful knowledge from consumer reviews. Chapter 3 details a Bayesian framework for this problem with a two-stage learning algorithms, and particularly attempts to discuss a theoretical question "is it possible to make machine learning transparent, by making prediction task and explanation task simultaneously?" Section 1.4 1.4 gives a general introduction of group testing and shows how ranking approach and its machine learning algorithm can power group testing techniques, and Chapter 4 will continue the topic of Section 1.4 and present a Bayesian framework for the problem of pooling experiments in the presence of consecutive positives.

1.2 Preference learning machine

In economics and other social sciences, a preference usually refers to an ordering relation between two or more items that, among a set of possible choices, is the one in a decision making process. Preference information plays a key role in automated decision making, and reasoning with preferences has been recognized as a particularly promising direction for artificial intelligence research and applications[55]. Preference learning is concerned with the automated acquisition of preference models from empirical data, to reveal preferences of an individual or a group of individuals. Emerging as a new branch of machine learning and data mining, preference learning

problems in general, ranking problems in particular, arise naturally in many applications such as information retrieval [81, 109, 163] and recommender systems [39, 118, 142, 141] where an increasing trend toward personalization of products and services can be recognized.

As smartphone usage has grown remarkably in recent years and become a most popular choice for communication and mobile computing, a huge amount of smartphone apps have also been developed and distributed via online markets such as Google Play Android app store and Apple app store. Correspondingly, new problems and challenges arose. Among them, how to conduct app quality assessment from a user's perspective is a fundamental problem and ongoing challenge. It is fundamental because quality assessment is expected to be able to provide reliable suggestions that are useful in various problem-solving and decision-making processes involved in app development and management. For example, quality assessment would be helpful for users to discover potentially useful apps, for content providers to improve information quality of content, for developers and product managers to monitor and improve app performance, for app stores to investigate user preferences and then recommend relevant apps to relevant users. Unfortunately, the explosion in variety and number of apps makes it an ongoing challenge. Correspondingly, ranking methods that could perform removal of non-relevant apps and sorting of relevant apps according to their quality assessment are desired in various practical applications concerning app development and management.

Chapter 2 discusses the problem of ranking smartphone apps according to their quality assessment from a user's perspective based on large-scale users' behavioral information which can automatically be collected without the requirement of users' active participation. The rankings of smartphone apps are understood as a preference learning process. In an application-specific project, we use a dataset of user profiles provided by FULLER Inc. From the data, it can be learned that which Android device installed what Android app(s) at what month during the period from October 2012 to June 2013. Each piece of the data is called an Android users' behavior record. Particularly, a device's sequential states of whether it has a specific app installed over a period of time can be represented as a binary sequence, called a usage record. The full dataset contains millions of the users' behavior records. With automatic data collection process, a growing number of such data become available, and new data will periodically arrive. The dataset of user profiles we use for app quality assessment is essentially different from those typically collected from developer self-reports, questionnaires, textual contents or star-ratings. This not only brings an opportunity to develop new insights into app quality assessment from a different angle but also cause new technical problems:

1. Users' behavior records are simply binary row sequences. But, as new data arrive periodically, the dataset expand rapidly in rows and columns. Therefore, we need to summarize the information and control the information loss.

2. To measure the predictive performance of a ranker, a loss function on rankings is usually needed. Apart from the type of ranking loss, which compares a predicted ranking with a given target ranking, it is also possible to compare a predicted ranking with a single class label. Unfortunately, since neither target ranking nor a single class label is available, there is no immediate performance measure to calibrate a learning procedure.

With this dataset, Chapter 2 proposes two ranking methods, one for extracting superior apps and the other for ordering the extracted apps. In each method,

a cognitive process of comparison is proposed and used for modeling the corresponding ranking rationale. The former ranking method produces rankings of apps in a pointwise way; the latter ranking method arranges a collection of apps into a linear order by pairwise comparisons. The pairwise-comparison-based method can be seen as a refinement of the point-comparison-based method. In both methods, context-dependent prior information as well as domain-specific knowledge was summarized and represented by the Bayesian approach. The resulting knowledge representation helps ease the technical problems mentioned above and also plays an important role in the preference learning phase and in the interpretation phase. Since the former ranking method is aimed at removing non-relevant apps efficiently and the latter is constructed for linearly ordering relevant apps effectively, the two ranking methods can work together to provide workable ranking results that might be useful for mining knowledge from users' behavioral information.

1.3 Knowledge discovery machine

Knowledge discovery in databases is an automatic, exploratory analysis and modeling of large data repositories. Knowledge discovery is the organized process of identifying valid, novel, useful, and understandable patterns from large and complex data sets. A full process of knowledge discovery involves several steps, depending on the specific problem at hand, while data mining is the core of the process, involving the inferring of algorithms that explore data, develop model and discover previously unknown patterns. A model is used for understanding phenomena from data, analysis and prediction. The accessibility and abundance of data today makes knowledge discovery and data mining a matter of considerable importance and necessity. There is a lot of hidden knowledge waiting to be discovered, which is the challenge created by today's abundance of data. Discovering and extracting knowledge from the complexity of available data is an intriguing task attracting many researchers and practitioners to accomplish. An overview of knowledge discovery and mining can be found in the books [57, 75, 158, 172, 171, 194].

Machine learning is a powerful tool in knowledge discovery process and learning algorithms have been invented that are effective for certain types of learning tasks. However, some data scientists criticized that machine learning is not transparent enough to be any practical in serious applications. Indeed, the philosophy of science is primarily interested in explanation and hence theorizing, while the area of machine learning and data mining is primarily interested in prediction and hence modeling. When explanation is the focus, one is theorizing; when prediction is the focus, the process is better described as modeling. Then, the general form of reasoning that encompasses both theorizing and modeling is sometimes called discovery [182]. Consequently, in some application areas concerning knowledge discovery, the question was not only how to learn but also what to learn and why. The need for explanations cannot be replaced by simply establishing the classes of each examples falling in. Sometimes, what is more interesting to know is to know the reasons behind the classification.

Motivated by this idea, we attempt to discuss whether it is possible to construct a learning machine that can make prediction and explanation simultaneously. In particular, when observing a collection of items ranked in a linear order, we are interested in the questions of why and how one item is ranked over another. Since ranking problems arise quite naturally in many application areas, and processing knowledge in terms of ranking is appealing as it allows one to specify desires in a

declarative way, to combine qualitative and quantitative modes of reasoning, and to deal with inconsistencies and exceptions in a quite flexible manner, this research direction may help us to develop insights into a ranking process when rankings of items are observable and prior knowledge is available.

E-commerce websites have proliferated at a rapid rate and improve customer experience and online businesses largely. Customers purchase products based on reviews provided by the consumers of the product, by finding out how other consumers have recommended the product based on its quality, usefulness and many other parameters. However, The explosion in number and variety of product rankings also brings new challenges in the way information is retrieved and knowledge is discovered. For example, it is oftentimes hard for a customer to determine the products that best match their requirements in an effective and efficient way. To ease this difficulty, e-Commerce service systems and information sharing platforms provide useful information such as the main attributes commented upon by customers and a variety of product rankings. For a variety of reasons, most of ranking systems and information sharing platforms don't provide the rationale behind their ranking results in an explicit and expressive way. Without such knowledge, when comparing similar products that provide almost the same functionality customers might wonder why and how one product is ranked over another. This gives rise to the need of discovering useful knowledge of product rankings from consumer reviews.

When consumer reviews of the ranked items are observed, Chapter 3 presents a method for discovering the knowledge of the rank of the items from the consumer reviews. The basic idea of the method is to construct a choice model and then to build it in the computational learning theory. This idea formulates the questions of interest as a biconvex optimization problem which has a relationship with SVM(Support Vector Machines). To facilitate the process of knowledge discovery, we develop a two-stage learning algorithm. In the first stage, we extract as much information as possible from the observed data, while in the second stage, the extracted information is further summarized into knowledge. Particularly, the learning algorithm is essential Bayesian, combined with Frequentist learning theory. Frequentist approach is adopted to deal with the difficulty that the likelihood function cannot be explicitly formulated, while Bayesian approach is used to incorporate scientific hypotheses into predicting model. Generalizing beyond the observed data given, the acquisition of this kind of knowledge and models may be useful for preference prediction, for example, to predict the preferences of a new individual or the same individual in a new situation.

1.4 Group testing machine

Group testing was proposed by Robert Dorfman [45] during World War II in order to efficiently test a large number of blood samples for a rare disease. The goal of group testing is to discover defective items in a large population with the minimum number of tests. Each test is applied to a subset of items, called pools, instead of testing these items one by one. When the outcome of a group test is negative, then all samples in the pool are good. Otherwise, there exists at least one defective sample, also called positive (but we do not know which one) in the pool and further testing on them is necessary. When the proportion of positives is relatively small, many of the outcomes of the pools are expected to be negative, and hence the total number of

tests is reduced. This powerful theory has been applied to many fields such as multi-access channel communication [12], coding theory [47], sparse signal recovery [65], network defective diagnosis [79], network security [165] and molecular biology [49]. The efficiency of pooling experiments has been studied by Barillot et al. [9], Berger et al. [11], Bruno et al. [21] and Sham et al. [155]. Here we refer to the books [164] by Thai and [84, 85] by Du and Hwang for an overview.

In general, group testing can be classified into two types: sequential (also called adaptive) and non-adaptive. A sequential group testing conducts the tests one by one by using the results of previous tests to determine the pool for the next test. At the end of each round, items in negative pools are identified as negative, while those in positive pools require to be further tested. Note that an item is identified as positive if and only if it is the only item in a positive pool. Thus sequential group testing requires several testing rounds to finish a whole procedure. thereby completing the test within several rounds. In contrast, A non-adaptive group testing completes the test within one round by conducting all tests simultaneously, thus output results of the previous tests cannot be used to design the latter test. However, by taking advantages of previous testing results, sequential group testing requires fewer tests in general and was frequently used in the design of group testing since the main goal of group testing, historically, is to minimize the number of such tests in identifying all the positive samples. Group testing has been introduced to the molecular biology research field such as the DNA sequencing and DNA library screening, and thus emerging new requirements. Although minimizing the number of tests is still very important, the time required to finish the whole testing procedure must be considered since each single test may take from several hours to a day. The focus has then shifted to non-adaptive group testing where we can conduct all tests simultaneously, thus minimizing the testing time.

Formally, a non-adaptive group testing consisting of m pools and n items including d positive items can be represented by a $m \times n$ binary matrix $A = (a_{ij})$, called a pooling design, where rows represent the pools and columns represent the items. An entry $a_{ij} = 1$ if and only if the ith pool contains the jth item; otherwise, $a_{ij} = 0$. Given pooling design A, a test result of these m pools can be represented by a m-dimensional column vector $\mathbf{r}$, called the test outcome vector. $\mathbf{r}$ is a binary vector, in which 1-entry represents a positive outcome and 0-entry represents a negative one. A positive result indicates that at least one positive item exists within this pool, whereas a negative one means that all the items in the current pool are negative. That is, if $\mathbf{r}_i = 0$ then all items in row i of A are negative; if $\mathbf{r}_i = 1$ then there exists at least one positive item in row i. Note that in the non-adaptive group testing, an item can be tested in several pools at the same time. Once items are arranged into pooling design, a decoding algorithm is required, aiming to identify the positive items by using the test outcome vector $\mathbf{r}$, pooling design A and other available prior knowledge such as the upper bound of the number of positives d.

Group testing can further be classified as either combinatorial or probabilistic, according to the classification rule whether the decoding algorithm is combinatorial or probabilistic. In combinatorial group testing, it is often assumed that the upper bound of the number of positives and that of experimental errors in pooling results are known a priori. The essence of combinatorial group testing is to construct pooling designs with desired combinatoric structures that guarantee a positive detecting algorithm to be efficient and effective as long as the prior knowledge is accurate. Related studies can be found in Du and Hwang [84], [85], [29], Dyachkov et al. [46] , Macula [113], and Ngo and Du [128]. Although combinatorial group testing

is very useful in well-controlled situations, constructing a non-adaptive combinatorial group testing with the minimum number of pools is very challenging, especially when one expects a pooling design with extra desired properties such as error-tolerant ability. Moreover, uncertainty prevails and application-specific prior information is available in real applications. For example, in the problem of screening a clone map where pooling experiments are used to detect positive DNA segments, the uncertainty and prior information can be summarized as following:

- The exact value of d is usually not known with full certainty. Instead, some broad prior information about d may be known.
- There may be a connective relationship between the items, making the positive items tend to appear in a consecutive way, and the prior information of the consecutiveness may be known.
- In order to facilitate testing automations implemented by robots, random pooling design are preferred. For example, instead of using a pooling design with complicated mathematical structure,
- There may exist some errors in experiments. The test may return some false negative or false positive results. In the false negative, the pool contains some positive items but the test result is negative due to some testing errors. Likewise, in the false positive, the pool contains all negative items.
- In order to extract as much information from pooling experiments as possible, observed pooling results could be multi-leveled, for example, $r : \{0,1\}^n \to \{0,1,2,3\}$, meaning negative, weak positive, medium positive and strong positive.

In this situation where various kinds of uncertainty and prior information arise, probabilistic group testing strategy may be welcomed and useful as a complement to combinatorial approaches. Specifically, Bayesian approach is adopted to deal with the experimental uncertainty and the prior information. Motivated by this idea, we reformulate the problem of non-adaptive group testing as a learning problem. In a simplest version of the non-adaptive group testing problem, we have some unknown function, $r : \{0,1\}^n \to \{0,1\}$ where $r(\mathbf{a}) = \mathbf{a} \cdot \mathbf{c}$, where $\mathbf{a}$ represents a row of pooling design of A, $\mathbf{c} \in \{0,1\}^n$ represents the state vector of the items to be positive or negative, and the logic addition summation of logic multiplication $\mathbf{a} \cdot \mathbf{c} = (a_1 \wedge c_1) \vee (a_2 \wedge c_2) \cdots \vee (a_n \wedge c_n)$. The aim is to construct the functional r using m evaluations, each of which corresponds to a group test. More specifically, this aim is equivalent to recovering $\mathbf{c}$, that is, the set of positives. To reconstructed $\mathbf{c}$, one may rank each item according to their posterior probabilities of being positive after observing m group evaluations. This probabilistic view opened a potential to reformulate a group testing problem as a Bayesian ranking problem. Unlike a typical machine learning problem in which $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ is a random sample with large m, pooling design A has a constrained combinatorial structure that m should be as small as possible. This makes it impractical to assume that $\{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$ is a large sample. In the presence of consecutive positives, to design a Bayesian learning algorithm, we also need to consider the following questions:

1. How to appropriately express the prior information of the consecutive structure of the positives? how to translate the prior information of d and that of consecutive structure into a mathematical prior of the positives?

2. How to construct a decoding algorithm for detecting the positives from error-prone pooling results? How can the prior of the consecutive positives guide the detecting process? When random pooling designs are used, is the decoding algorithm still applicable?

3. What is the strategy for choosing the parameters that controls the generating procedure of random pooling designs? As the parameters of pooling designs vary, how can the variation influence the performance of the decoding algorithm? How many pooling results are least necessary to identify the consecutive positives?

4. Is the prior information of consecutive positives helpful to reduce the number of pools needed, compared with the case where the positives are consecutive but no prior information of consecutiveness is used?

Probabilistic group testing is developed by Bruno et al. [21], Knill et al. [101], Mezard and Toninelli [116] and Uehara and Jimbo [174]. In particular, Knill et al. [101] proposed a positive detecting algorithm called MCPD by using Markov chain Monte Carlo simulation method, in which questions 1-2 are partially discussed. To answer questions 1-4, we extend their work to the cases in which prior information of consecutive positives is available. Chapter 4 details a Bayesian framework for the problem of pooling experiments for detecting consecutive positives, in which a Bayesian machine learning algorithm is proposed to decode error-prone multi-leveled pooling results given by random designs. The choice of important parameters are also discussed within the framework. It shows some happy coincidences between theoretical computation results and previously known simulation results. Numerical simulations show that the Bayesian framework is promising to deal with the uncertainty in real settings and that the prior information is helpful to reduce the number of pools needed to detect the positives.

Chapter 2

Ranking smartphone apps based on users' behavior records

2.1 Background

Smartphone usage has grown remarkably and become a most popular choice for communication and mobile computing. Supported by an increasing convergence of mobile telecommunication devices, smart sensors and personal computers, smartphones are no longer only a modern form of cell phone, but also based on encapsulated pieces of computer program software called smartphone applications - or simply "apps" - to provide diverse functionalities and mobile services to their users.

In company with the prosperity of smartphone market and the prevalent usage of apps across smartphone users, a huge amount of interface-friendly apps have been developed to provide diverse functionality and extended capabilities of smartphones. Smartphone apps were primarily offered for general productivity, mobile entertainment and information retrieval. Users' demand and the availability of increasingly powerful hardware and user-friendly developer tools drove a rapid expansion of app development into other categories, such as health care, business, finance, tourism and education. App development has brought many profits and opportunities to app vendors and relevant business companies, and still expands at an amazing growth. A plethora of studies have focused on understanding app development and management from different perspectives for a wide range of contexts. Tucker [173] discussed the economic value of various online data for marketing, product development and so on. Recently, user reviews, comments, votes, and the like are extensively considered as elements of measures for various evaluation purposes. Chen and Liu [28] employed machine learning techniques for predicting popularity of online distributed applications based on the data collected from iTunes App Store. Other related work can be found, for example, in Bredican and Vigar-Ellis [20], Bomhold [145], Charani et al. [27], d'Heureuse et al. [42], Gavalas et al. [61], Gupta, et at. [72], Holzer and Ondrus [82], Khan et al. [97], Mo Kwon et al. [121], Oreku [131], Yan et al. [185], Yan et al. [187] and Yin, et al. [189].

However, new problems and challenges arose. Among them, how to assess the quality of smartphone apps from a user's perspective is a fundamental problem and ongoing challenge. It is fundamental because quality assessment is expected to be able to provide reliable suggestions that are useful in various problem-solving and decision-making processes involved in app development and management. For example, quality assessment would be helpful for users to discover potentially useful apps, for content providers to improve information quality of content, for developers and product managers to monitor and improve app performance, for app stores to investigate user preferences and then recommend relevant apps to relevant users. Unfortunately, the explosion in variety and number of apps makes it an ongoing

TABLE 2.1: Example of Users' Behavior Records

Device ID	App	Dec	Jan	Feb	Mar	Apri
5048···a94d	com.google.android.apps.plus	1	1	1	1	1
5048···a94d	com.twitter.android	1	1	0	0	0
5048···a96e	com.facebook.katana	1	1	1	1	0
5048···a96e	jp.mixi	1	1	1	1	1
5048···a96e	com.co_mm	1	1	0	0	0
5048···a96e	jp.ameba.candy	0	1	1	1	1
5048···a96e	com.twitter.android	0	0	1	1	1

challenge. Correspondingly, ranking methods that could perform removal of non-relevant apps and sorting of relevant apps according to their quality assessment are desired in various practical applications concerning app development and management.

This chapter first develops a ranking method for assessing app quality from a user's perspective by using user behavioral information extracted from big data of app usage. The usage data of our interest is essentially different from those that are collected from developer self-reports or user questionnaires. This brings a potential to assess app quality from a different angle and leads to a new and complementary insight. This ranking method can be used to distinguish relevant apps from non-relevant ones for further study. Then, this chapter presents another ranking method for arranging relevant apps into a linear order with their usage information. This ranking method produces an explanation-based quality assessment and can be used to mine useful knowledge from user behavioral information.

2.2 Dataset Used

App developers usually trace and study users' behaviors for management purpose in order to improve the quality of their apps as well as their service. In this section, we describe the dataset that we will use through this chapter.

FULLER Inc. provided a variety of datasets concerning app development and management. We used one of the datasets to conduct app quality assessment. From the data, it can be learned which Android device installed what Android app(s) at what month during the period from Oct. 2012 to Jun. 2013. Each piece of the data is called a user's behavior record. Particularly, a device's sequential states of whether it has a specific app installed over a period of time can be represented as a binary sequence, called a usage record. Notice that it may happen that an Android smartphone user has more than one devices and hence several computer generated device IDs. Since each device corresponds to a unique computer generated device ID, we assume that each user has only one device and hence can be labeled by a unique computer generated ID. So far, the dataset contained millions of users' behavior records, and its size is still increasing due to the arrival of new data. With focus on Android apps, we show in Tab. 2.1 a small part of the data.

Next, we introduce some notation that will be consistently used throughout this chapter, to describe the structure of the dataset. Let $\mathcal{D}$ and $\mathcal{A}$ be collections of n devices (users) and m apps, respectively. The structure of the time range consisting of l consecutive time intervals can be described as an ordered set $\mathcal{T} = (\cup_{i=1}^{l}\{T_i\}, \prec_t)$, where $T_i \prec_t T_{i+1}$ for $i = 1, \ldots, l-1$.

Let $r(D, A, T)$ be indicate function of $D \in \mathcal{D}$, $A \in \mathcal{A}$ and $T \in \mathcal{T}$ such that $r(D, A, T) = 1$ if app A can be found installed on device D at T, otherwise 0. This allows us to define the usage record of D with respect to A during $\mathcal{T}$, denoted by $r(D, A, \mathcal{T})$, as a binary sequence $(r(D, A, T_i))_{i=1}^{l} \in \{0, 1\}^l$. Moreover, without loss of generality, $\mathcal{A}$ is assumed to contain the apps that are installed by as least one user during $\mathcal{T}$. That is, for any $A \in \mathcal{A}$, it holds that $\sum_{D\in\mathcal{D}} \sum_{T\in\mathcal{T}} r(D, A, T) \geq 1$. Similarly, $\mathcal{D}$ is assumed to contain the users who have ever installed as least one app of $\mathcal{A}$ during $\mathcal{T}$. That is, for any $D \in \mathcal{D}$, it holds that $\sum_{A\in\mathcal{A}} \sum_{T\in\mathcal{T}} r(D, A, T) \geq 1$.

For any application A, let $\mathcal{D}(A)$ be the collection of active users with respect to A, i.e.,

$$\mathcal{D}(A) = \big\{D \in \mathcal{D} : \sum_{T\in\mathcal{T}} r(D, A, \mathcal{T}) \geq 1\big\}.$$

Let $\mathcal{D}(A, X)$ be the collection of active users of usage pattern $X \in \mathcal{U} \setminus \{II\}$ with respect to A, i.e.,

$$\mathcal{D}(A, X) = \big\{D \in \mathcal{D} : r(D, A, \mathcal{T}) \in X\big\}.$$

Particularly,

$$\mathcal{D}(A, II) = \mathcal{D} \setminus \mathcal{D}(A).$$

As usage records accumulate and have a sufficient length (here, we assume $l \geq 4$), though simple in form, they could be informative and expressive. However, the number of all possible usage records is also exponentially large, that is, 2^l. This makes it less appropriate to classify users if users' usage records are directly used, especially when l is large.

To ease this difficulty, the concept of usage pattern was introduced, which can be used to reduce the number of user types. The basic idea is to classify users' usage records into a small number of classes, each of which is called a usage pattern. The basic observation is that, the 1s or 0s near the end of a user's usage record can be given a more detailed meaning, capable of summarizing the user's historical behavior records. For example, given $r(D, A, \mathcal{T}) = (0, 0, 0, 0, 1, 1)$, the 1 at the fifth coordinate can be interpreted as a state that A is newly installed by D with respect to $\mathcal{T}$ and the following 1 means a state of that app A is continuously installed.

This example also casts a light on the basic idea for classifying user types according to their usage records. Formally, given $r(D, A, \mathcal{T})$ and some integer h ($1 \leq h \leq l - 3$), for each $l - h + 1 \leq i \leq l$, $r(D, A, T_i)$ will be associated with a state p_i taken from the symbol set $\{N, R, C, I, U\}$, such that

$$p_i = \begin{cases} N, & \text{if } r(D, A, T_i) = 1 \text{ and } \sum_{j=1}^{i-1} r(D, A, T_j) = 0, \\ R, & \text{if } r(D, A, T_i) = 1, r(D, A, T_{i-1}) = 0, \text{ and} \\ & \sum_{j=1}^{i-2} r(D, A, T_j) \geq 1, \\ C, & \text{if } r(D, A, T_i) = 1 \text{ and } r(D, A, T_{i-1}) = 1, \\ I, & \text{if } r(D, A, T_i) = 0 \text{ and } \sum_{j=1}^{i-1} r(D, A, T_j) = 0, \\ U, & \text{if } r(D, A, T_i) = 0 \text{ and } \sum_{j=1}^{i-1} r(D, A, T_j) \geq 1. \end{cases}$$

TABLE 2.2: Classification of Usage Records With $h = 2$

1	2	$\cdots$	$l-2$	$l-1$	l	Usage Pattern
All zeros				0	0	II
All zeros				0	1	IN
All zeros				1	1	NC
At least a one			0	1	1	RC
Any			1	1	1	CC
At least a one				0	1	UR
All zeros				1	0	NU
At least a one			0	1	0	RU
Any			1	1	0	CU
At least a one				0	0	UU

The resulting symbol sequence $p_{l-h+1}p_{l-h+2}\cdots p_l$ is called the h length usage pattern of $r(D, A, \mathcal{T})$. Intuitively speaking, N represents the state of new installation with respect to $\mathcal{T}$, R re-installation, C continuous installation, U uninstallation and I ignorance (the state of being unknown, or already known but not yet installed) with respect to $\mathcal{T}$. Therefore, any possible usage record $r(D, A, \mathcal{T}) \in \{0,1\}^l$ can be classified by their usage patterns of a prescribed length h, as long as $l \geq 4$ and $1 \leq h \leq l-3$. Denote by $\mathcal{C}(h)$ the set of all usage patterns of length h. Obviously, $\mathcal{C}(1) = \{N, R, C, I, U\}$. $\mathcal{C}(2)$ is shown in Tab. 2.2. Let $P(A, X) = |\mathcal{D}(A, X)|/|\mathcal{D}|$ be the proportion of usage pattern X with respect to A. We denote by $\mathbb{P}^A$ the distribution of usage patterns with respect to A, such that $\mathbb{P}^A(X) = P(A, X)$, for $X \in \mathcal{U}$; moreover, let $\mathcal{P}$ denote the space of all possible distribution of usage patterns, that is,

$$\mathcal{P} = \{\mathbb{P} = (P_1, \ldots, P_{|\mathcal{U}|}) \in \mathbb{R}^{|\mathcal{U}|} : \sum_{i=1}^{|\mathcal{U}|} P_i = 1, P_i \geq 0, i = 1, \ldots, |\mathcal{U}|\}.$$

It is not hard to see that $\mathcal{C}(h)$ can be constructed from $\mathcal{C}(h-1)$ and hence recursively from $\mathcal{C}(1)$. For the sake of convenience, let $\mathcal{C}_1 = \mathcal{C}(1) \setminus \{I\}$ and $\mathcal{C}_2 = \mathcal{C}(2) \setminus \{II\}$

This classification method is sensitive to both users' historical behavior records and their present records (with respect to a time range). With this classification method, users' changes of usage behavior during a period of time can be summarized and represented. By choosing a proper value of h, the total number of usage patterns $5 \times 2^{h-1}$ can be much smaller than that of all possible usage records, 2^l. Hence, h can be regarded as a trade-off between completeness of information and efficiency in expressiveness. Without loss of generality and for the sake of simplicity, we set $h = 2$ throughout this paper. Let $\mathcal{U} = \mathcal{U}_+ \cup \mathcal{U}_-$, where $\mathcal{U}_+ = \{IN, NC, RC, CC, UR\}$ and $\mathcal{U}_- = \{NU, RU, CU, UU, II\}$. A usage pattern is said to be positive if it belongs to $\mathcal{U}_+$, otherwise negative. Each usage pattern $X \in \mathcal{U}$ can be represented as a subset of $\{0,1\}^l$ according to Fig. 2.2. With this classification, we can continue to obtain the distribution of usage patterns for each app.

2.3 A ranking method for app extraction

This section constructs quality assessment measures based on users' behavior records. These measures can be used to build pointwise ranking methods for extracting relevant apps with high quality assessment.

2.3.1 Related works and new contributions

Recent studies discussing the problem of app quality assessment can be found in, for example, Chen et al. [28], Choe [30], Hale et al. [74], Huckvale et al.[83], Noh et al. [129], Patel et al. [134], Reynoldson et al. [146], Stippig et al. [160], Stoyanov et al. [161], Van Singer et al. [176] and Väätäjä [175]. Besides, there is a great effort to address the problem of how to measure the quality of use, or "usability". Several different standards or models for assessing app usability have been proposed, and relevant studies can be found, for example, in Ahmad et al. [3], Bevan [14], Finstad [50], Harrison et al [78], Noh et al. [129] and Verkasalo et al. [178]. In these studies, the prevailing methodology is mainly based on questionnaire investigation or other similar processes in which volunteers are required to response to a series of previously designed tasks. Questionnaire-based methods have an advantage in allowing users to fully express their personal evaluation on specific aspects of apps and hence lead to meaningful and reliable measurements within specific context of study. However, it is impractical to apply questionnaire-based methods to large-scale study, in consideration of the cost and time consumption of the survey process.

To conduct large-scale assessment of app quality, some authors proposed to collect user feedback data such as user reviews and star-ratings from app stores and to construct feedback-based measures. See Yin et al [189] for an example. However, feedback-based measures also rely on users' active involvement. However, in the absence of any incentive, it is not practical to expect users' active participation. In particular, there are few users who would regularly update their feedback during their use of apps.

Moreover, to construct appropriate measures for assessing the objects to be managed is usually one of the key ingredients in various domain-specific management issues. However, how to construct such measures is often nontrivial and case-dependent. Particularly, unlike ordinary commodity goods, smartphone apps have their own particularities. One obvious particularity is that most of the smartphone apps are free to install and use. In addition, smartphone apps can also be reinstalled or uninstalled repeatedly free from any restriction whenever users would like to. This allows users to change their opinions and behaviors of app usage from time to time. With these particularities, smartphone apps are essentially different from many ordinary commodity goods that we are familiar with. Consequently, how to define the semantics of goodness of free apps and how to measure the degree of goodness are the new challenges. To avoid potential shortcomings of questionnaire-based and feedback-based measures, new quality assessment measures are needed for comparative or complementary use.

In order to assess the quality of free apps, the basic idea is to convert the structural information of user resource into quality assessment measures. The rationale behind this idea is that the structure of an app's user resource is closely related to the app's degrees of popularity, attractivity and usability and hence reflects its quality assessment. With automatic data collection process, growing amounts of users' behavior records become available. This may open an opportunity to conduct large-scale assessment of app quality from a user's perspective.

The rest part of this section will be organized as follows. In section 2.3.2, we begin with a discussion of the semantics of goodness and introduce the concept of random user in order to construct a standard of comparison. Then, we give a mathematical formulation of the standard of comparison that will be used for quality assessment. In section 2.3.3, with the mathematical formulation of the standard of comparison, we propose a ranking method for extracting superior apps by constructing quality assessment measures with desirable properties. Section 2.3.4 shows the extraction results of the ranking method.

2.3.2 Standard of comparison and random user

Given a collection of apps subjected to be assessed, when we think of that a subset of apps are superior in terms of quality assessment to the other apps, the cognitive process of the judgement-making activity may roughly be expressed as follows: At the first stage, we inherently compare all the apps with a selected app, either a real one in the app store or an imaginary one in our mind, that serves as a standard of comparison. At the second stage, the apps whose quality assessment is judged above that of the standard of comparison are extracted as the superior ones, while the rest are considered inferior. In summary, judgments of quality assessment are made with reference to a contextually determined standard. In this application, we do not assume the standard app for comparison has to be real and only focus on the case where the standard app is cognitively imaginary.

If judgments of quality assessment are understood as a cognitive process, immediate questions follows:

1. How to determine the standard of comparison?

2. How to make comparison to the standard?

Here, we discuss the first question; the second one will be discussed in Section 2.3.3.

In this data-driven application, we compare apps based on users' behavior records, and hence determining the standard of a comparison should also rely on users' behavior records to decide which possible standard of comparison is the relevant one in this context. Since as is mentioned that app quality will be assessed by using $\mathbb{P}^A$, the distribution of usage patterns, for $A \in \mathcal{A}$, the standard of comparison should also be expressed in the form $\mathbb{P}^a$, for some imaginary app a. If we say a user's behavior record with respect to app A associated with a positive (negative) usage pattern as the user's expression of a positive (negative) attitude toward app A, then $\mathbb{P}^A$ can be seen as a numeric summary of all users' positive and negative attitudes toward app A; moreover, if a neutral state is assumed to be between positiveness and negativeness, then we may define the standard of comparison $\mathbb{P}^a$ as a numeric summary of users' neutral attitudes toward the imaginary app a. Then, the key to determining the standard of comparison is to model users' neutral state of attitudes.

To this end, we introduce a notion called random user. A random user is an imaginary user, denoted by R, who follows a random process to install or uninstall app $A \in \mathcal{A}$ during time range $\mathcal{T}$. The random process serves a mathematical description of the neutral state of attitudes. More concretely, behavior records $r(R, a, \mathcal{T})$ can be randomly generated, from which we can obtained $\mathbb{P}^a$.

Determining a standard of comparison and hence a corresponding random process is a pragmatic matter concerning application-dependent context such as prior information and domain-specific knowledge. In this application, we discuss a possible construction by presenting a bayesian random process consisting of two random

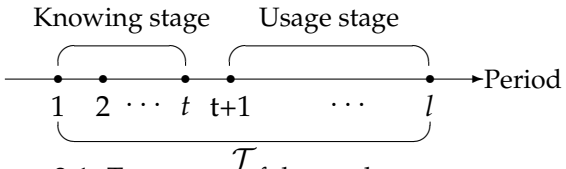


FIGURE 2.1: Two stages of the random process

stages: knowing stage and usage stage. The usage stage will proceed only if the knowing stage ends within time range $\mathcal{T}$ as is shown in Fig. 2.1.

At the knowing stage, a coin C_θ with hyper-parameter θ is first created, where θ is randomly and independently drawn from uniform distribution $U[0,1]$. Let X_θ be random variable of coin C_θ, such that

$$X_\theta = \begin{cases} 1, & \text{if } C_\theta \text{ shows a head,} \\ 0, & \text{if } C_\theta \text{ shows a tail,} \end{cases}$$

and $\Pr(X_\theta = 1) = \theta$. Then, C_θ will be randomly and independently flipped until a head is shown, but at most $|\mathcal{T}|$ times. Correspondingly, $r(R, a, T_i) = 0$ if C_θ shows a tail. If no head is shown during $\mathcal{T}$, the knowing stage ends and the usage stage does not proceed. The resulting usage record is thus $r(R, a, \mathcal{T}) = \mathbf{0}$.

The usage stage will proceed if and only if for some integer $t(\theta)$ between 1 and l such that at $T_{t(\theta)} \in \mathcal{T}$, C_θ shows the first head. When the knowing stage ends at $T_{t(\theta)}$ and the usage stage proceeds, a fair coin C_f will be used instead. Let Y be random variable of coin C_f, such that

$$Y = \begin{cases} 1, & \text{if } C_f \text{ shows a head,} \\ 0, & \text{if } C_f \text{ shows a tail,} \end{cases}$$

and $\Pr(Y = 1) = 0.5$. Denote by y the random value of Y. Then, C_f will be randomly and independently flipped, but at most $|\mathcal{T}| - i + 1$ times. Correspondingly, $r(R, a, T_j) = y_j$, for $j = t(\theta), \ldots, |\mathcal{T}|$.

For the sake of clarity, we denote by $\mathbb{Q}$ the standard distribution of usage patterns obtained from the random process that the random user follows. Let $\mathbb{Q}^\theta$ be the conditional probability distribution of usage patterns of the random user given θ is fixed and known. Noting that

$$\mathbb{Q}^\theta(X) = \sum_{T \in \mathcal{T}} \mathbb{Q}^\theta(X | T_{t(\theta)} = T) Pr(T_{t(\theta)} = T),$$

we have

$$\mathbb{Q}^\theta(IN) = \sum_{i=0}^{l-1} 0.5^{l-i}(1-\theta)^i\theta;$$

$$\mathbb{Q}^\theta(II) = \mathbb{Q}^\theta(IN) + 1 - \sum_{i=0}^{l-1}(1-\theta)^i\theta\ ;$$

$$\mathbb{Q}^\theta(NC) = \mathbb{Q}^\theta(NU) = \sum_{i=0}^{l-2} 0.5^{l-i}(1-\theta)^i\theta;$$

$$\mathbb{Q}^\theta(RC) = \mathbb{Q}^\theta(RU) = \sum_{i=0}^{l-4}(0.5^3 - 0.5^{l-i})(1-\theta)^i\theta;$$

$$\mathbb{Q}^\theta(UR) = \mathbb{Q}^\theta(UU) = \sum_{i=0}^{l-3}(0.5^2 - 0.5^{l-i})(1-\theta)^i\theta;$$

$$\mathbb{Q}^\theta(CC) = \mathbb{Q}^\theta(CU) = \sum_{i=0}^{l-3} 0.5^3(1-\theta)^i\theta.$$

Then, using the assumption that θ is randomly and independently drawn from $U[0,1]$, for $X \in \mathcal{U}$, we have

$$\mathbb{Q}(X) = \int_{\theta\in[0,1]} \mathbb{Q}^\theta(X) f_U(\theta) d\theta,$$

where $f_U(\theta) = 1$ is the density function of the uninformative kernel $U[0,1]$. In this way, we can obtain the closed form of expression about $\mathbb{Q}$.

In summary, to determine a context-dependent standard of comparison, we first interpret the standard of comparison as a neutral state with which each app will be compared, and then interpret the neutral state as a random process that captures the context-dependent characteristics of the comparison process. Consequently, there could be a variety of possible formulations to specify the random process, depending on how we incorporate our prior knowledge and beliefs into interpreting and quantifying the cognitive process of comparison. This brings a potential to construct a variety of structural null hypotheses concerning cognitive process of comparison for further study.

2.3.3 Score Function

In this part, we discuss the question of how to compare $\mathbb{P}^A$ with the standard $\mathbb{Q}$ and present the quality assessment measure we used for ranking and extracting apps.

When testing a statistical hypothesis, a null hypothesis is used as a standard to be compared, and a significance score is derived from observation data and parameters of the null hypothesis to decide whether or not to reject the null hypothesis. As has been previously mentioned, $\mathbb{Q}$ can be viewed as a metaphorically structured null hypothesis. With this idea, analogously, one possible way to compare $\mathbb{P}^A$ with the standard $\mathbb{Q}$ is to derive a score from $\mathbb{P}^A$ and $\mathbb{Q}$ that can produce a measurement of how much $\mathbb{P}^A$ deviates from the standard $\mathbb{Q}$. More concretely, we may construct such a measure by introducing a function $S : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ such that if $S(\mathbb{P}^A, \mathbb{Q})$ has a larger score value of deviation measurement, then we may say that it is more likely that $\mathbb{P}^A$ is different from $\mathbb{Q}$; moreover, also taking into account that positive (negative) attitudes $\mathcal{U}_+$ ($\mathcal{U}_-$) make positive (negative) contribution to the measurement of deviation, we may construct the score function in a ratio form in order to stress the contribution of positive attitudes over that of the negative ones. Then, if $S(\mathbb{P}^A, \mathbb{Q})$ has a larger score value, we may not only say that it is more likely that $\mathbb{P}^A$ is different from $\mathbb{Q}$ but also say that the positive part of $\mathbb{P}^A$ contributes more to the deviation measurement and hence app A is more likely an app of higher quality assessment.

If we follow the idea and clues mentioned above to construct $S(\mathbb{P}^A, \mathbb{Q})$, then we need to separate and measure the contributions of positive and negative parts of $\mathbb{P}^A$ with respect to $\mathbb{Q}$.

We first consider the separation issue. To this end, let

$$\mathbb{P}^A_+ = \sum_{X\in\mathbb{U}_+} \mathbb{P}^A(X)$$

and

$$\mathbb{P}^A_- = \sum_{X\in\mathbb{U}_-} \mathbb{P}^A(X),$$

and define

$$\mathbb{P}^A_{+,\mathbb{Q}}(X) = \begin{cases} \mathbb{P}^A(X) + \mathbb{Q}(X)\mathbb{P}^A_- & \text{if } X \in \mathcal{U}_+ \\ \mathbb{Q}(X)\mathbb{P}^A_- & \text{if } X \in \mathcal{U}_-, \end{cases}$$

and

$$\mathbb{P}^A_{-,\mathbb{Q}}(X) = \begin{cases} \mathbb{Q}(X)\mathbb{P}^A_+ & \text{if } X \in \mathcal{U}_+ \\ \mathbb{P}^A(X) + \mathbb{Q}(X)\mathbb{P}^A_+ & \text{if } X \in \mathcal{U}_-, \end{cases}$$

where $\mathbb{P}^A_{+,\mathbb{Q}}$ and $\mathbb{P}^A_{-,\mathbb{Q}}$ are called the positive augmentation and the negative augmentation of $\mathbb{P}^A$ with respect to $\mathbb{Q}$, respectively.

The rationale behind the construction of $\mathbb{P}^A_{+,\mathbb{Q}}$ can be intuitively interpreted as follows: to separate the contribution of the positive part of $\mathbb{P}^A$ from that of the negative part, we removed the negative part $\mathbb{P}^A_-$ from $\mathbb{P}^A$ in the sense that the negative attitudes were first substituted by neutral attitudes, and then the neutral attitudes were converted back into positive attitudes and negative attitudes in proportion to $\mathbb{Q}$. As a consequence, for a positive usage pattern $X \in \mathcal{U}_+$, its proportion increased by $\mathbb{Q}(X)\mathbb{P}^A_-$ and hence increased to $\mathbb{P}^A(X) + \mathbb{Q}(X)\mathbb{P}^A_-$; for a negative usage pattern $X \in \mathcal{U}_-$, its proportion decreased to $\mathbb{Q}(X)\mathbb{P}^A_-$. The resulting $\mathbb{P}^A_{+,\mathbb{Q}}$ augmented $\mathbb{P}^A$ in the sense that the positive part of $\mathbb{P}^A$ was enriched while its negative part was diluted. The construction of $\mathbb{P}^A_{-,\mathbb{Q}}$ follows a similar rationale.

Next, we consider the measurement issue. Since $\mathbb{P}^A_{+,\mathbb{Q}}, \mathbb{P}^A_{-,\mathbb{Q}} \in \mathcal{P}$, the contributions of the positive part and negative part of $\mathbb{P}^A$ with respect to $\mathbb{Q}$, denoted by S^A_+ and S^A_-, can be measures by the l_2 distance between the augmented distributions of $\mathbb{P}^A$ and $\mathbb{Q}$,

$$S^A_+ = \left(\sum_{X \in \mathcal{U}} |\mathbb{P}^A_{+,\mathbb{Q}}(X) - \mathbb{Q}(X)|^2 \right)^{\frac{1}{2}}$$

and

$$S^A_- = \left(\sum_{X \in \mathcal{U}} |\mathbb{P}^A_{-,\mathbb{Q}}(X) - \mathbb{Q}(X)|^2 \right)^{\frac{1}{2}},$$

respectively. Finally, one possible construction of the score function S in the ratio form can be given by

$$S(\mathbb{P}^A, \mathbb{Q}) = \frac{S^A_+}{S^A_-}.$$

The score function S has two desirable properties.

Property 1 For any $\mathbb{Q} \in \mathcal{P}$,

$$S(\mathbb{Q}, \mathbb{Q}) = 1.$$

Property 2 For any $\mathbb{Q} \in \mathcal{P}$,

$$S(\mathbb{P}^A, \mathbb{Q}) = \begin{cases} 0, & \text{if } \mathbb{P}^A_+ = 0, \\ \infty, & \text{if } \mathbb{P}^A_+ = 1. \end{cases}$$

Property 1 means that the constructed score function will keep the discrimination threshold constant and invariant for any selected standard of comparison.

App	12.10-13.3	12.11-13.4	12.12-13.5	13.1-13.6
Facebook	0.83	0.80	0.78	0.77
Twitter	0.75	0.76	0.77	0.78
Mixi	0.38	0.35	0.32	0.29
Gree	0.17	0.20	0.21	0.23
Comm	0.16	0.15	0.13	0.11
Ameba	0.13	0.12	0.12	0.12
Bump	0.09	0.08	0.07	0.07
Mbga	0.05	0.05	0.05	0.05
Instagram	0.05	0.05	0.05	0.05
Sockets Live	0.03	0.03	0.03	0.03
Twicca	0.03	0.03	0.02	0.02
MixiSH	0.03	0.02	0.02	0.02
Saitosan	0.02	0.02	0.02	0.02
2chMate	0.02	0.02	0.02	0.02

TABLE 2.3: The apps with highest score values of S

Property 2 means that the constructed score function will assess an app as one of the worst apps in terms of quality assessment in any sense of standard of comparison, if no user has a positive attitude toward the app; similarly, the constructed score function will assess an app as one of the best apps in terms of quality assessment in any sense of standard of comparison, if all users have a positive attitude toward the app.

Ideally, with properties 1 and 2, we can use the score function S and a properly constructed $\mathbb{Q}$ to assign a score value to each app. The apps whose score value is larger than 1 are extracted as the superior ones, while the rest are considered inferior.

2.3.4 Experimental Result

The dataset used is a collection of Android users' behavior records obtained during the period from Oct. 2012 to Jun. 2013, provided by FULLER, Inc. It contained over three millions of users' behavior records. For the sake of convenience, the dataset was divided according to the categories of apps, and the social category was used for numeric experiment. In the experiment, we set $l = 6$ and $h = 2$. The average probability mass of the random user is that $\mathbb{Q}(IN) = 0.0447$, $\mathbb{Q}(II) = 0.1876$, $\mathbb{Q}(RC) = \mathbb{Q}(RU) = 0.0755$, $\mathbb{Q}(NC) = \mathbb{Q}(NU) = 0.0328$ and $\mathbb{Q}(CU) = \mathbb{Q}(CC) = 0.1$. The top fourteen apps assessed by the score function were extracted and shown in Table. 2.3.

By comparing the rank result given by Google play, we found that eight out of the fourteen apps were ranked within top ten on the Google play's ranking result. our ranking result might be improved if we removed from consideration the pre-installed apps such as MixiSH, because these apps were excluded from Google play's ranking results. This ranking method produces pointwise quality assessment measurements according to which superior apps can be extracted.

2.4 A ranking method for app ordering

This section will present a ranking method for arranging a collection of apps into a linear order by using the same dataset of users' behavior records. The ranking method incorporates the knowledge of life cycle management of app development as prior information and produces pairwise quality assessments between apps, which in return can be used to mine useful knowledge from user behavioral information.

2.4.1 Related works and new contributions

The work in this section is inscribed within an emerging body of work that analyzes user behavior and mining useful knowledge from user behavioral data of apps. For example, the studies investigating the possibilities of characterizing and predicting user behavior using user behavioral information can be found in Benbunan [10], Do and Gatica-Perez [44], Taylor and Levin [56], Liao et al. [106] and Xu et al. [184]. Based on user behavioral data analysis, there are also other useful applications, such as discovering associations between user interactions, inferring social network structure, tracking network information flow, modeling user preferences, characterizing human mobility and designing app recommender system. These studies can be found in, for example, Costa-Montenegro et al. [35], Eagle et al. [48], Perunani and Tabourier [137], Shi and Ali [157], Zarmpou et al. [190] and Zhu et al. [193].

In the previous section, we have discussed a pointwise-comparison-based ranking method for extracting superior apps. In this section, we will discuss a pairwise-comparison-based ranking method for app quality assessment by using the dataset of users' behavior records. Instead of converting distributions of usage patterns into a series of numeric scores, this ranking method will consider more detailed information of usage patterns and then produce a linear order of apps according to their quality assessments. More importantly, domain-specific knowledge and common sense, serving as prior information, can be incorporated into the ranking process, which provides an opportunity to produce interpretable ranking results that may be useful for mining knowledge from users' behavioral information. As a consequence, the pairwise-comparison-based ranking method can be seen as a refinement of the previously proposed pointwise-comparison-based ranking method.

It can be learned from the dataset what usage pattern each user assigns to each app. With this observation, if usage patterns themselves can be properly ordered, then the information of how each user will rank the apps is also available. As a consequence, in order to arrange a collection of apps into a linear order from the users' perspective, one possible idea is to view the desirable linear order of apps as an observable representation of users' latent aggregate preference obtained by balancing users' conflicting individual preferences as far as possible. To realize this idea, we have to discuss the following three questions:

- How to assign a linear order to usage patterns?
- How to get a pairwise comparison between two apps by balancing users' conflicting individual preferences?
- How to merge all pairwise comparisons between apps into a linear order?

The rest part of this section will be organized as follows. In section 2.4.2, we discuss the first question by taking into consideration domain-specific knowledge and common sense about life cycle management of app development, and then based on

the linear ordering of usage patterns we propose a model of users' individual preferences. In section 2.4.3, the second question is discussed from a Bayesian perspective. That is, users' conflicting individual preferences of a pair of apps are formulated as a pair of probability distributions that summarize users' posterior degree of beliefs about the comparison between the pair of apps. With this knowledge representation of users' conflicting individual preferences for a pair of apps, the third question is interpreted as a stochastic acyclic subgraph problem on a complete digraph with each arc associated with a constant weight obtained from a known probability distribution that summarizes users' posterior degree of beliefs about the comparison between the pair of apps. Section 2.4.4 shows the extraction results of the ranking method.

2.4.2 Users' individual preferences

This section introduces a deterministic model of users' individual preferences of the apps to be ranked. The main idea is to give an ordinal structure to the set of usage patterns, which can be used to reflect users' satisfaction level during their usage of the apps. And, the preference relation of a user's preference of a pair of apps that the user has ever installed within a given period of time can be determined by comparing the user's satisfaction level of the two apps.

Linear ordering of usage patterns

To model a user' individual preferences of apps based on the user's usage records, one possible way is to introduce an ordinal structure on $\{0,1\}^l$. With this ordinal structure, a user's individual preferences of two apps can be obtained by reading the user's usage patterns of the apps. Recall that usage pattern is a compact expression of a user's behavior record. Thus, it would be more convenient to assign an ordinal structure to $\mathcal{C}(h)$. In particular, we discuss the ordinal structure of $\mathcal{C}(2)$. Let $\prec_2$ be the order relation on $\mathcal{C}(2)$. At the vey beginning, however, it can be noticed that what order II should be assigned under $\prec_2$ has to be discussed. Without having ever installed A within $\mathcal{T}$, user D may hardly make any meaningful comparison between A and those apps that the user has ever been installed within $\mathcal{T}$. Therefore, it may be reasonable to remove usage pattern II from our consideration. Denote $\mathcal{C}_2 = C(2)\backslash\{II\}$. Similarly, we can also define C_k with other values of k. With this presumption, we give two possible ordinal structures restricted to $\mathcal{C}_2$.

The first construction of the linear order of $\mathcal{C}_2$ is given as follows:

$$UU \prec_2 NU \prec_2 RU \prec_2 CU \prec_2 IN$$

$$\prec_2 UR \prec_2 NC \prec_2 RC \prec_2 CC.$$

This construction is due to a lexicographic order $\prec_L$ on the set of all possible usage records (i.e., $\{0,1\}^l$), that is, for any two distinct vectors $r, r' \in \{0,1\}^l$, we say $r \prec_L r'$ if there exists a l_0 , where $0 \leq l_0 \leq l-1$, such that $(r)_j = (r')_j$, for $j = l_0+1, ..., l$, but $(r)_{l_0} = 0$ and $(r')_{l_0} = 1$. Then, recalling that usage pattern $\mathbf{p} \in \mathcal{C}_2$ can be viewed as a subset of $\{0,1\}^l$, for any two distinct usage patterns $\mathbf{p}, \mathbf{p}' \in \mathcal{C}_2$ we say $\mathbf{p} \prec_2 \mathbf{p}'$ if and only if $\forall r \in \mathbf{p}$ and $\forall r' \in \mathbf{p}'$, we have $r \prec_L r'$. In other words, this construction can be viewed as a lexicographic order on $\mathcal{C}_2$, which is obtained by using a linear order on $\mathcal{C}_1$, that is,

$$U \prec_1 N \prec_1 R \prec_1 C.$$

More concretely, for any two distinct usage patterns $p_2p_1, p_2'p_1' \in \mathcal{C}_2$, where $p_1, p_2, p_1', p_2' \in \mathcal{C}_1$, we say $p_2p_1 \prec_2 p_2'p_1'$ if and only if $p_1 \prec_1 p_1'$, or $p_1 = p_1'$ and $p_2 \prec_1 p_2'$.

The second construction of the linear order of $\mathcal{C}_2$ is given as follows:

$$CU \prec_2 RU \prec_2 NU \prec_2 UU \prec_2 CC$$

$$\prec_2 RC \prec_2 NC \prec_2 UR \prec_2 IN.$$

In this construction, usage pattern is thought of as the observable reflection of the change of satisfaction level in the last two consecutive time intervals. Notice that, the usage patterns representing a transition from uninstallation (or ignorance) to installation will be associated with higher ranks than any other patterns, and that those patterns representing a transition from installation to uninstallation (or ignorance) will be associated with lower ranks. Moreover, if a usage pattern represents fewer installation states in a near past period, then it will be associated with a higher rank. We call the first construction and the second one the lexicographic order of $\mathcal{C}_2$ and the jump order, respectively.

The basic idea of constructing the linear orders of $\mathcal{C}_2$ comes from pattern analysis of product life cycle. An overview of the theory of product life cycle can be found in [80, 99, 105, 130, 147]. To explain the basic idea of the orders, we employ the classical theory of product life cycle where entire industry of product life cycle can be roughly divided into four stages, based on a bell-shaped curve of sales volume. As a digital product of information technology, sales volume is not a standard measure to evaluate the successfulness of smartphone apps. Instead, quantitative indices of market share and structure of new users and loyal users are usually more meaningful measures for evaluating smartphone apps. Therefore, we divide the product life cycle of smartphone apps into similar four stages: market development, market growth, market maturity and market decline.

In the market development stage, when a new product is first released to market, before there is a proved demand for it, and often before it has been fully proved out technically in all respects. The number of users is low and creeps along slowly. In this stage, new smartphone apps usually need a period of time to self-iteratively update in order to adjust themselves for meeting the needs of targeted users, so smartphone app vendors not only pay attention to absorbing new users while losing as few new users as possible. In the stage of market growth, demand begins to accelerate and the size of the total market expands rapidly. In this stage, smartphone app vendors simultaneously would like to expand the market share of their products and also make profits from loyal users. Hence, absorbing new users and loss of loyal users are attached to great importance. This observation thus gives the jump order to capture the characteristics of app market growth,

$$CU \prec_2 RU \prec_2 NU \prec_2 UU \prec_2 CC$$

$$\prec_2 RC \prec_2 NC \prec_2 UR \prec_2 IN.$$

In the mature stage, market demand levels off and grows at a low rate. In this stage, market segmentation has been reinforced and targeted users are converted into loyal users. To increase profits, smartphone app vendors pay more attention to improving usage experience of loyal users instead of absorbing new users. This observation thus gives the lexicographic order to capture the characteristics of mature market,

$$UU \prec_2 NU \prec_2 RU \prec_2 CU \prec_2 IN$$

$$\prec_2 UR \prec_2 NC \prec_2 RC \prec_2 CC.$$

In the decline stage, the product begins to lose consumer appeal and sales drift downward. In this stage, loyal users begin to lose their interest to old apps and seek for modern substitutes, while in order to maximize profit, app vendors also stop to update smartphone apps at some point where margin profit decreases fast but cost steadily increases. At the time being of our analysis, we are interested in analyzing popular smartphone apps and they are not in their decline stage, so we may focus on the lexicographic order and jump order for further analysis.

As there could be a variety of possible formulations to specify the random process for a standard of comparison discussed in the previous section, there could also be a variety of possible formulations to specify a meaningful ordinal structure assigned to $\mathcal{C}_2$, depending on how we incorporate our prior knowledge and beliefs into interpreting the cognitive process of ordering. This knowledge representation brings a potential to construct a variety of structural hypotheses concerning cognitive process of comparison for further study.

A model of a user's individual preferences

This part will propose a model of a user's individual preferences of two apps with the ordinal structure of $\mathcal{C}_2$. If a ordinal structure of $\mathcal{C}_k$ for other values of k can be obtained, then this method can be generalized to model a user's individual preferences with the ordinal structure of $\mathcal{C}_k$.

Let $\varphi_2 : \{0,1\}^l \to \mathcal{C}_2$ be the classification function of behavior records, that is, mapping a behavior record to its usage pattern of length 2. Denote by $\prec_{D,\mathcal{T}}$ (without causing confusion, short for $\prec_D$) the preference relation of D's individual preferences of two apps with respect to $\mathcal{T}$ and by $\sim_{D,\mathcal{T}}$ (without causing confusion, short for $\sim_D$) the relation of user D's indifference between two apps with respect to $\mathcal{T}$. We say that

$$\begin{aligned} A_i \prec_D A_j \Leftrightarrow r(D, A_i, \mathcal{T}) \neq \mathbf{0}, r(D, A_j, \mathcal{T}) \neq \mathbf{0}, \text{ and} \\ \varphi_2(r(D, A_i, \mathcal{T})) \prec_2 \varphi_2(r(D, A_j, \mathcal{T})), \end{aligned}$$

and that

$$\begin{aligned} A_i \sim_D A_j \Leftrightarrow r(D, A_i, \mathcal{T}) \neq \mathbf{0}, r(D, A_j, \mathcal{T}) \neq \mathbf{0}, \text{ and} \\ \varphi_2(r(D, A_i, \mathcal{T})) = \varphi_2(r(D, A_j, \mathcal{T})). \end{aligned}$$

Similarly, we can also define $\succ_D$. Intuitively speaking, we say there exists a preference relation of a user's individual preferences between two apps with respect to $\mathcal{T}$ only if the user has, at least once, ever installed both applications within $\mathcal{T}$ (but, needless to have to install both in the same time interval). Let $\mathcal{D}(A_i, A_j) = \{D \in \mathcal{D} : r(D, A_i, \mathcal{T}) \neq \mathbf{0} \text{ and } r(D, A_j, \mathcal{T}) \neq \mathbf{0}\}$ be the collection of the active users with respect to A_i and A_j.

2.4.3 Users' aggregate preference

Denote by $\prec$ the relation of users' aggregate preference between two application. $A \prec A'$ means A' is preferred to A by $\mathcal{D}$. Our goal is to find a linear order $A_{\sigma(1)} \prec \ldots \prec A_{\sigma(m)}$, for some permutation σ on the set $\{1, 2, \ldots, m\}$, which is consistent with users' behavior records as far as possible. In this section, we first model users'

TABLE 2.4: Divergences of users' individual preferences

Scenario 1	$a_{ij} = 70$	$b_{ij} = 10$	$c_{ij} = 10$
Scenario 2	$a_{ij} = 2$	$b_{ij} = 0$	$c_{ij} = 0$
Scenario 3	$a_{ij} = 0$	$b_{ij} = 0$	$c_{ij} = 0$
Scenario 4	$a_{ij} = 21$	$b_{ij} = 50$	$c_{ij} = 20$

aggregate preference between two apps from a Bayesian view, and then interpret the app ranking problem as a stochastic acyclic subgraph problem.

Divergence of users' individual preferences

As has been discussed, usage patterns can be viewed as ordinal numbers, which leads to a deterministic model of the preference relation of users' individual preferences between two apps. However, when we proceed to discuss the relation of users' aggregate preference between two apps, uncertainties arise due to the divergence of users' individual preferences. Therefore, a compromise between users' conflicting preferences has to be introduced. This raises the problem of how to summarize and represent the preference relation of users' aggregate preference between two apps that could meet users' divergent individual preferences as far as possible.

In this application, due to the existence of the neutral users and the ignorant users, we have to deal with users' conflicting preferences in the context of permitting users' indifference and ignorance. To investigate with this problem, we begin with considering several artificial scenarios. For convenience, define $a_{ij} = |\{D \in \mathcal{D} : A_i \prec_D A_j\}|$, $b_{ij} = |\{D \in \mathcal{D} : A_i \sim_D A_j\}|$, and $c_{ij} = |\{D \in \mathcal{D} : A_i \succ_D A_j\}|$. Imagine that, A_i and A_j are compared by $|\mathcal{D}(A_i, A_j)| = 100$ users. Then, we discuss the scenarios shown in Tab. 2.4.

In Scenario 1, since 70 out of the 90 users prefer A_j to A_i , A_j is more likely to be preferred. It also allows a degree of possibility that A_i could be preferred to A_j . In Scenario 2, however, we would have few confidence in concluding so with only two observations, even though the users both prefer A_j to A_i . Scenario 3 looks extreme but is still possible to happen, for example, if both apps are relatively new. In this case, we can hardly say anything since no evidence is available to support any conclusion. In scenario 4, it is more comfortable to conclude that A_i is almost as good/bad as A_j, although the number of the users who prefer A_j to A_i is slightly larger. From these scenarios, we can observe that : (1) the percentage $\frac{a_{ij}}{a_{ij}+b_{ij}+c_{ij}}$ is not always practical to reflect the relation of users' aggregate preference; (2) the number of the ignorant users will influence the degree of our certainty to draw a conclusion; (3) $A_i \prec A_j$ and $A_j \prec A_i$ are not mutually exclusive events due to the appearance of the neutral users.

Comparison of two applications

We assume that there is an underlying but unknown probability p_{ij} that the relation of users' aggregate preference is $A_i \prec A_j$. The relations of users' individual preferences obtained from their behavior records are thought of a random sampling of $|\mathcal{D}(A_i, A_j)|$ independent Bernoulli trials with an identical probability of success p_{ij}. Formally, for any $D \in \mathcal{D}(A_i, A_j)$, let R_{ij}^D be a Bernoulli trial with respect to A_i and A_j such that

$$R_{ij}^D = \begin{cases} 1, \text{ if } A_i \prec_D A_j, \\ 0, \text{ otherwise,} \end{cases}$$

and $\Pr(R_{ij}^D = 1) = p_{ij}$, where 1 means "success" and 0 means "failure". If we further assume that A_i and A_j are independently compared, then the likelihood function of s successes and f failures (notice that $|\mathcal{D}(A_i, A_j)| = s + f$) given $p_{ij} = x$ can be given in the form

$$l(s, f|p_{ij} = x) = x^s(1-x)^f.$$

And we would like to estimate p_{ij} with s and f.

When the value of $s + f$ is very small, however, the naive estimate of p_{ij} by the percentage $\frac{s}{s+f}$ could be misleading. Here, we take Scenario 2 for an example. In this case, based only on two observations, $\hat{p}_{ij} = 1$ is nevertheless a bad estimate. This is because we usually begin our estimation with an amount of prior belief concerning prior expectations and uncertainties. For example, before estimating p_{ij} we may have thought of that one app is quite unlikely to be completely preferred to another one, unless sufficiently convincing evidence has already been provided before our estimation. To overcome this difficulty, we adopt a Bayesian approach to deal with users' conflicting individual preferences. For an overview of Bayesian statistics, we refer the books [72] and [93].

To begin with, p_{ij} is assumed to be distributed according to a beta distribution. Suppose that a continuous random variable X has a beta distribution with parameter α and β, where $\alpha > 0$ and $\beta > 0$. Then the probability density function of X has the following form

$$f(x|\alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}, \quad 0 \le x \le 1,$$

where $B(\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}$ is the beta function and where $\Gamma(\alpha)$ is the gamma function:

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1}e^{-x}dx.$$

The mean and variance of the beta random variable X are $\mu = \frac{\alpha}{\alpha+\beta}$ and $\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$, respectively.

The beta distribution is often viewed as a probability distribution of probabilities, and plays a crucially important role in many Bayesian statistical analyses. Here we refer the book [72] for an overview. Since the beta distribution can take a variety of forms, it can be used for summarizing our prior belief of p_{ij} to be estimated. More concretely, the values of α and β can be thought of as α successes and β failures that have been imaginarily observed, and thus it allows us to express our prior expectations and uncertainties with properties specified by the values of α and β.

Moreover, it is also very convenient to update our expectations and uncertainties after a collection of observations is obtained, due to the fact that beta distribution is a conjugate prior with respect to the likelihood function of Bernoulli trials. Assume that the prior probability distribution of p_{ij} is beta(α_{ij}, β_{ij}). After a_{ij} successes and $b_{ij} + c_{ij}$ failures are observed, the posterior probability distribution of p_{ij} is given by Bayes' theorem (in the form of density probability function),

$$
\begin{aligned}
& f(p_{ij} = x | a_{ij}, b_{ij} + c_{ij}) \\
=& \frac{f(x|\alpha_{ij}, \beta_{ij}) l(a_{ij}, b_{ij} + c_{ij}|x)}{\int f(x|\alpha_{ij}, \beta_{ij}) l(a_{ij}, b_{ij} + c_{ij}|x) dx} \\
=& \frac{\frac{x^{\alpha_{ij}-1}(1-x)^{\beta_{ij}-1}}{B(\alpha_{ij},\beta_{ij})} x^{a_{ij}} (1-x)^{b_{ij}+c_{ij}}}{\int_0^1 \frac{y^{\alpha_{ij}-1}(1-y)^{\beta_{ij}-1}}{B(\alpha_{ij},\beta_{ij})} y^{a_{ij}} (1-y)^{b_{ij}+c_{ij}} dy} \\
=& \frac{x^{\alpha_{ij}+a_{ij}-1}(1-x)^{\beta_{ij}+b_{ij}+c_{ij}-1}}{B(\alpha_{ij}+a_{ij}, \beta_{ij}+b_{ij}+c_{ij})},
\end{aligned}
$$

which is another beta distribution with parameters $\alpha_{ij} + a_{ij}$ and $\beta_{ij} + b_{ij} + c_{ij}$, summarizing our posterior expectations and uncertainties. In this way, each relation of users' aggregate preference can be associated with a beta distribution, representing our posterior belief of the underlying probability that the relation will happen.

If the preference relation between $A_i \prec A_j$ and $A_j \prec A_i$ has to be decided, then their associated beta distributions may be useful for a comparison. For example, if both distributions are highly concentrated at their mean values with distinguishably non-overlapping support, then one may conclude that the one with smaller mean value is unlikely to be preferred to the other. This measure usually called mean-variance ratio MVR, written as $\frac{\mu}{\sigma}$, seems proper to be used to capture this idea and hence may provide a useful criterion for dealing with the case where the non-overlapping support is less distinguishable. For example, if beta(1, 1) is used as the priors of the relations $A_i \prec A_j$ and $A_j \prec A_i$, then the MVRs of $A_i \prec A_j$ and $A_j \prec A_i$ are $\sqrt{\frac{(3+a_{ij}+b_{ij}+c_{ij})(1+a_{ij})}{1+b_{ij}+c_{ij}}}$ and $\sqrt{\frac{(3+a_{ij}+b_{ij}+c_{ij})(1+c_{ij})}{1+a_{ij}+b_{ij}}}$, respectively. Notice that, if $a_{ij} > c_{ij}$ is assumed, then $\frac{1+a_{ij}}{1+b_{ij}+c_{ij}} > \frac{1+c_{ij}}{1+a_{ij}+b_{ij}}$ suggests that $A_i \prec A_j$ be chosen since it is more likely to happen. This choice coincides with the majority rule if the neutral users are removed from consideration.

Stochastic acyclic subgraph problem

Before discussing how one may rank the apps into a linear order based on pairwise comparisons of the apps, we first introduce some necessary notation concerning digraphs. A strict digraph $G = (V, E)$ consists of a finite nonempty set V of nodes and a set E of arcs that are ordered pairs of different elements of V without any loop or parallel arc. If (u, v) is an arc then (u, v) is said to go from u to v. We also say u dominates v.

If $G = (V, E)$ is a digraph, then every digraph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$ is called a subdigraph of G. A nonempty set of arcs

$$P = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{k-2}, v_{k-1}), (v_{k-1}, v_k)\}$$

in $G = (V, E)$ such that $v_i \neq v_j$ for $i \neq j$ is called a (v_1, v_k)-dipath of length $k - 1$. If P is a (v_1, v_k)-dipath and $(v_k, v_1) \in E$ then $C = P \cup (v_k, v_1)$ is called a k-dicycle. A digraph $G = (V, E)$ which contains no dicycle is called acyclic. A standard instance of (weighted) acyclic subgraph problem can be described as follows. We are given a digraph $G = (V, E)$ with constant arc weight w_{ij} for every $(i, j) \in E$, and we look for an acyclic subdigraph $G' = (V', E')$ of G such that $w(B) := \sum_{(i,j)\in E'} w_{ij}$ is maximized.

With the model of users' aggregate preference of two apps, we interpret the app ranking problem as a stochastic acyclic subgraph problem on a complete digraph with each arc associated with a weight distributed according to a beta distribution. To see this, let $G = (V, E)$ be a complete digraph, where $|V| = |\mathcal{A}|$, node v_i of G represents app A_i, and arc $(v_i, v_j) \in E$ represents the relation $A_j \prec A_i$, for all $i \neq j$. Each arc $(v_i, v_j) \in E$ is associated with a beta distribution Beta($\alpha_{ji} + a_{ji}, \beta_{ji} + b_{ji} + c_{ji}$), representing our posterior belief of that $A_j \prec A_i$ will hold, where α_{ji} and β_{ji} specify our prior belief. Based on the information of the beta distributions associated to the arcs, we are to find an acyclic subgraph $G'(V', E')$ and hence a linear order of $\mathcal{A}$ that meets our posterior belief of users' aggregate preference as far as possible. This problem is stochastic, since each arc weight is a random variable of beta distribution; it is combinatorial, since for all $i \neq j$ either $(u_i, u_j) \in E'$ or $(u_j, u_i) \in E'$ will be chosen, but not both.

Deterministic reformulation method is often used to obtain a workable solution of stochastic combinatorial problem. In this application, a possible way of doing this is to provide a reasonable criterion of "optimality" for choosing the optimal acyclic subgraph. Recall that MVR is a criterion for comparing two apps, reflecting our posterior degree of "expectation per unit of uncertainty". Therefore, MVR may be used as a measure for choosing the linear order between apps based on their pairwise comparisons. To do this, each arc $(v_i, v_j) \in E$ is associated with weight w_{ij}, which is the the MVR of the beta distribution associated to the relation $A_j \prec A_i$, that is,

$$w_{ij} = \sqrt{\frac{\alpha_{ji} + a_{ji}}{\beta_{ji} + b_{ji} + c_{ji}}(\alpha_{ji} + \beta_{ji} + a_{ji} + b_{ji} + c_{ji} + 1)}.$$

Then, the stochastic acyclic subgraph problem can be converted to a deterministic acyclic subgraph problem, given by

$$\begin{aligned}
\max \quad & \frac{2}{m(m-1)} \sum_{i \neq j} w_{ij} x_{ij} \\
\text{s.t.} \quad & x_{ij} + x_{ji} = 1, \text{ for all distinct pairs of } i \text{ and } j, && \text{(Type I)} \\
& x_{ij} + x_{jk} + x_{ki} \leq 2, \text{ for all distinct triples of } i, j \text{ and } k, && \text{(Type II)} \\
& x_{ij} \in \{0, 1\}, \text{ for all distinct pairs of } i \text{ and } j, && \text{(Type III)}
\end{aligned}$$

where Type I together with Type III constraints means a binary choice has to be made between each pair of apps; Type II constraints are used to remove cyclical relations between any subset of apps and hence restrict the search of optimal solution within the space of all possible linear orders. The problem has been well studied and has many important applications. The study of acyclic subgraph problem can be found in, for example, Garey and Johnson [60], Grötschel et al. [69], [70] and [71], and Pedings et al. [136].

2.4.4 Experimental result

This part shows the experimental results of the ranking method for app ordering and also explains how the ranking results can be used to mine potentially useful knowledge from the users' behavior records. As has been mentioned, since the pairwise-comparison-based ranking method can be seen as a refinement of the previously proposed pointwise-comparison-based ranking method, we apply the pairwise-comparison-based ranking method to sort the superior social apps extracted

TABLE 2.5: The ranking results of SR, LR and JR

App	SR	LR	JR
Twitter	1	3	2
Facebook	2	2	4
Mixi	3	4	8
Gree	4	10	12
Ameba	5	7	9
Comm	6	14	14
Bump	7	8	11
Mbga	8	13	10
Instagram	9	12	1
PersonalSpace	10	6	3
Sockets Live	11	1	7
Twicca	12	9	5
MixiSH	13	11	13
2chMate	14	5	6
Saitosan	15	15	15

by the pointwise-comparison-based ranking method. The ranking result obtained by the pointwise method assigned a score ranking (SR) number to each of the extracted apps; the ranking results obtained by the lexicographic order and by the jump order are denoted by LR and JR, respectively. In the experiment, the data collected from January 2013 to June 2013 were used for extracting superior social apps; moreover, uninformative priors were used for specifying our prior, that is, $\alpha_{ij} = \beta_{ij} = 1$, for all distinct $i, j = 1, ..., 15$. Then, the MVRs between the extracted apps can be efficiently obtained, due to the fact that most of the Android users only use a few of the apps of the same category. Finally, to obtain LR and JR of the extracted apps, we solved the acyclic subgraph problem on a weighted complete digraph of 15 nodes by using the commercial software Xpress. In the experiment, the exact solutions can be obtained within 0.1 second on a regular PC. Tab. 2.5 shows the results of SR, LR and JR.

First of all, from the experiment results, it can be seen that the ranking results obtained by the pairwise-comparison-based method are different from those obtained by the pointwise-comparison-based method. This suggests that the two ranking methods may be essentially different and hence may be for complementary use. Secondly, the ranking results obtained by LR are also different from those obtained by JR, which suggests that different ordinal structures may lead to different ranking results, even with the same ranking process and algorithm. This observation is desirable and appealing, because LR and JR are constructed to incorporate different prior knowledge and hence to reflect different aspects of app quality assessment.

However, since neither target ranking nor a single class label is available, there is no immediate performance measure to calibrate the ranking procedure. This is, to testify the statistical significance of the ranking results is not possible. To overcome this problem and evaluate our ranking method, we turn to testifying practical significance of the ranking results by using public open information. In particular,

this strategy may be possible since the extracted apps are daily used social apps with great popularity and hence useful information may be available for evaluation purpose. Since the lexicographic order and the jump order correspond to mature stage and growth stage, respectively, they can be used in a complementary way to identify potentially promising apps of high quality assessment, especially those ones still in their rapid growth stage. From Tab. 2.5, Twitter, Facebook and Instagram can be identified as the superior apps. Especially, Instagram seemed to be the most promising one as it probably still experienced a fast growth far from maturity. In particular, we are interested in the question of whether Instagram was indeed a promising one in the real-world sense. We investigated public information source to verify whether our method works. According to [52] and other related commercial reviews, we learned the following news about Instagram:

- In April 2012, Facebook, the social network giant was about to merge with the photo-sharing platform Instagram, a company that only employed 13 people and so far did not generate any revenue. However, Mark Zuckerberg, the founder and CEO of Facebook believed that the company he was about to acquire was of great value to his own firm. In previous negotiations, the sum of 1 billion was mentioned as a price for Instagram. The 28-year-old Zuckerberg had to act quickly. He wanted to finish the merger before the Initial Public Offering of Facebook scheduled for May 2012. Zuckerberg knew that other services like Google and Twitter were also interested in Instagram.

This information supports the hypothesis that Instagram was indeed a promising app of high quality assessment. The ranking method seems able to provide a workable ranking result for mining useful knowledge from the dataset of users' behavior records.

2.5 Conclusion

In this chapter, we discussed the problem of ranking smartphone apps according to their quality assessment from a user's perspective based on large-scale users' behavioral information which can automatically be collected without the requirement of users' active participation. The dataset we used contained millions of users' behavior records. At the data preprocessing stage, we proposed a method to classify users into a controllable number of types, called usage patterns, based on their behavioral data of app usage. With the processed data, two ranking methods were proposed, one for extracting superior apps and the other for ordering the extracted apps. In both methods, context-dependent prior information as well as domain-specific knowledge was summarized and represented from a Bayesian perspective. The resulting knowledge representation played an important role both in the preference learning phase and in the interpretation phase of the ranking methods.

In order to extract superior apps, a cognitive process of pointwise comparisons was first discussed and mathematically formulated by introducing the notions: standard of comparison and random user; then, a score function with desirable properties were constructed and used to assign a value of quality assessment measurement to each of the apps to be ranked. More concretely, the rationale behind this ranking method is that (1) an app's distribution of usage patterns is an observable reflection of quality assessment; (2) if a standard of comparison can be constructed, then the apps whose distribution is "above" the distribution of the standard can be extracted as the desirable superior apps; (3) a score function with desirable properties is thus

needed to quantify how much a distribution deviates from another. Data experiment showed that the pointwise-comparison-based ranking method is efficient and effective to extract superior apps. Specifically, the score function $S(\mathbb{P}, \mathbb{Q})$ can be seen as a class of dissimilarity distance functionals used in a clustering analysis, where $\mathbb{Q}$ are the parameters specifying the standard of comparison. Although the construction of S and $\mathbb{Q}$ is an application-specific matter, the basic idea and rationale for constructing a class of context-dependent score functions with desirable properties may be analogously applied and hence properly generalized for other similar applications. This provides a potential to construct a functional space of application-specific hypotheses for various knowledge-driven ranking tasks of automatic knowledge discovery.

In order to arrange the extracted apps into a linear order, a cognitive process of comparisons from an end-user's perspective was first discussed and mathematically formulated by assigning an ordinal structure to the set of usage patterns, and then based on the results of pairwise comparisons between the extracted apps, the aggregate preference into which users' individual preferences merged can be obtained and used as the ranking result. Because the dataset we used provided neither target observations nor class labels, the ranking algorithm we proposed to learn users' preferences from the data is essentially unsupervised. The basic idea and rationale for this learning algorithm is to construct an application-specific loss function which serves as a rational decision-making agent to judge app quality assessment, instead of blindly fitting data with commonly used loss functions that stress more on the mathematical or statistical senses. More concretely, the desirable linear order of the extracted apps is the one that least violates the users' individual preferences, which can be formulated and solved as a stochastic acyclic subgraph problem on a complete digraph with each arc associated with a weight distributed according to a known distribution. Data experiment showed that the ranking results given by the pairwise-comparison-based method are different from the ranking result given by the the pairwise-comparison-based method; moreover, they can work together to provide workable ranking results that might be useful for mining knowledge from users' behavioral information.

Chapter 3

Discovering the knowledge of item rank from consumer reviews

3.1 Background

E-commerce websites have proliferated at a rapid rate over the last decade, which improves customer experience and online businesses largely. Customers purchase products based on reviews provided by the consumers of the product, because without any physical look-and-feel experience of a product, prospective consumers have to rely on the feedback from past consumers to make the final decision of whether or not to purchase the product. Consumers try to find out how other consumers have recommended the product based on its quality, usefulness and many other parameters. Since opinion of consumers plays an important role in decision making for business, information processing techniques that can discover useful knowledge from consumer reviews are desired in a variety of business applications involving consumer analytics. As the number of consumers shifting to online platforms increases, the number of consumer reviews available online also increases at a rapid rate and presents new opportunities in the way information is searched and knowledge is discovered.

With the ever increasing competition in product development, consumers have numerous options to choose from. As a consequence, it is also hard for a user to determine those products that best match their requirements in an effective and efficient way. To ease this difficulty, e-Commerce service systems and information sharing platforms employ opinion mining techniques to classify and extracting useful information such as the main features commented upon by customers from a variety of information sources of online consumer reviews. The processed information that contains the knowledge of consumer preferences is usually further summarized and presented in a ranking form. Therefore, a variety of rankings regarding consumer reviews and products can be found online. Good product rankings constructed from multi-attributive, semantic-based and consumer-centric features will often enrich a customer's knowledge and experience of products and hence help prospective consumers improve their efficiency of decision-making by reducing their cost of search. This makes the role of consumer reviews and product rankings regarding a product highly critical to businesses.

The explosion in number and variety of product rankings also brings new challenges in the way information is retrieved and knowledge is discovered. For a variety of reasons, most of ranking systems and information sharing platforms don't provide the rationale behind their ranking results in an explicit and expressive way. Without such knowledge, when comparing similar products that provide almost the same functionality customers might wonder why and how one product is ranked over another. The case may be even worse if customers unfortunately find several

conflicting ranking results at different information sources. This gives rise to the need of discovering useful knowledge of product rankings from consumer reviews. From a practical perspective, a deeper knowledge of product rankings will not only help customers find only those products that suit their individual needs best but also help e-Commerce vendors better understand the ranking rationale used by others including their competitors.

Motivated by this observation, we present a novel method for discovering the knowledge of rank information from consumer reviews. The proposed method integrates a general cognitive process of comparison and hence may be used to understand the rationale of a general ranker as well as to learn its preferences simultaneously.

3.2 Related works and new contributions

Our work is inscribed within the body of consumer reviews mining. Online consumer reviews, functioning both as informants and as recommenders, provide product information and recommendations from the customer perspective and thus are important in making purchase decisions and for product sales. Related studies can be found in [133, 5, 132, 154, 192]. This literature mainly discusses the impact of product reviews on purchasing intention and sales volumes based on numeric measures representing the valence and volume of reviews or semantic measures representing quality of reviews, demonstrating how textual data can be used to learn consumers' preferences or used for predictive modeling of future changes in sales. In this chapter, we use textual information regarding product descriptions in consumer reviews to reveal the ranker's preferences.

Our work is also related to preference learning. Roughly speaking, preference learning is about inducing predictive preference models from empirical data. In the literature on choice and decision theory, two main approaches to modeling preferences can be found, namely in terms of utility functions and in terms of preference relations. From a machine learning point of view, these two approaches give rise to two kinds of learning problems: learning preference relations and learning utility scoring functions. Related works in learning utility functions can be found in [73, 76, 77, 92, 180]. In particular, we notice the work by Dembczyński et al. [40], in which they proposed a methodology based on learning of an ensemble of decision rules. Since decision rule models are "glass box" providing a clear justification of decisions, their work emphasized on explanation.

The method we proposed in this chapter is an inverse problem of the ranking method proposed in the section 2.3 of Chapter 2 where we proposed a ranking model by introducing a cognitive process of comparison. Recall that a ranking model of a ranking process gives a mathematical description of the operating behavior of a ranker, as well as its qualitative and quantitative operating conditions, by specifying the principal quantities of the process, namely: input, system parameters and output. The ranking method for app extraction is direct, in the sense that given the input of users' behavior records and the system parameters $\mathbb{Q}$, we computed a score for each app by using a specific score function S and hence find out the output of the ranking model. Because there isn't any system parameter, target observation or class label in that dataset we used for app quality assessment, we had to construct $\mathbb{Q}$ and S manually and hence were left with no alternate but the direct approach. The constructions of $\mathbb{Q}$ and S are essentially domain-specific and context-dependent, making it less convenient in automatic knowledge discovery.

In the cases where rankings of items are observable and known, but other principal quantities of ranking process are unknown or only partially known, we may be interested to know how the items are ranked by the ranker and thus the ranking rationale behind the ranking process. When explanation is also the focus, a theory of the ranking rationale, taking account of available prior knowledge, should be developed and useful to guide the direction of explanation. The general form of reasoning that encompasses both modeling and theorizing is called discovery, and the endeavor of discovering previously unknown knowledge is related to knowledge discovery. In general, knowledge discovery is the organized automatic process of identifying valid, novel, useful, and understandable patterns from complex data sets, involving formulating hypotheses, developing model and algorithms, and discovering previously unknown patterns. In particular, we define the knowledge discovery problem, in a more specific sense, as the combination of the reconstruction problem and the identification problem, emphasizing that the system parameters are used to extract as much as information as possible, namely:

- **Knowledge discovery problem:** Given the output and some prior knowledge of the input and the system parameters, find out input and determine the system parameters simultaneously, such that the input has led to this output and the system parameters are in agreement with the prior knowledge and the relation between input and output.

Equipped with target observation, the problem of constructing $\mathbb{Q}$ and S is an identification problem of finding out unknown parameters of known consequences. That is, given the input data and the output target observation, determine the system parameters $\mathbb{Q}$ and S which are in agreement with the relation between input and output.

Motivated by this idea, this chapter discusses a method for discovering the knowledge about the rank of items by constructing a learning machine that makes prediction and explanation simultaneously. To concretize the discussion, a collection of books ranked in a linear order are used as the observed output, and consumer-generated product reviews of these books are used as the input. The observed ranking of the books is assumed to be the ranking result of a ranking process, in which consumer reviews are used as input. The ranking rationale is that the ranker compares positive and negative aspects of different book features with its expectations of those features and gives higher ranks to the books exceeding its expectations. A theory of the ranking rationale is formulated, which leads to a semantic space of hypotheses described by system parameters. With the input, system parameters and observed output, we are interested in the questions of why and how one book is ranked over another. This actionable knowledge may be helpful to give a better understanding of the behavior of the book ranker and hence the ranking of the books.

3.3 Problem description

Observing k ranked items (books) $I_1 \prec \ldots \prec I_k$ as well as their consumer reviews, we think of that the item rank is determined by a ranker, who evaluated the qualities of item attributes based on the consumer reviews. Reviews in itself are not sufficient. The entire content posted by consumers may not be relevant or useful. We need to display the content that is most relevant and useful. Filtering out key attributes out of this plethora of content is a task that requires critical analysis. Here, assume that we have discovered a set of relevant and useful attributes. To describe the problem, we first show how to process the data and introduce some notation and symbols.

Let $\{f_1, \ldots, f_m\}$ be the set of item attributes. For simplicity, each attribute f is described either positively or negatively, denoting respectively by f^+ and f^-. Let $T = (t_1^+, \ldots, t_m^+, t_1^-, \ldots, t_m^-) \in \{0,1\}^{2m}$ be a vector of a consumer review,

$$t_s^+ = \begin{cases} 1, & \text{if a positive description of } f_s \text{ is observed,} \\ 0, & \text{otherwise.} \end{cases}$$

We define t_s^- in a similar way. In addition, $t_s^+ + t_s^- \leq 1$, for $s = 1, \ldots, m$.

For each item, let $\mathcal{T}_i = \{T_{i,1}, \ldots, T_{i,N(i)}\}$ be the set of vectors of consumer reviews of item i and define $\mathbf{p}_i = \frac{\sum_{j=1}^{N(i)} T_{i,j}}{\sum_{j=1}^{N(i)} |T_{i,j}|}$ as the ranker's impression vector towards item i, where $|T|$ is the number of ones in T. It can be seen $\mathbf{p}_i \in \mathcal{P}$, where

$$\mathcal{P} = \left\{ \mathbf{p} = (p_1^+, \ldots, p_m^+, p_1^-, \ldots, p_m^-) \in [0,1]^{2m} : \sum_{j=1}^{m} (p_j^+ + p_j^-) = 1 \right\}.$$

To process the data of ranked items, associate item I_i with label y_i, such that $y_i = 1$ if and only if $i \leq r$ for some $r < k$, otherwise $y_i = -1$. Rewriting the rank into binary classification will great simplify our discussion and will not lose the generality of the method.

Given the data $(\mathbf{p}_1, y_1), \ldots, (\mathbf{p}_k, y_k)$, our goal is to simultaneously explain and predict the behavior of the ranker.

3.4 Problem formulation

To explain and predict the behavior of the ranker, this section first introduces a generative ranking model. Then, with this model we formulates the questions of interest as a learning problem.

3.4.1 A generative model of comparison

Recall that in section 2.3 of Chapter 2, a cognitive process of comparison is introduced to model the process of ranking. Rankings are made with reference to a contextually determined standard in a pointwise way. Because there isn't any observation of ranking results in that case, we had to construct $\mathbb{Q}$ manually. Therefore, the construction of $\mathbb{Q}$ involved strong assumptions, making it less convincing for a general use. Besides this, the construction of S was also specified manually, making it inconvenient for automatic knowledge discovery, although S is proved to have desirable properties. In this work, we employ the same cognitive process to model a ranker's ranking behavior, and also discuss the constructions of the standard of comparison and score functions, but from a new perspective.

To establish a dependency relationship between $\mathbf{p}$ and y, we introduce a latent and unknown vector $\mathbf{q}$ called anchor vector, serving as the role of $\mathbb{Q}$. That is, $\mathbf{q}$ is the neutral standard to be compared with $\mathbf{p}$. Unlike what was done before, we construct a class of score functions by considering what desirable properties a score function should have.

First of all, S should be in the form $S(\mathbf{p}, \mathbf{q})$ to summarize the cognitive process of comparison. Since the semantics of q plays a role of the standard of comparison, it is also comfortable to have $\mathbf{q} \in \mathcal{P}$. Otherwise, $\mathbf{p}$ and $\mathbf{q}$ may not be comparable. Second, given a fixed standard of comparison $\mathbf{q}$, if a product receives more positive

comments on every attribute than another product does, then the product with more positive comments should have a larger score value. Third, if $\mathbf{q}$ is served as the standard and $\mathbf{p}$ is ranked higher than $\mathbf{q}$, then it is intuitive to say $\mathbf{q}$ is ranked lower than $\mathbf{p}$ when $\mathbf{p}$ is served as the standard; in the numerical sense, when $\mathbf{p}$ and $\mathbf{q}$ exchange their positions in S, the score value should change its sign. Fourth, score functions with a linear structure should be investigate first, because they are easy to understand and also useful for approximating other functions.

We say $\mathbf{p} \succeq \mathbf{p}'$ if and only if a product receives more positive comments on every attribute, that is, $p_j^+ \geq p_j'^+$ and $p_j^- \leq p_j'^-$, for $j = 1, \ldots, m$. Then, a class of scoring functions $S : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ can be constructed from the properties mentioned above as follows:

B1 $\mathbf{q} \in \mathcal{P}$ (existence)

B2 $S(\mathbf{p}, \mathbf{q}) \geq S(\mathbf{p}', \mathbf{q})$ if $\mathbf{p} \succeq \mathbf{p}'$ (monotonity)

B3 $S(\mathbf{p}, \mathbf{q}) + S(\mathbf{q}, \mathbf{p}) = 0$ (asymmetry)

B4 $S(\theta\mathbf{p} + (1-\theta)\mathbf{p}', \mathbf{q}) = \theta S(\mathbf{p}, \mathbf{q}) + (1-\theta)S(\mathbf{p}', \mathbf{q}), \forall\theta \in [0, 1]$ (linearity).

It can be verified each S is a bilinear function, that is, $S(\mathbf{p}, \mathbf{q}) = \mathbf{p}^T A \mathbf{q}$, for some asymmetric matrix A,

$$A = \begin{bmatrix} A_{m\times m}^{++} & A_{m\times m}^{+-} \\ A_{m\times m}^{-+} & A_{m\times m}^{--} \end{bmatrix}$$

called interaction matrix, satisfying a system of equalities and inequalities derived from **B2** to **B4** such as $A_{m\times m}^{++} \leq A_{m\times m}^{+-}$ and $A_{m\times m}^{-+} \leq A_{m\times m}^{--}$ entry-wisely. For simplicity, let $\mathbf{p} = (\mathbf{p}^+, \mathbf{p}^-)$ and $\mathbf{q} = (\mathbf{q}^+, \mathbf{q}^-)$. In addition, $S(\mathbf{p}, \mathbf{q})$ also has the following appealing properties:

P1 $S(\mathbf{q}, \mathbf{q}) = 0$, for any $\mathbf{q} \in \mathcal{P}$. It is an immediate result from **B3**.

P2 $S\big((\mathbf{p}^+, \mathbf{0}), (\mathbf{0}, \mathbf{q}^-)\big) \geq 0$. This can be seen as follows: by **B2** and **P1**, we have

$$S\big((\mathbf{p}^+, \mathbf{0}), (\mathbf{0}, \mathbf{q}^-)\big) \geq S\big((\mathbf{0}, \mathbf{q}^-), (\mathbf{0}, \mathbf{q}^-)\big) = 0$$

since $(\mathbf{p}^+, \mathbf{0}) \succeq (\mathbf{0}, \mathbf{q}^-)$.

P3 S is continuous on $\mathcal{P} \times \mathcal{P}$. This can be seen by treating S as a bilinear function of $\mathbb{R} \times \mathbb{R}$ restricted on $\mathcal{P} \times \mathcal{P}$.

P4 S is bounded, say $|S(\mathbf{p}, \mathbf{q})| \leq M$, for some $M > 0$ and for all $\mathbf{p}, \mathbf{q} \in \mathcal{P}$. To see this, by **B2** it suffices to maximize S subject to $\mathbf{p} = (\mathbf{p}^+, \mathbf{0})$ and $\mathbf{q} = (\mathbf{0}, \mathbf{q}^-)$. Then, $S(\mathbf{p}, \mathbf{q}) = \mathbf{p}^+ A^{+-} (\mathbf{q}^-)^T$. Suppose $M = A_{i_0 j_0}^{+-} = \max_{i,j=1,\ldots,K} A_{ij}^{+-}$, we have $S(\mathbf{e}_{i_0}^+, \mathbf{e}_{j_0}^-) = M$ and $S(\mathbf{e}_{j_0}^-, \mathbf{e}_{i_0}^+) = -M$ by definition of A^{+-} and **B3**, which implies $|S(\boldsymbol{\lambda}, \boldsymbol{\mu})| \leq M$.

This construction of S brings a potential to incorporate more *a priori* information into a ranking model, for example, by specifying the range and sign of each entry of the interaction matrix. Hence, it presents a general method to construct a class of flexibly context-dependent hypotheses for a variety of learning purposes. Let $\mathcal{A}$ be a convex set of all possible interaction matrices constructed from available *a priori* information. In this ranking model, it can be seen that the ranker's behavior is described by A and $\mathbf{q}$. Therefore, given $(\mathbf{p}_1, y_1), \ldots, (\mathbf{p}_k, y_k)$, we say our goal is to

estimate the values of the interaction matrix A and the anchor vector $\mathbf{q}$. However, the immediate challenges are that (i) under what inductive principe to infer A and $\mathbf{q}$, (ii) how to obtain robust estimation even when k is small, and (iii) how to compute A and $\mathbf{q}$.

3.4.2 Problem reformulation with learning theory

We provide a possible answer to challenges (i) and (ii) with learning theory. Assume that $\mathbf{p}_1, \ldots, \mathbf{p}_k$ are random independent observations generated according to a fixed but unknown distribution $\mathcal{F}(\mathbf{p})$ and that $y_1, \ldots, y_k$ are random independent observations generated according to a fixed but unknown conditional distribution $\mathcal{F}(y|\mathbf{p})$. Therefore, it follows that $(\mathbf{p}_1, y_1), \ldots, (\mathbf{p}_k, y_k)$ can be seen as a sample drawn randomly and independently according to a fixed but unknown joint distribution $\mathcal{F}(\mathbf{p}, y)$, where $\mathcal{F}(\mathbf{p}, y) = \mathcal{F}(\mathbf{p})\mathcal{F}(y|\mathbf{p})$.

Let $\gamma(\mathbf{p}, \alpha)$, $\alpha \in \Lambda$ be a set of functionals, where Λ is determined by available *a priori* information. Given $(\mathbf{p}_1, y_1), \ldots, (\mathbf{p}_k, y_k)$, the goal is to choose from $\gamma(\mathbf{p}, \alpha)$, $\alpha \in \Lambda$, the one which best predicts $\mathcal{F}(y|\mathbf{p})$.

More concretely, consider the loss function

$$L(y, \gamma(\mathbf{p}, \alpha)) = \begin{cases} 0 & \text{if } y = \gamma(\mathbf{p}, \alpha) \\ 1 & \text{if } y \neq \gamma(\mathbf{p}, \alpha) \end{cases}$$

and the expected value of the loss, given by the risk function

$$R(\alpha) = \int L(y, \gamma(\mathbf{p}, \alpha)) d\mathcal{F}(\mathbf{p}, y).$$

Then, the goal is to find the function $\gamma(\mathbf{p}, \alpha^*)$ that minimizes the risk function $R(\alpha)$, $\alpha \in \Lambda$. To minimize risk function $R(\alpha)$ for unknown $\mathcal{F}(\mathbf{p}, y)$, we may consider to employ the Empirical Risk Minimization(ERM) Principle or the Structural Risk Minimization(SRM) Principle [177]. The empirical risk functional is defined as $R_{emp}(\alpha) \triangleq \frac{1}{k}\sum_{i=1}^{k} L(y_i, \gamma(\mathbf{p}_i, \alpha))$. According to ERM principle, one chooses $\gamma(\mathbf{p}, \alpha_k^*)$ by minimizing $R_{emp}(\alpha)$, $\alpha \in \Lambda$. However, this principle suggests that the only strategy we should adopt is to minimize the empirical risk, which may be unfavorable when k is small. Instead, given the size of empirical data is fixed, SRM principle seems more appropriate since it finds the function that achieves the minimum of the guaranteed risk by optimizing the relationship between the quality of approximation of empirical data and the complexity of the set of functionals $\gamma(\mathbf{p}, \alpha)$, $\alpha \in \Lambda$. We apply SRM principle to estimate A and $\mathbf{q}$ and adopt the support vector estimation strategy of SRM principle that keeps the value of the empirical risk fixed and then minimizes the complexity of the set of the functionals. We consider the case where $\mathbf{p}_1, \ldots, \mathbf{p}_k$ can be separated by a hyperplane $\mathbf{p}^T\mathbf{w} - b = 0$, for some $\mathbf{w} \in \mathbb{R}^{2m}$ and $b \in \mathbb{R}$. In particular, we use a special type of hyperplane, the so-called maximal margin hyperplane [177]. A set of vectors is separated by a maximal margin hyperplane if it is separated without error and the distance between the closest vector to the hyperplane is maximal. The maximal margin hyperplane is given by

$$\begin{aligned} \min \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_i(\mathbf{p}_i^T\mathbf{w} - b) \geq 1, \ i = 1, \ldots, k. \end{aligned}$$

In this case $\gamma(\mathbf{p}, \alpha) = \mathbf{p}^T\mathbf{w} - b$, where $\mathbf{ff} = (\mathbf{w}, b)$. Noting that $S(\mathbf{p}, \mathbf{q}) = \mathbf{p}^T A\mathbf{q}$ is

linear in $\mathbf{p}$, we replace $A\mathbf{q}$ with $\mathbf{w}$ and let $b = 0$. This leads to the support vector estimation of A and $\mathbf{q}$ and formulates the questions of interest as a biconvex optimization problem (for the definition of biconvex problem, see [68])

$$\begin{aligned} \min_{A,\mathbf{q}} \quad & \frac{1}{2}||A\mathbf{q}||^2 \\ \text{s.t.} \quad & y_i(\mathbf{p}_i^T A\mathbf{q}) \geq 1,\ i = 1,\ldots,k \\ & \mathbf{q} \in \mathcal{P}, \\ & A \in \mathcal{A}. \end{aligned}$$

To simplify computations, one can introduce a generalized concept of maximal margin hyperplane which is determined by the following problem Q,

$$\begin{aligned} Q: \quad \min \quad & \frac{1}{2}||A\mathbf{q}||^2 + C\sum_{i=1}^{k}\eta_i \\ \text{s.t.} \quad & y_i(\mathbf{p}_i^T A\mathbf{q}) \geq 1 - \eta_i,\ i = 1,\ldots,k \\ & \mathbf{q} \in \mathcal{P}, \\ & A \in \mathcal{A}, \\ & \eta_i \geq 0,\ i = 1,\ldots,k. \end{aligned}$$

Here C is a fixed positive value.

3.5 Two-stage learning algorithm

Although the questions of interest have been formulated into a single optimization problem, how to obtain robust estimation of A and $\mathbf{q}$ is a challenging task. It is because (1) biconvex minimization problems are global optimization problems in general, thus it is hard even to find one global optimal solution; (2) however, it is expected to find several distinguished optimal solutions if possible, since there might be several competing explanations that predict the behavior of the ranker.

To ease the difficulty of estimation of A and $\mathbf{q}$, a two-stage learning algorithm is proposed. In the information extraction stage, we extract as many high-quality suboptimal solutions as possible by making use of the biconvex structure of Q. Then, in the knowledge formation stage, the elite suboptimal solutions are further processed and summarized into knowledge.

3.5.1 Information extraction stage

Stage 1.1 Given a fixed integer N, draw a random sample of $\mathbf{q}^{(1)},\ldots,\mathbf{q}^{(N)}$ independently and uniformly from $\mathcal{P}$.

Stage 1.2 For each $\mathbf{q}^{(j)}$, solve $Q(\mathbf{q}^{(j)})$

$$\begin{aligned} Q(\mathbf{q}^{(j)}): \quad \min \quad & \frac{1}{2}||A\mathbf{q}^{(j)}||^2 + C\sum_{i=1}^{k}\eta_i \\ \text{s.t.} \quad & y_i(\mathbf{p}_i^T A\mathbf{q}^{(j)}) \geq 1 - \eta_i,\ i = 1,\ldots,k \\ & A \in \mathcal{A}, \\ & \eta_i \geq 0,\ i = 1,\ldots,k. \end{aligned}$$

to obtain the optimal value $v^{(j)}$ and the unique solution $A^{(j)}$.

Stage 1.3 Sort the $v^{(j)}$'s, say $v^{(j_1)} \leq \cdots \leq v^{(j_N)}$. Then find $\mathbf{q}^{(j_1)}, \ldots, \mathbf{q}^{(j_{\overline{N}})}$ and $A^{(j_1)}, \ldots, A^{(j_{\overline{N}})}$, for some fixed integer $\overline{N} < N$.

3.5.2 Knowledge formation stage

Stage 2.1 Group the elite samples of $\mathbf{q}$'s and A's obtained in stage 1.3 into K_q and K_A clusters $\mathcal{C}^q = \{C_1^q, \ldots, C_{K_q}^q\}$ and $\mathcal{C}^A = \{C_1^A, \ldots, C_{K_A}^A\}$, respectively, by solving

$$\min_{\mathcal{C}^q = \{C_1^q, \ldots, C_{K_q}^q\}} \sum_{h=1}^{K_q} \sum_{\mathbf{q} \in C_h^q} ||\mathbf{q} - \mathbf{q}_h||^2,$$

$$\min_{\mathcal{C}^A = \{C_1^A, \ldots, C_{K_A}^A\}} \sum_{l=1}^{K_A} \sum_{\mathbf{A} \in C_l^A} ||\mathbf{A} - \mathbf{A}_l||^2,$$

where K_q and K_A are fixed and relatively small integers. This is exact the standard formulation of K-Means clustering [95, 107, 144].

Stage 2.2 Denote by $\mathbf{q}_h^*$ and A_l^* the clustering centers obtained in step 2.1. Moreover, let $\mathbf{w}_F$ be the optimal solution of the following problem

$$\begin{aligned} \min \quad & \frac{1}{2}||w||^2 + C \sum_{i=1}^{k} \eta_i \\ \text{s.t.} \quad & y_i(\mathbf{p}_i^T w) \geq 1 - \eta_i, \ i = 1, \ldots, k \\ & \eta_i \geq 0, \ i = 1, \ldots, k. \end{aligned}$$

For each center $\mathbf{q}_h^*$, let

$$A_{\mathbf{q}_h^*} = \arg \min_{A \in \{A_1^*, \ldots, A_l^*\}} ||A\mathbf{q}_h^* - \mathbf{w}_F||.$$

The obtained K_q pairs of $\mathbf{q}_h^*$ and $A_{\mathbf{q}_h^*}$ are defined as the desired knowledge of the rank of the items.

The two-stage learning algorithm inherently incorporates both Bayesian approach and Frequentist approach. The rationale of the algorithm is framed from a Bayesian perspective and can be interpreted as follows: Given prior information $\mathcal{I}$ about S, $\mathcal{P}$ and $\mathcal{A}$ as well as consumer reviews $\mathcal{T}$ and target observation data $\mathcal{R}$, in order to estimate the pair of A and $\mathbf{q}$, we calculate the posterior probability of system parameters A and $\mathbf{q}$ that best suit the relation between consumer reviews data $\mathcal{T}$ and the target observation data $\mathcal{R}$. Let $\mathcal{D} = (\mathcal{T}, \mathcal{R})$ be the total data available. The idea mentioned above can be formulated as

$$\begin{aligned} & Prob(A, \mathbf{q} | \mathcal{D}, \mathcal{I}) \\ \propto & Prob(A, \mathbf{q}, \mathcal{I}) Prob(\mathcal{D} | A, \mathbf{q}, \mathcal{I}) \\ & Prob(\mathbf{q}, \mathcal{I}) Prob(A | \mathbf{q}, \mathcal{I}) Prob(\mathcal{D} | A, \mathbf{q}, \mathcal{I}) \\ \propto & Prob(\mathbf{q} | \mathcal{I}) Prob(\mathbf{A} | \mathcal{I}) Prob(\mathcal{D} | A, \mathbf{q}, \mathcal{I}). \end{aligned}$$

Then, if A and $\mathbf{q}$ are respectively assumed to distribute on $\mathcal{A}$ and $\mathcal{P}$ uniformly, then $Prob(\mathbf{q}|\mathcal{I})$ and $Prob(\mathbf{A}|\mathcal{I})$ are both constants and hence it follows

$$Prob(A, \mathbf{q}|\mathcal{D}, \mathcal{I}) \propto Prob(\mathcal{D}|A, \mathbf{q}, \mathcal{I}).$$

In this formulation, however, $Prob(\mathcal{D}|A, \mathbf{q}, \mathcal{I})$ is hard to derive explicitly, since $\mathcal{I}$ involves the information of structure of S and hence seems very complicated. This motivates us to employ Frequentist learning approach to ease this difficulty. The basic idea is to approximate the likelihood in a functional space with learning techniques, instead of giving it an explicit expression. The biconvex minimization problem we proposed can capture this idea and integrate the axiomatic, Bayesian and Frequentist approaches together within a single model. The biconvex minimization problem also serves a sampler drawing samples according to an approximation of the posterior probability $Prob(A, \mathbf{q}|\mathcal{D}, \mathcal{I})$. That is the reason why after drawing a random sample of $\mathbf{q}$ uniformly from $\mathcal{P}$, we continue to solve a series of convex optimization problems specified by $\mathbf{q}$ and then only keep the pairs of A and $\mathbf{q}$ with a smaller optimal value as the elite samples. In fact, since $Prob(\mathbf{q}|\mathcal{I})$ is a constant, the estimation of the posterior probability of A and $\mathbf{q}$ is reformulated by

$$Prob(A, \mathbf{q}|\mathcal{D}, \mathcal{I}) \propto Prob(\mathbf{q}|\mathcal{I}) Prob(\mathcal{D}|A, \mathbf{q}, \mathcal{I}).$$

From a computation perspective, the biconvex minimization problem accelerates the estimation of posterior probability. This is because, instead of drawing random samples of A and $\mathbf{q}$ separately, we only draw a random sample of $\mathbf{q}$, and then find out the best approximation A with respective to $\mathbf{q}$ by solving a convex optimization problem and hence make a random sample of A needless. Because there are mathematical structures underlying A and $\mathbf{q}$, instead of averaging the samples directly, clustering method was employed to summarized the posterior probability based on the elite samples.

3.6 Simulation

The main purpose of this section is to inspect the effectiveness of the two-stage learning algorithm. Particularly, we would like to see how the *a priori* information helps to predict the behavior of the ranker and whether the knowledge discovered by the two-stage learning algorithm is useful to explain the behavior of the ranker.

3.6.1 Simulation procedure

Step 1 Generate underlying true interaction matrix $\mathbf{A}_T$ and anchor vector $\mathbf{q}_T$.

Step 2 For pre-fixed k and Δ, generate k random samples $\mathbf{p}$ independently and uniformly drawn from $\mathcal{P}$ with balanced numbers of labels given by

$$y_i = \begin{cases} 1, & \text{if } \mathbf{p}_i^T \mathbf{A}_T \mathbf{q}_T \geq \Delta, \\ -1, & \text{if } \mathbf{p}_i^T \mathbf{A}_T \mathbf{q}_T \leq -\Delta, \end{cases}$$

otherwise, regenerate $\mathbf{p}$. These $\mathbf{p}$'s are used as the simulation data. It can be seen that the simulation data are linearly separable.

Step 3 Determine $\mathcal{A}$. First, for each $A \in \mathcal{A}$, it should have the properties derived from **B1** $-$ **B4**. Then, we give a box constraint for each non-diagonal entry of

A by letting $(A_T)_{ij} - B \leq A_{ij} \leq (A_T)_{ij} + B$, for $i \neq j$. Here, B is a fixed positive number serving as a measure the amount of *a priori* information.

Step 4 Apply the two-stage learning algorithm to the randomly generated simulation data under various parameter settings.

Step 5 Compare (1) $\{\mathbf{q}_h^*\}_{h=1}^{K_q}$ with $\mathbf{q}_T$, (2) $\{A_{\mathbf{q}_h^*}\}_{h=1}^{K_q}$ with A_T and (3) $\{A_{\mathbf{q}_h^*}\mathbf{q}_h^*\}_{h=1}^{K_q}$ with $A_T\mathbf{q}_T$ and $\mathbf{w}_F$.

3.6.2 Parameter setting

In the simulation, the 2-stage learning algorithm was tested under the parameter settings: $m = 4, k \in \{10, 100, 950\}, \Delta \in \{1, 1.5\}, B \in \{0.1, 1, 5, 10\}, N \in \{100, 1000, 3000, 10000\}$, $\overline{N} = N/5$ and $K_q = K_A = 4$.

3.6.3 Simulation result

Summary of simulation result

Result 1 Even when $\Delta = 1$, $k = 10$ and $B = 10$, the learning algorithm still tends to give a good estimation of $\mathbf{q}$ and A. This result is appealing since it suggests a validation of the effectiveness of the algorithm. When $\Delta = 1.5$ (the maximal margin gets larger) while keeping the other parameters unchanged, the quality of the estimation of $\mathbf{q}$ and A seems unaffected. When $B = 0.1$, the quality of the estimation of $\mathbf{q}$ has a noticeable but limited improvement. When $k = 950$, the quality of the estimation of $\mathbf{q}$ has a slight improvement. $N = 3000$ seems a good trade-off between estimation quality and computation time.

Result 2 In the simulation under various parameter settings, the learning algorithm tends to find $\mathbf{q}_{h_0}^*$ and $A_{q_{h_0}^*}$ such that

$$||A_T\mathbf{q}_T - A_{q_{h_0}^*}\mathbf{q}_{h_0}^*|| < ||A_T\mathbf{q}_T - \mathbf{w}_F||.$$

This appealing result seems to provide a numerical evidence that the knowledge of underlying behavioral structure of the ranker would help to improve the accuracy of the prediction.

Instances of simulation result

From Fig. 3.1 to 3.20, we show ten instances of computer simulation. The parameters in these instances are fixed as follows: $m = 4$, $\Delta = 1$, $k = 10$, $B = 10$, $C = 1000$, $N = 3000$ and $K_q = K_A = 4$. In each instance, q_T is randomly generated and A_T is randomly modified as needed. In the figures, the values of q_T and A_T (A_T has 28 free variables since A_T is asymmetric) are connected by the black lines, and the dots represent the support vector estimations of the values of q_T and A_T given by ten data. The following ten simulation results show the learning ability of the two-stage learning algorithm in the linear separable case where only small data are available. In each simulation, $\{\mathbf{q}_h^*\}_{h=1}^4$ and $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ are compared with $\mathbf{q}_T$ and A_T, respectively.

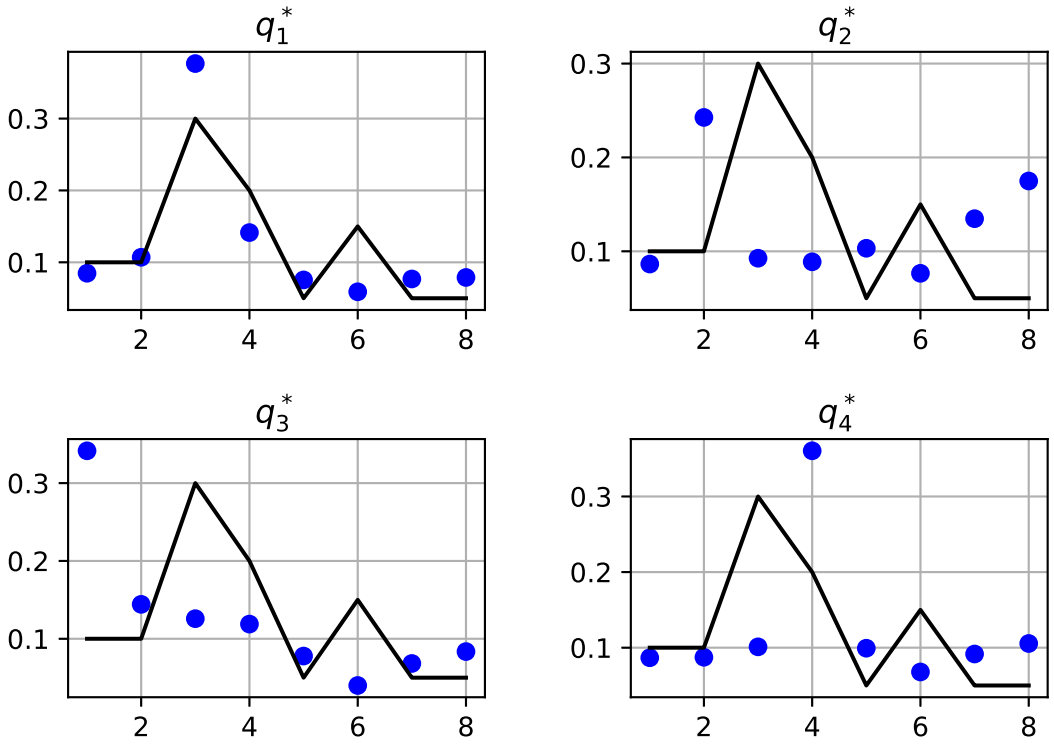


FIGURE 3.1: Simulation 1 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and $\mathbf{q}_T$

3.7 Experiment

We used the data collected from the website[1] which consisted of a list of ranked python books and their consumer reviews. These consumer reviews evaluated the python programming books mainly from four attributes: readability, beginner-friendliness, content highlight and content coverage. After removing the books less focused on the topic of python programming, the data set consisted of ten data as shown in Table.1. In the column of attribute description are the unnormalized impression vectors. The books 4 and 8 are removed from the data set since they had few consumer reviews. In addition, the book 10 is removed since it is an outlier.

However, the data obtained are not linearly separable. Moreover, there are zeros in some unnormalized impression vectors. To deal with the nonlinearity, we tried to reduce the value of C from 1000 to 10. To deal with the zeros, we consider the impression vectors given by

$$\mathbf{p}_i = \frac{S\mathbf{1} + \sum_{j=1}^{N(i)} T_{i,j}}{2mS + \sum_{j=1}^{N(i)} |T_{i,j}|},$$

where S is a positive number and $\mathbf{1}$ is the all-ones vector. In this experiment, we set $S = 15$, $C = 10$, $m = 4$, $N = 3000$, $B = 10$, $K_q = K_A = 4$ and $A_T = \begin{bmatrix} 0 & \mathbf{5}J \\ -\mathbf{5}J & 0 \end{bmatrix}$. Here, A_T represents the *a priori* information of A and $\mathbf{J}$ is the all-ones matrix.

Applying the two-stage learning algorithm, the experiment result of

$$\mathbf{q}^* \doteq (0.35, 0.1, 0.1, 0.1, 0.13, 0.07, 0.06, 0.09)$$

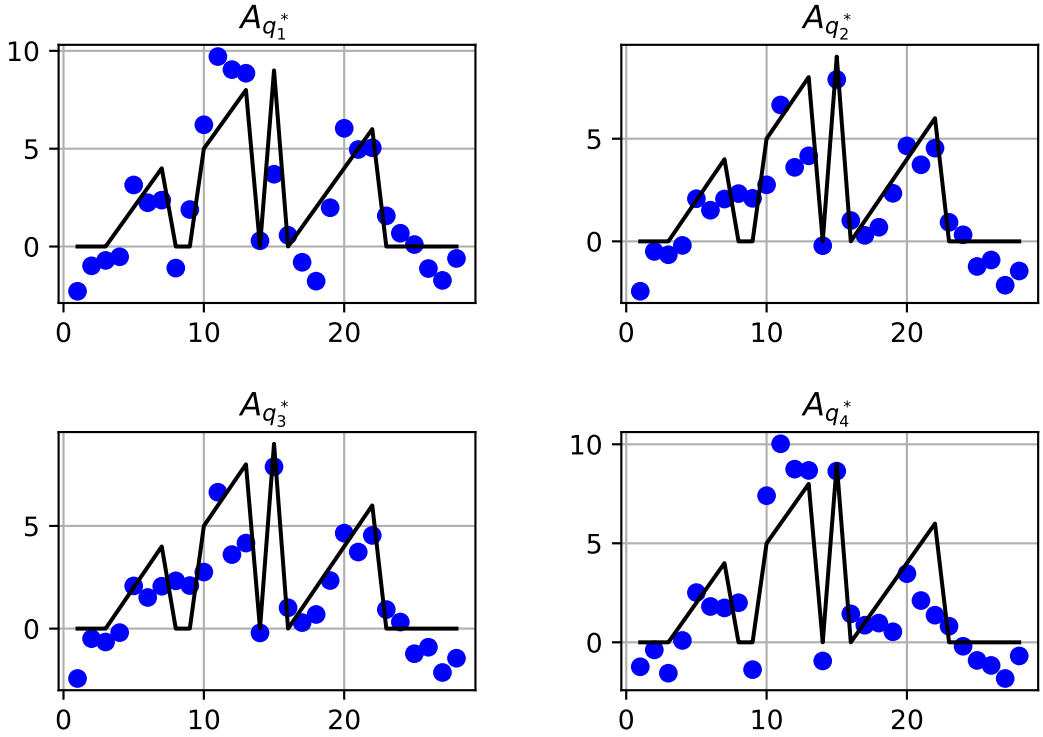


FIGURE 3.2: Simulation 1 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and A_T

showed that the ranker seems to have much higher expectation for positive readability and also have a slightly higher tolerance for negative readability. This implies that the positive evaluation of readability may be an important factor to produce a high rank but a low rank may not be the result of the negative evaluation of readability. Moreover, the first row of interaction matrix $(0, 1.1, 2.0, 2.6, 1.0, 1.2, 2.0, 2.6)$ implies that positive readability is relatively more important. However, since the data are not linearly separable, we caution that the behavioral model may suffer from being falsified.

3.8 Conclusion

We proposed a method of how to discover the knowledge of item rank by learning the behavior of the ranker. This method also showed a possibility that the questions concerning explanation and prediction may be simultaneously answered under some conditions.

More specifically, the basic idea behind the method is to construct a choice model and then to build it in a computational learning framework to learn its parameters. This idea formulates the questions of interest as a single biconvex minimization problem which has a relationship with SVM(Support Vector Machines). To facilitate the process of knowledge discovery, we develop a two-stage learning algorithm. In the first stage, we extract as much information as possible from the observed data, while in the second stage, the extracted information is further summarized into knowledge.

The main ideas behind the algorithm design are that (1) the cognitive process of comparison proposed in section 2.3 of Chapter 2 is reformulated from an axiomatic perspective and produces a generative model of comparison; (2) the identification

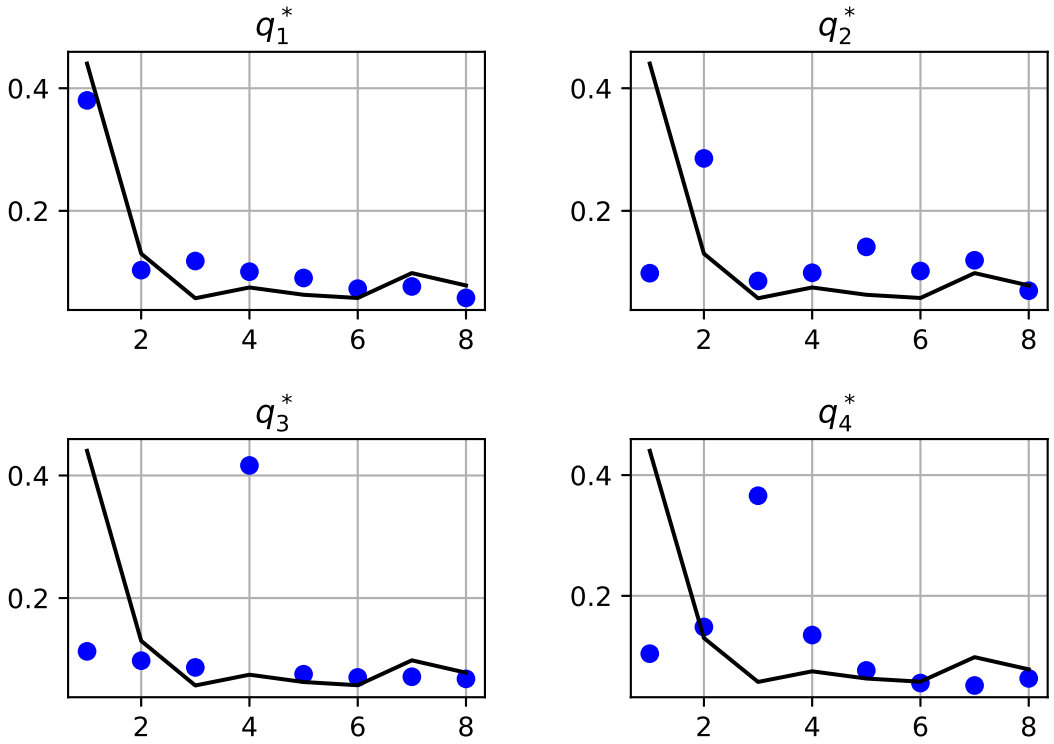


FIGURE 3.3: Simulation 2 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

problem is then formulated from a Bayesian perspective, aiming to incorporate the cognitive process of comparison into prior knowledge; (3) since it is hard to derive an explicit likelihood function, Frequentist learning perspective is employed to ease the difficulty. After the parameters of the ranking model are learned, the ranking rationale can be interpreted by using these parameters and hence the knowledge of the rankings can be discovered; moreover, the preferences of the ranker can be obtained from generalization beyond the observed data given. The simulation result gave a positive support to this possibility. An experiment is designed to test the proposed method using consumer reviews collected from websites online for book reviews. Although, as the data experiment showed, this method needs further extensions to deal with more general nonlinearly separable case, it still seems promising.

The proposed method is not limited to e-Commerce applications, but also works for more general topics as well. For example, for a consumer analysis concerning behavioral data, there will positive and negative behavioral patterns suggesting a consumer's positive and negative attitudes, and hence this method may discovered useful knowledge from the behavioral data as well as a collection of target observations expressed in the form of rankings. It can also be generalized in other applications where positive and negative attributes are involved in a decision-making process. With the knowledge of the rankings of interest discovered, it may assist potential customers to make a better decision based on the attributes being commented up by past consumers. It can also help companies understand which attribute of the product is well received by the audience and which is not.

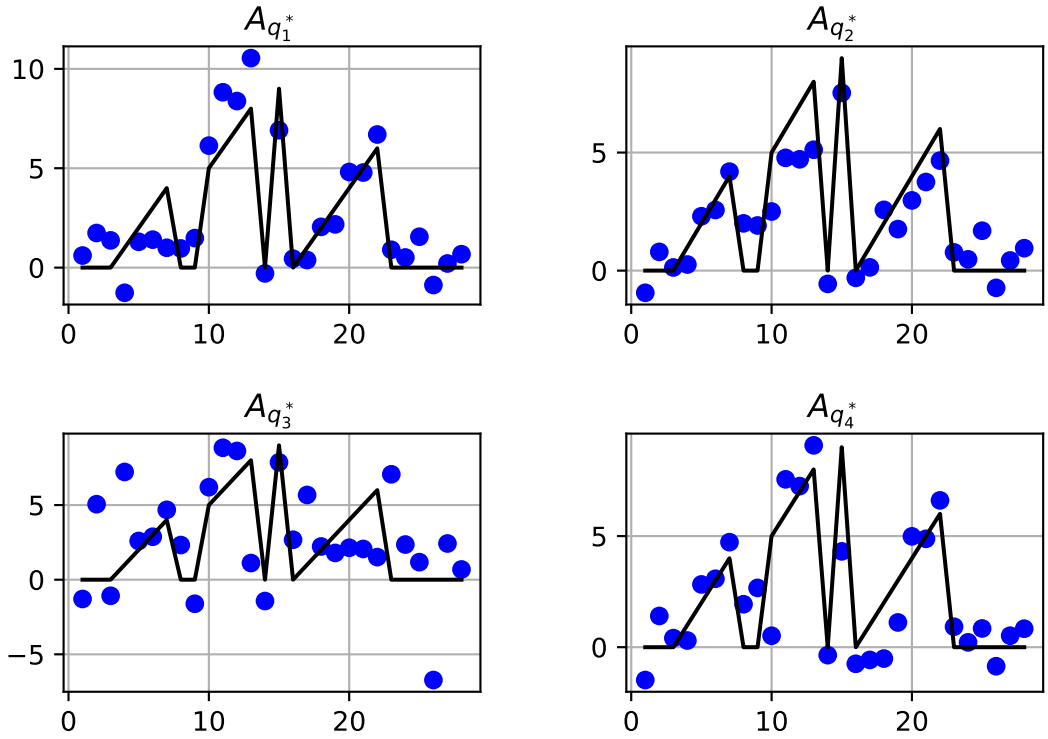


FIGURE 3.4: Simulation 2 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

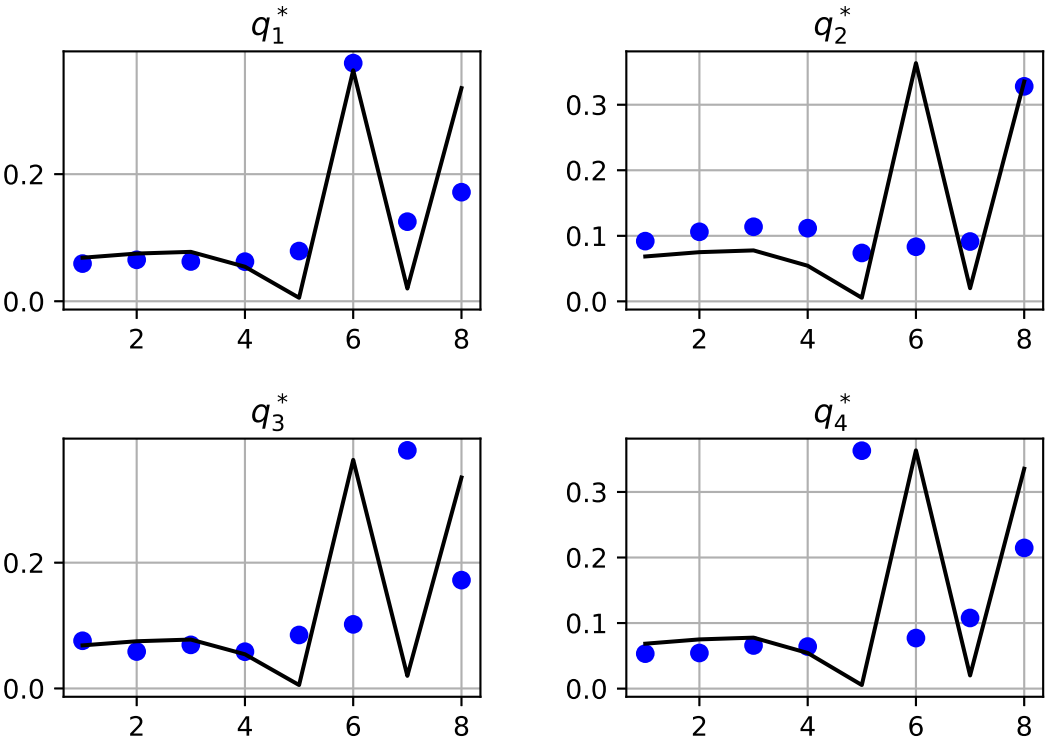


FIGURE 3.5: Simulation 3 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

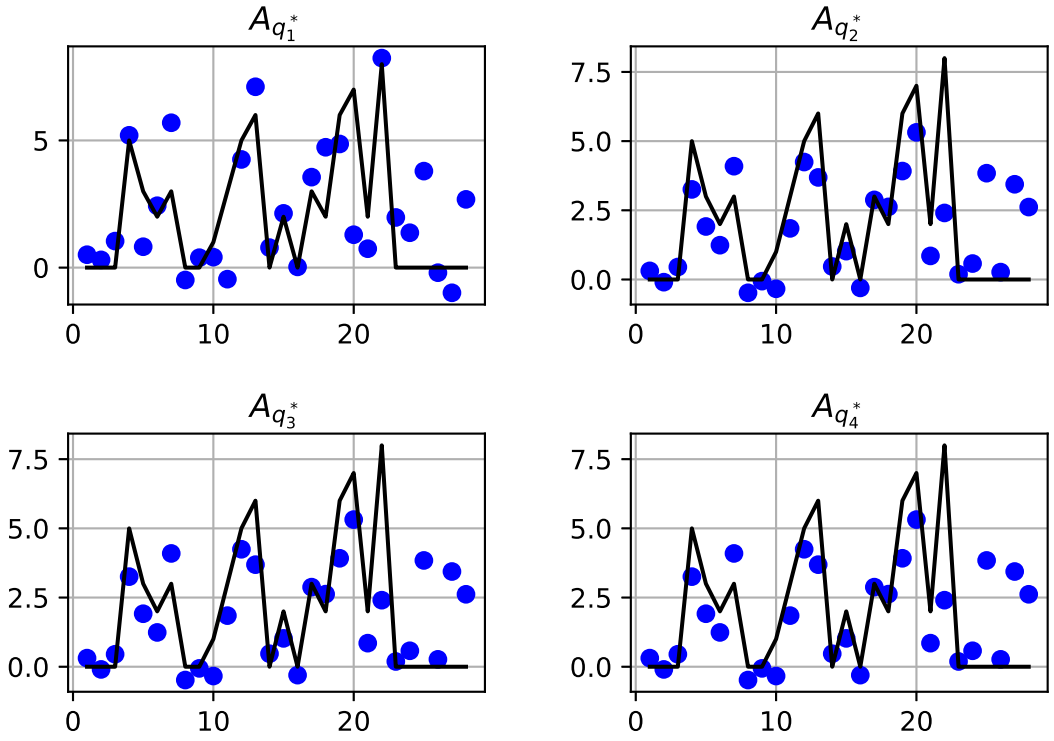


FIGURE 3.6: Simulation 3 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

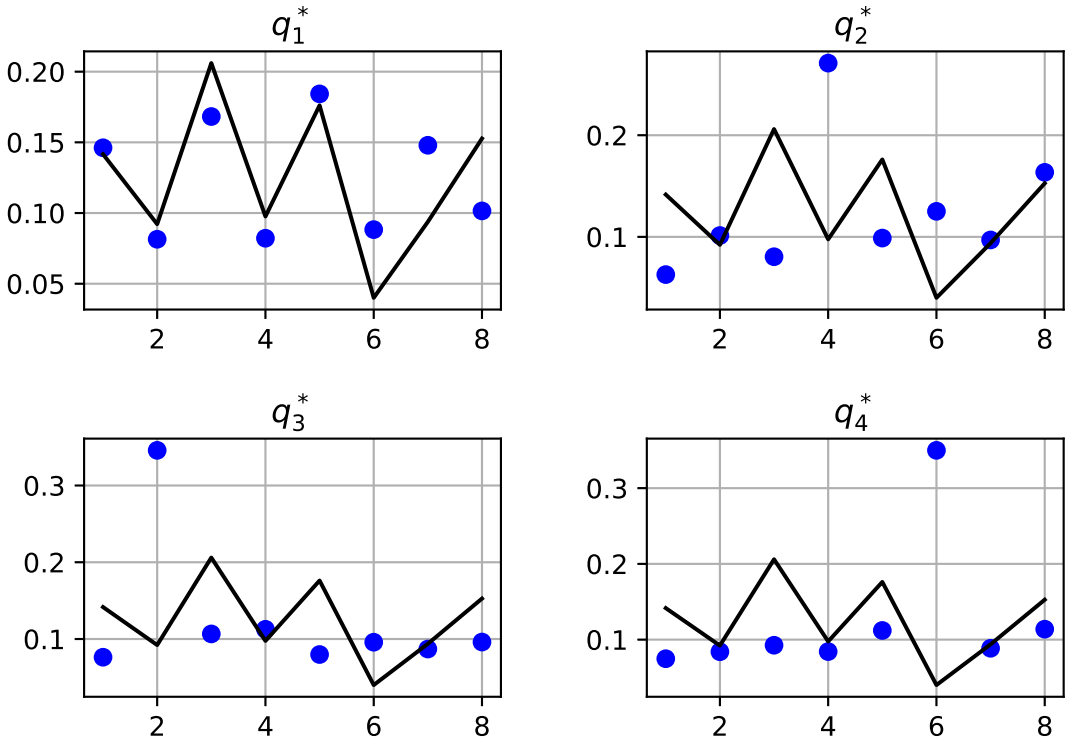


FIGURE 3.7: Simulation 4 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

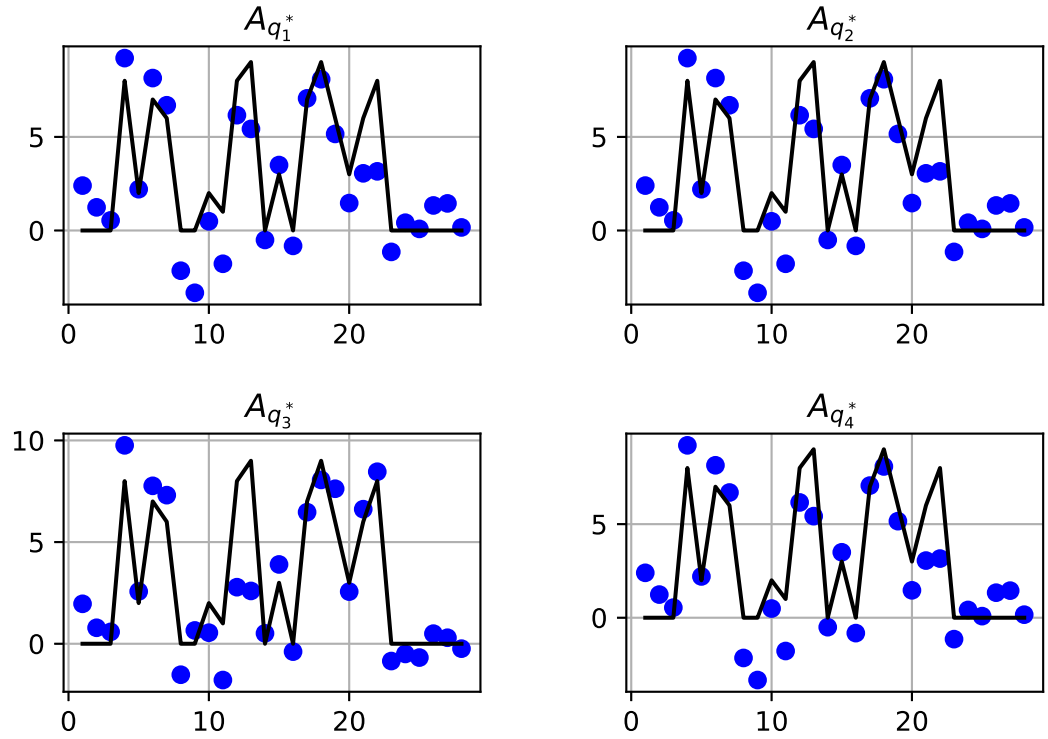


FIGURE 3.8: Simulation 4 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

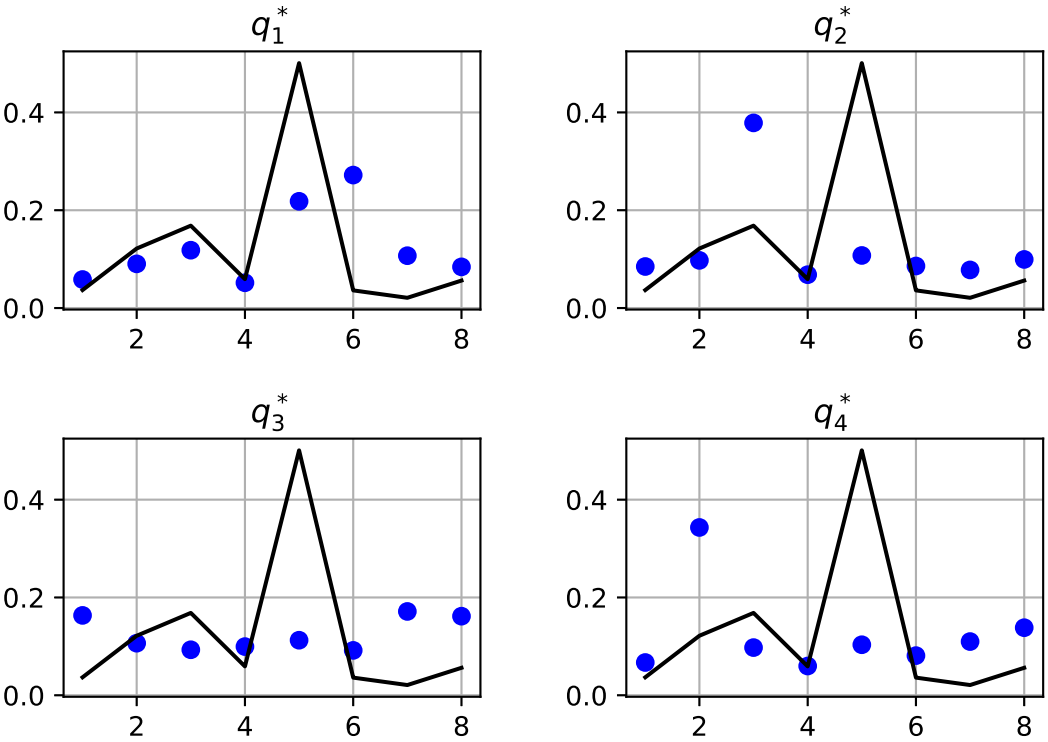


FIGURE 3.9: Simulation 5 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

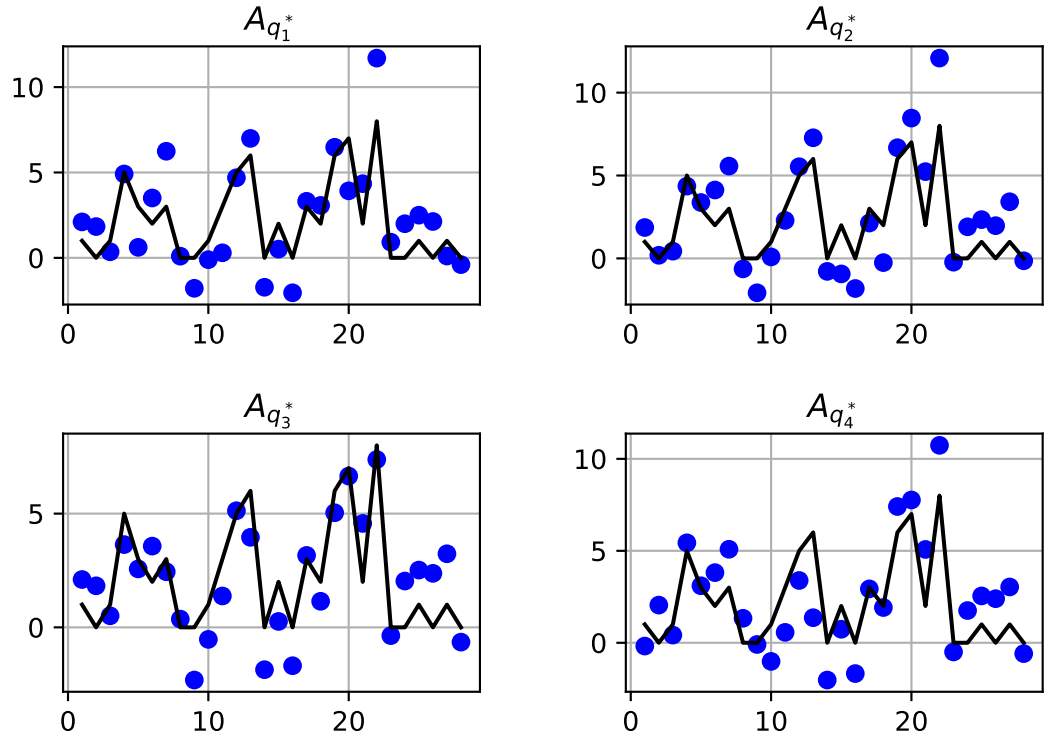


FIGURE 3.10: Simulation 5 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

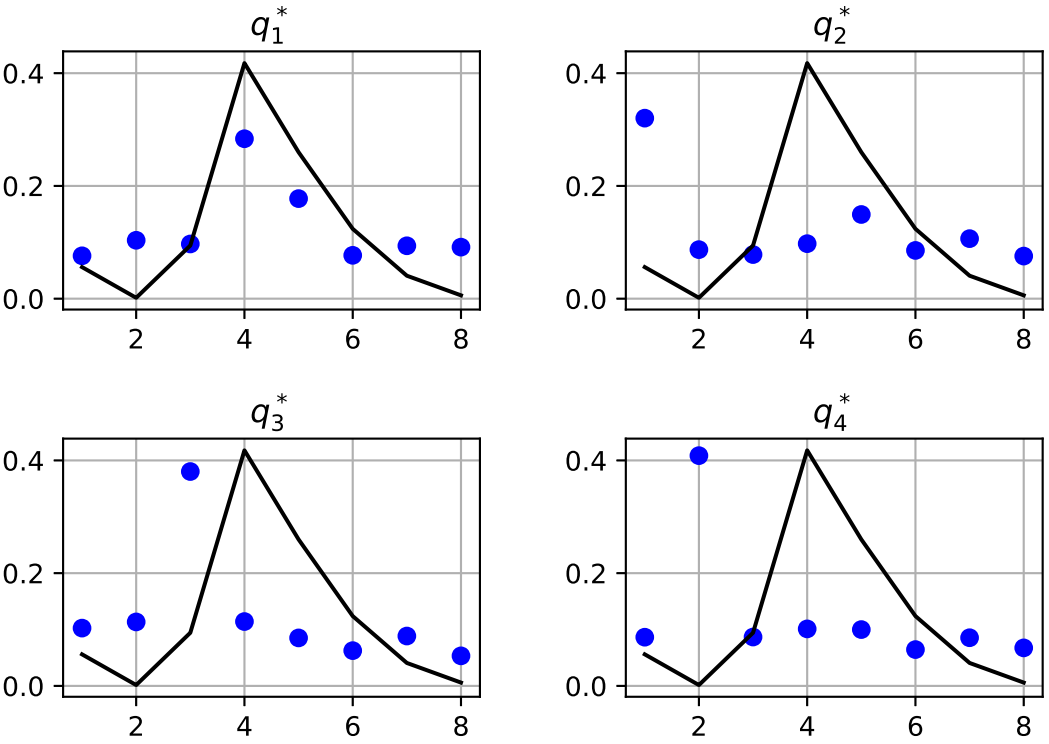


FIGURE 3.11: Simulation 6 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

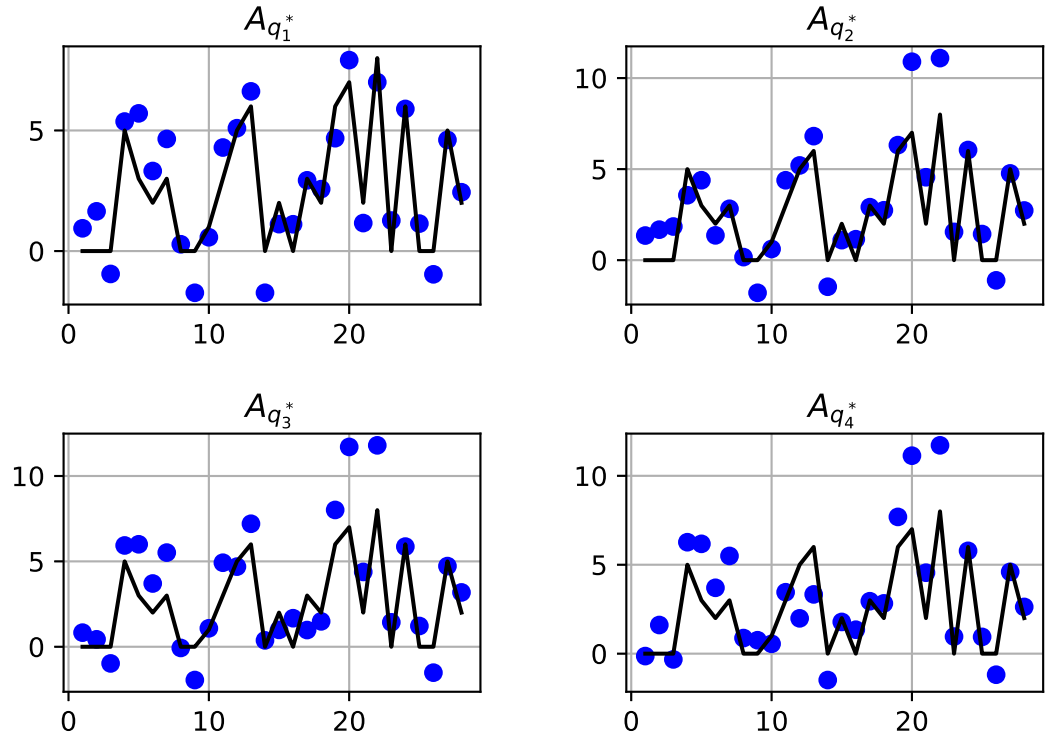


FIGURE 3.12: Simulation 6 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

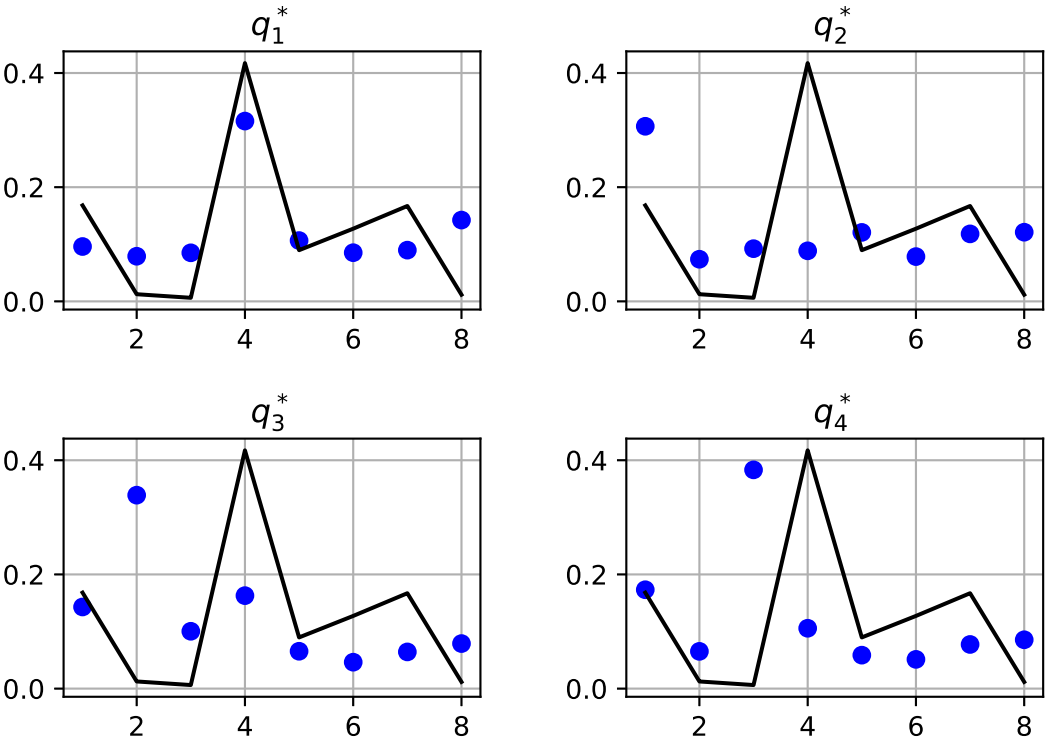


FIGURE 3.13: Simulation 7 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

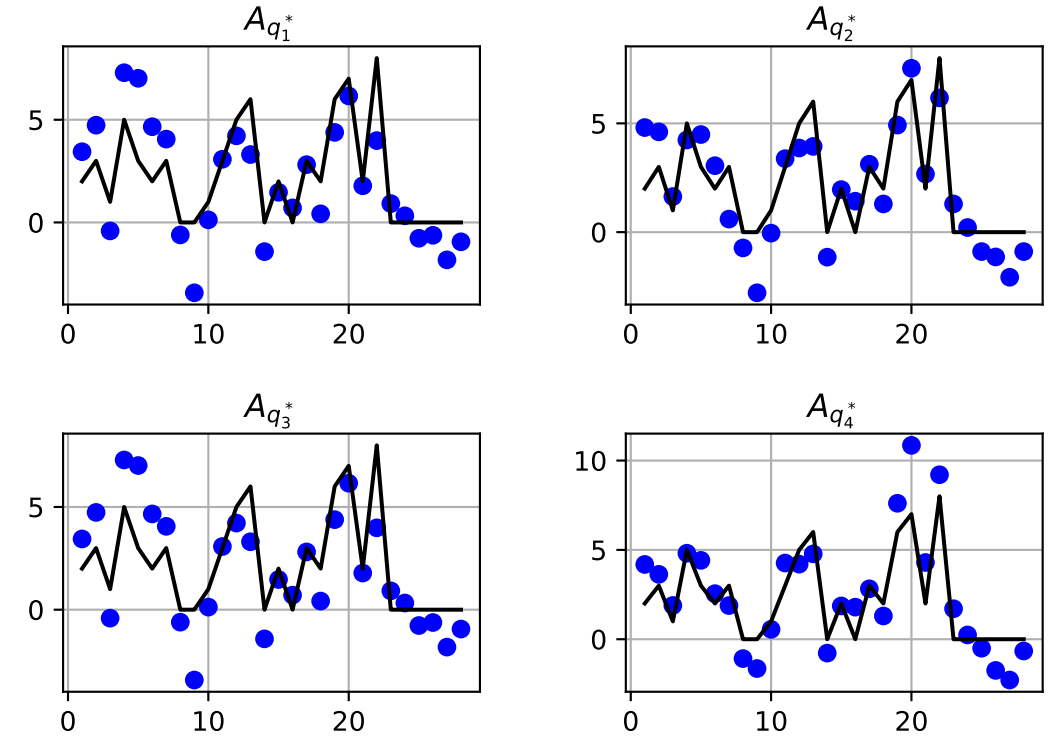


FIGURE 3.14: Simulation 7 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

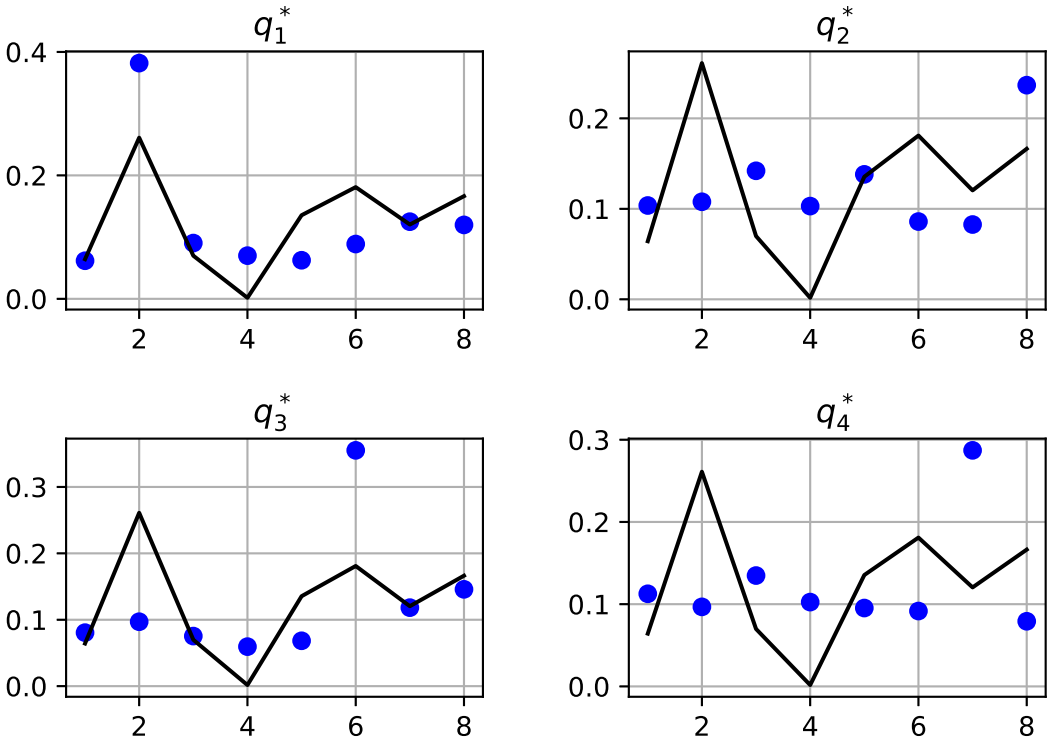


FIGURE 3.15: Simulation 8 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

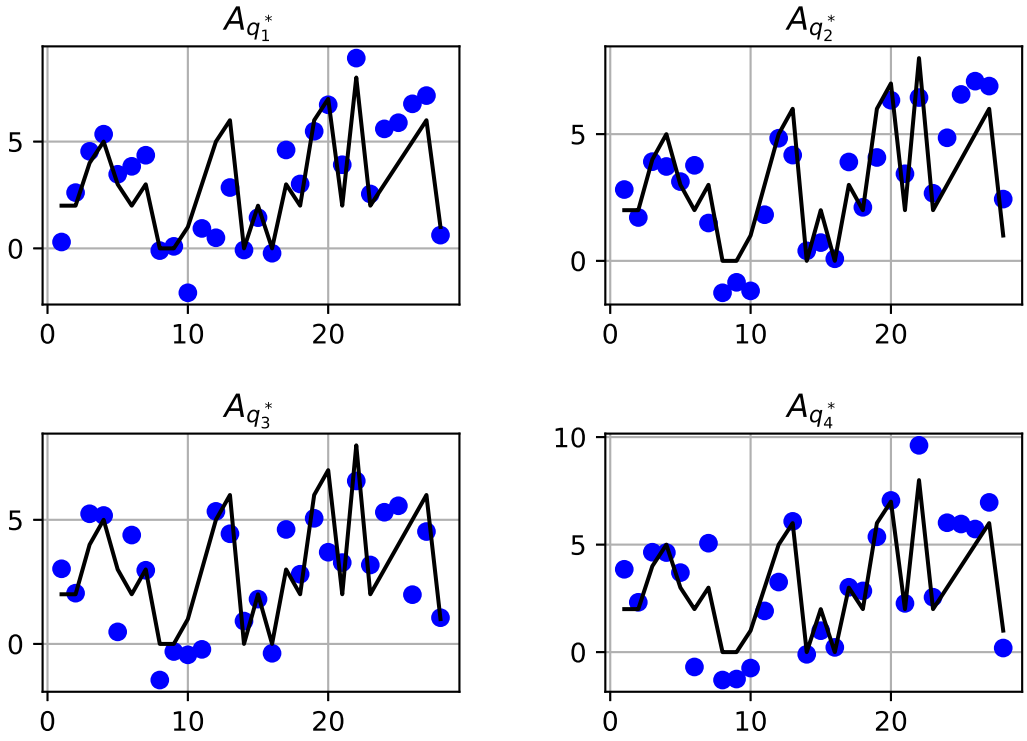


FIGURE 3.16: Simulation 8 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

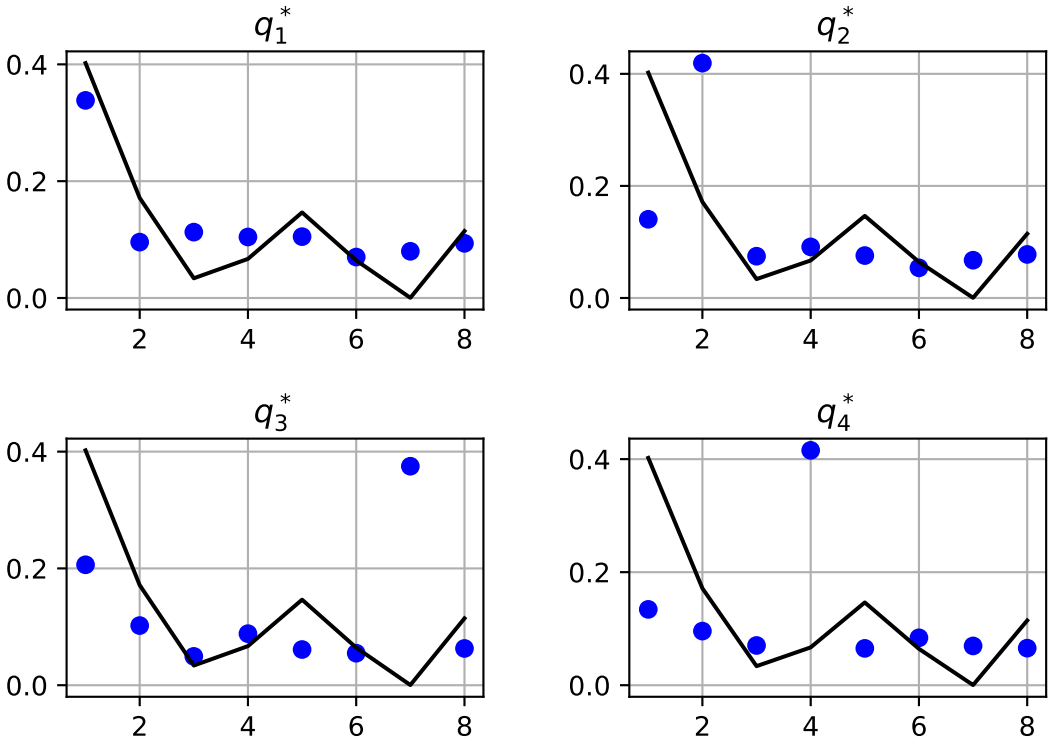


FIGURE 3.17: Simulation 9 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

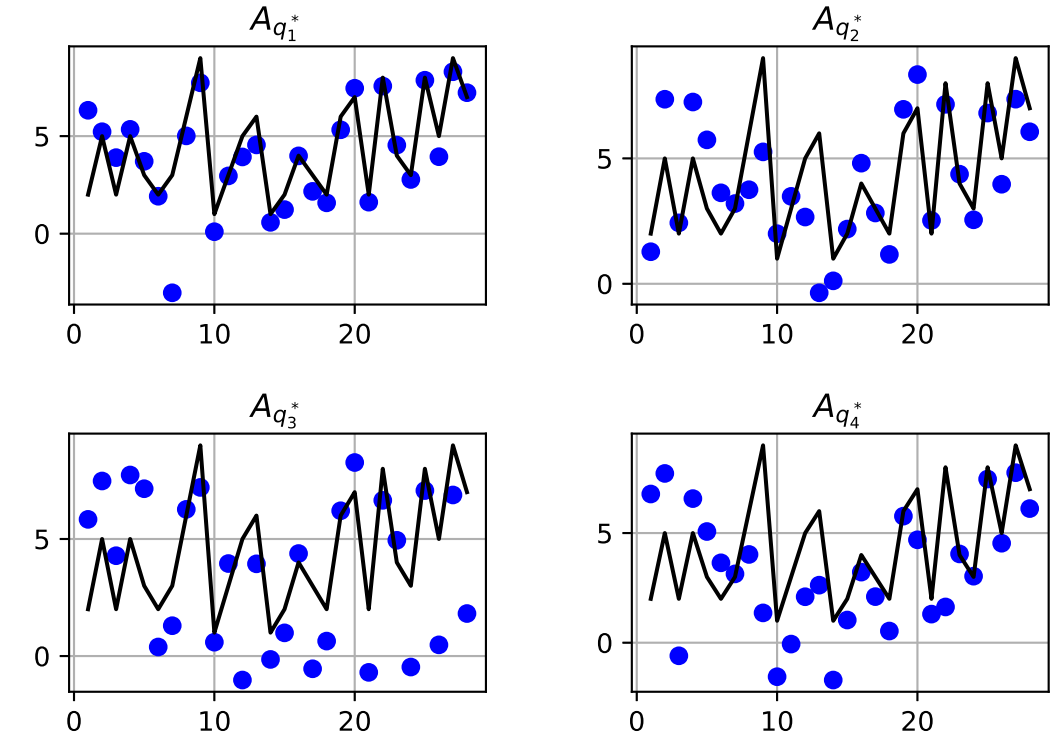


FIGURE 3.18: Simulation 9 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

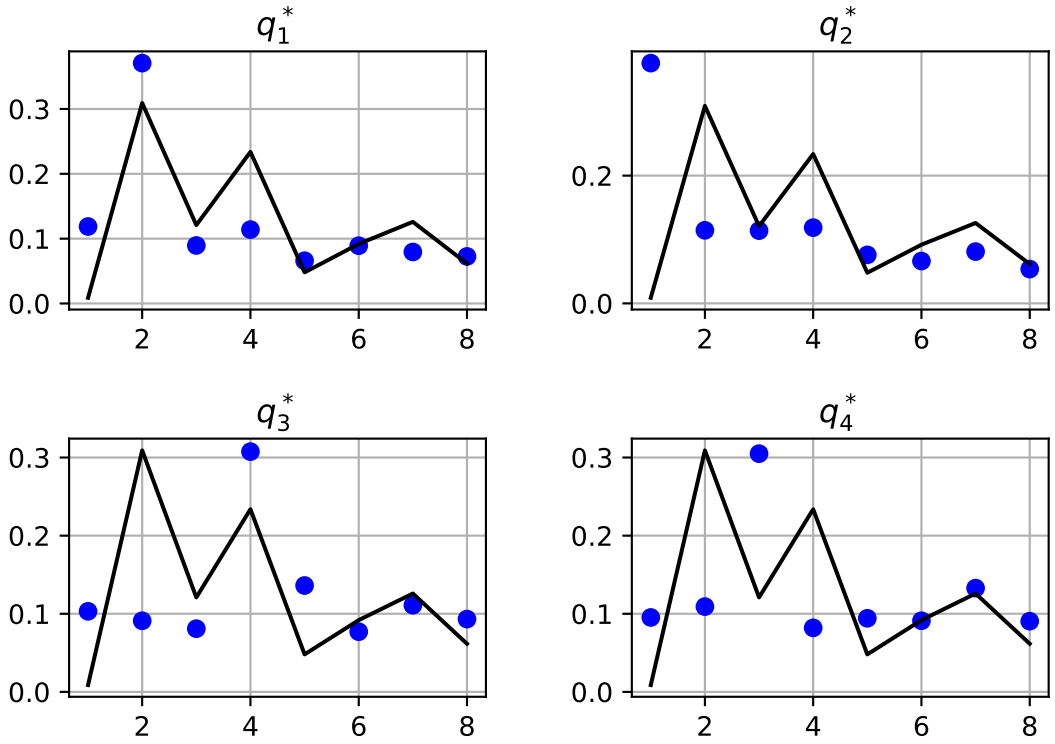


FIGURE 3.19: Simulation 10 of comparisons between $\{\mathbf{q}_h^*\}_{h=1}^4$ and randomly generated $\mathbf{q}_T$

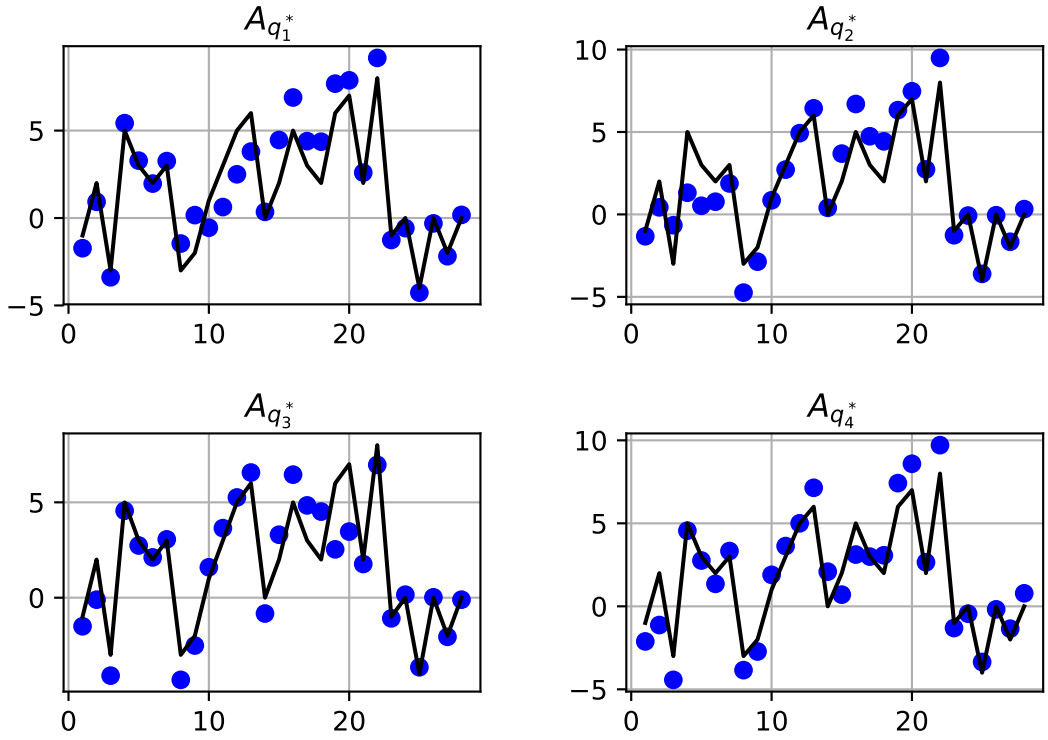


FIGURE 3.20: Simulation 10 of comparisons between $\{A_{\mathbf{q}_h^*}\}_{h=1}^4$ and randomly modified A_T

TABLE 3.1: Experiment Data

Item	Attribute description	Label
1	$(27, 17, 35, 24, 6, 6, 12, 25)$	1
2	$(18, 8, 16, 9, 2, 3, 3, 4)$	1
3	$(3, 3, 7, 1, 4, 0, 2, 3)$	1
4	$(1, 0, 3, 1, 0, 2, 1, 1)$	removed
5	$(17, 20, 36, 33, 33, 14, 6, 2)$	-1
6	$(8, 0, 14, 6, 1, 7, 1, 0)$	-1
7	$(2, 5, 11, 2, 0, 0, 1, 3)$	-1
8	$(1, 1, 3, 0, 1, 0, 1, 1)$	removed
9	$(14, 20, 17, 6, 1, 1, 0, 5)$	-1
10	$(6, 6, 4, 2, 0, 0, 0, 0)$	removed

Chapter 4

Pooling experiments for consecutive positives

4.1 Background

Group testing was proposed by Robert Dorfman during World War II in order to efficiently test a large number of blood samples for a rare disease. Since then, a large body of literature has enriched this problem, and various applications of group testing have been found in many fields such as multiple communication, coding theory, information security, sparse signal recovery and others. Particularly, biology-motivated group testing has been developed into one of the most important tools in the study of gene functions. For example, in Human Genome Project, well-designed group testing schemes have been proved to be useful in both saving materials and accelerating the process of reconstructing high-quality DNA libraries. These high-quality libraries have been frequently and repeatedly used for extensive studies.

A DNA library is a collection of cloned DNA segments taken from a specific organism. Those DNA segments are called clones. Determining whether a clone contains a particular DNA sequence of interest can be accomplished by screening it with a probe. The clone is called a positive for the probe if it contains a particular DNA sequence of interest, and a negative otherwise. Due to the large size of the library and the cost of time and materials of screening, in order to identify and isolate the clones containing the DNA sequence, instead of screening each clone individually, combinations of the clones are screened. The combinations of the clones are called pools. Each pool is screened with the probe to learn whether any of the clones in the pool contain the DNA sequence. Screening pools in this way is called a pooling experiment. Ideally, if a pool gives a negative outcome, then all clones in the pool are negatives, and otherwise the pool contains at least one positive. Generally speaking, when the proportion of positives is relatively small compared with the library size, many of the outcomes of the pools are expected to be negative, and hence the total number of tests is reduced.

Since the procedure of library screening is error prone and only a broad prior knowledge of the positives is available, a trade-off between complete identifiability and maximal efficiency has to be made and hence the goal of group testing for DNA library screening is to identify as many positives as possible by screening as few pools as possible.

In molecular biology, there will present a large quantity of information regarding experimental uncertainty, domain-specific prior knowledge and experimental observations. Therefore, scientific discovery is conducted in an interactive way and involves the task of experiment design again and again by taking into consideration a variety of domain-specific knowledge back and forth. This oftentimes requires an automatic knowledge discovery process to represent domain-specific structures

in the mind and computational procedures that operate on those structures. This chapter discusses a ranking method for detecting consecutive positives in a pooling experiment based on group testing techniques. We are more concerned with the cognitive aspect of the ranking model and emphasize on domain knowledge representation. More specifically, we show (1) how domain-specific prior knowledge regarding pooling experiments can be represented, (2) how this knowledge representation can be used for automatic knowledge discovery, (3) why and to what extent this knowledge representation will work for screening a structured DNA library.

4.1.1 Classification of group testing

Any group testing consists of a pooling procedure and a positive detecting procedure. A pooling procedure is a procedure of constructing a collection of pools called pooling design, determining which clones are put into which pools. A positive detecting procedure is a procedure of determining which clones are positives from the outcomes of group tests. While minimizing the number of tests is still important, biology-motivated group testing is simultaneously concerned with three other goals: (i) time consumption is minimized, (ii) error-tolerant ability is expected, and (iii) prior knowledge of positives is used to full capacity. Those goals usually merge. Depending on different experimental situations, various group testing schemes have been proposed to achieve a balance between those goals.

Group testing can be classified as either adaptive or nonadaptive, based on how a pooling design is constructed in the pooling procedure. In adaptive group testing, a pooling design is constructed in multiple-stage. In each stage, the outcomes of the group tests from previous stages are learned to construct the pools in the next stage. In nonadaptive group testing, a pooling design is constructed in one-stage. This is to say, pools are determined before any outcome of the group test is known. From the perspective of minimizing the number of tests, adaptive group testing is preferred to nonadaptive group testing, because adaptive group testing takes advantages of utilizing more information. However, things change if other goals are simultaneously considered. It is preferable to screen pools in parallel or with as few stages as possible in order to save time. Between fully adaptive and nonadaptive, Knill [100] proposed trivial two-stage pooling procedures of considerable interest. In the first stage, pools are screened in parallel, and a set of candidate positives is selected based on the outcomes of tests; in the second stage, each of the candidate positives is subject to an individual confirmatory test.

Group testing can be also classified as either combinatorial or probabilistic.To implement a combinatorial group testing, we have to construct a pooling design with desirable combinatorial properties such that all positives can be distinguished from negatives under the assumption on the maximum number of positives and that of experimental errors. Related studies can be found in Du and Hwang [84], [85], [29], Dyachkov et al. [46] , Macula [113], and Ngo and Du [128]. To implement a probabilistic group testing, not only stochastic models for positives and for pooling results are needed but also an efficient positive detecting algorithm to infer positives from erroneous pooling results based on a stochastic inference model. Probabilistic group testing is developed by Bruno et al. [21], Knill et al. [101], Mezard and Toninelli [116] and Uehara and Jimbo [174]. Bruno et al. [21] and Knill et al. [101] proposed a positive detecting algorithm called MCPD by using Markov chain Monte Carlo simulation method. Uehara and Jimbo [174] proposed another efficient algorithm called BNPD by using Bayesian network inference. Provided a proper pooling design, BNPD converges much faster than MCPD does. As far as we know, MCPD and

BNPD are the only known efficient positive detecting algorithms when experimental errors exist.

Probabilistic group testing has several merits in practice. First, from the perspective of detecting procedure, probabilistic group testing may perform stable when a relatively larger number of positives or/and experimental errors occur than expected. Second, probabilistic group testing can make good use of information and reduce information loss by interpreting a measurement of a test into a multilevel state such as "negative", "weak positive", "medium positive" or "strong positive". Third, from the perspective of pooling procedure, implementing probabilistic group testing requires fewer or no restriction on the combinatorial structure of pooling designs. Particularly, random pooling designs are allowed. In fact, random pooling designs are often preferred in a real setting. This is because (i) a well-designed random pooling design may have a satisfactory efficiency with desirable error-tolerant ability, (ii) it is unrealistic to expect to be able to find an appropriate combinatorial pooling design for every new pooling experiment, and (iii) random pooling designs facilitate robot automation and thus are efficient to construct.

4.1.2 Linear DNA library and consecutive positives

Motivated from applications to DNA library screening, Balding and Torney [7] considered the problem of pooling experiments for screening unique-sequence on a 1530-clone map of Aspergillus nidulans. The clone map has the properties that the clones are, with possibly a few exceptions, linearly ordered and no more than two of them cover any point on the genome. The goal of screening clone maps is to identify where a particular DNA segment occurs on a clone map.

In this problem, since the clones are overlapped, it may happen that one segment of interest occurs in a relatively large number of clones, but typically the number is predictable as it is related to the clone coverage. By introducing the assumption that the DNA segment occurs only once, Colbourn [34] introduced the d-consecutive property saying that the set of positives forms a consecutive set under the linear order and contains at most d positives, and applied combinatorial group testing to this problem. Following his work, related studies can be found, for example, in Müller and Jimbo [124] and[125], Juan and Chang [94] and Ge et al. [62]. By using heterogeneous priors, Bruno et al. [170] discussed optimization issues of nonadaptive random pooling designs on the assumption that an effective detecting procedure exists and experimental errors do not exist.

However, to our best knowledge, when experimental errors exist, probabilistic group testing for consecutive positives has not yet been studied. In this chapter, we study this problem to fill the gap.

4.1.3 Related works and new contributions

Since group testing is useful for finding genes and other features of interest on clone maps and probabilistic group testing is very helpful when experimental errors exist, we are motived to seek probabilistic group testing algorithms with consecutive positives. To this end, we make several contributions to this topic. First, we introduce a prior probability distribution for consecutive positives. This is done by first translating the d-consecutive positives property proposed by Colbourn [34] into a probability prior. Then, enlightened by the work of Uehara and Jimbo [174] where

Bayesian network is used as a probabilistic inference engine, we construct a reasonable prior probability distribution for consecutive positives with the overlap structure. This prior plays an important role both in constructing positive detecting algorithm and in optimizing pooling designs. Second, we apply Knill et al. [101]'s method to construct a positive detecting algorithm called MMCPD for identifying consecutive positives. It contains MCPD as a special case, depending on what form of the prior knowledge of positives is given.

Moreover, in order to optimize the positive detectability of MMCPD, we discuss how the presence of consecutive positives and experimental errors will affect the construction of random pooling designs. Our discussion leads to an efficient algorithm for explicitly estimating the information-theoretic lower bound of the minimal number of pools required for complete identifiability with high probability. This not only leads an efficient algorithmic procedure for choosing the parameters that controls the generating procedure of random pooling design, but also may be used to explain how these parameters influence the performance of the decoding algorithm. Our computation method shares some commonplaces either in purpose or in formulation with prior work such as Bruno et al. [170], Knill et al. [100], Wang et al. [181] and Wainwright [179]. However, the algorithmic consideration of consecutive positives and experimental errors makes our method different from any of them.

4.2 Stochastic models

This section first discusses how to collect and express prior knowledge of consecutive positives, and introduces a prior probability distribution which approximately incorporates the prior knowledge. Then, we show how Knill et al [101]'s method can be applied to detecting consecutive positives.

4.2.1 Prior knowledge of consecutive positives

The following notations will be consistently used throughout. Let $\mathcal{C} = \{c_1, c_2, ..., c_n\}$ be a set of clones. $\mathcal{C}$ is called a DNA library. Each clone c_i has an associated state $\sigma_i \in \{0, 1\}$. c_i is a positive if $\sigma_i = 1$, otherwise a negative. Let $P = \{c_i : \sigma_i = 1\}$ and $X_P = (\sigma_1, \ldots, \sigma_n)$ be the set of positives and its vector form corresponding to P, respectively. For convenience, P and X_P are used interchangeably, referred to the positive set. Denote by x_i the random variable of σ_i such that

$$x_i = \begin{cases} 0, & \text{if } \sigma_i = 0, \\ 1, & \text{if } \sigma_i = 1, \end{cases}$$

and by $X = (x_1, \ldots, x_n)$ the random vector of the associated states of $\mathcal{C}$.

When a DNA library is constructed by uniformly and randomly cloning of DNA segments, the expected number of positives $\mathbb{E}\big[\sum_{i=1}^{n} x_i\big] = d$ is used as the prior knowledge, for some positive number d. This prior knowledge appeared in Knill [100], Knill et al. [101] and Uehara and Jimbo [174]. The overlap information can analogously be collected and expressed in a similar form. The basic idea is to extend the d-consecutive positives property proposed by Colbourn [34] into a broader and probabilistic version.

$\mathcal{C}$ is said to be linear if it is associated with an linear order $c_i \prec c_{i+1}$, for $1 \leq i < n$. The positive set of linear $\mathcal{C}$ is said to have the multi-d-consecutive property if any

subset of P that forms a consecutive set (under the ordering $\prec$) contains at most d positives. A subset of positives P' is said to be a maximum consecutive subset, if P' is a consecutive set but $P' \cup \{c\}$ is not a consecutive set for any $c \in P \setminus P'$. Notice that P corresponds to a unique partition with maximum consecutive subsets. The positive set with the multi-d-consecutive property is allowed to have more than one maximum consecutive subsets, each of which contains at most d consecutive positives.

Without loss of generality and for the sake of simplicity, we analyze the overlap structure of the clone map of Aspergillus. When the clone map is screened, if the positive set is believed to approximately have the multi-2-consecutive positive property, then the following prior knowledge of the positives can be collected:

Info 1: the portion of positives among all clones is relatively small;

Info 2: although the positives are sparse, some of the positives tend to be consecutive;

Info 3: although some of the positives tend to be consecutive, maximum consecutive subsets tend to not get close to each other;

Info 4: no more than two of the positives tend to form a maximum consecutive subset.

Therefore, based on Info 1 through Info 4, we can express the prior knowledge of consecutive positives as follows:

$$\begin{cases} \mathbb{E}\big[\sum\limits_{i=1}^{n} x_i\big] = d_1 \\ \mathbb{E}\big[\sum\limits_{i=1}^{n-1} x_i x_{i+1}\big] = d_2 \\ \mathbb{E}\big[\sum\limits_{i=1}^{n-2} x_i x_{i+2}\big] = d_3 \\ \mathbb{E}\big[\sum\limits_{i=1}^{n-2} x_i x_{i+1} x_{i+2}\big] = d_4 \end{cases} \tag{I}$$

for some positive numbers d_1, d_2, d_3 and d_4. However, the values of d_1 through d_4 depend on both segments of interest and structures of clone maps, and is not within our present concern. Formally, we introduce the following assumption.

Assumption 1. *Appropriate values of d_1 through d_4 have been given as parameters.*

Generally, we denote by $\mathcal{D}$ a family of polynomials of $\sigma_1, \ldots, \sigma_n$, where

$$\mathcal{D} = \Big\{ \sum_{i=1}^{t} \prod_{c_j \in \mathcal{C}_i} \sigma_j : \exists t \in \mathbb{N} \text{ and } \exists\ t \text{ distinctive sets} \\ \mathcal{C}_1, \ldots, \mathcal{C}_t \in 2^{\mathcal{C}} \setminus \{\emptyset\} \text{ such that } \mathcal{C} \subseteq \bigcup_{i=1}^{t} \mathcal{C}_i \Big\}.$$

Each polynomial of $\mathcal{D}$ is called an overlap polynomial of $\mathcal{C}$. We can collect and explicitly express prior knowledge of consecutive positives by using overlap polynomials in accordance with overlap information. If s overlap polynomials D'_j have been chosen from $\mathcal{D}$, P is said to have positive pattern $\boldsymbol{d}$ if $(\boldsymbol{d})_j = D'_j(X_P)$, for $j = 1, \ldots, s$. In addition, we denote by $|\boldsymbol{d}|$ the number of positives in positive pattern $\boldsymbol{d}$.

4.2.2 Prior probability distribution of consecutive positives

In this part we seek a prior probability distribution that, in some sense of approximation, incorporates the prior knowledge of consecutive positives (I).

The principle of maximum entropy

Since Shannon's theorem [156] established the uniqueness of entropy as an information measure of uncertainty, the principle of maximum entropy has been wildly used to derive prior probability distributions. Intuitively speaking, more information means less uncertainty. Any probability distribution satisfying the constraints that has less uncertainty will contain more information, and thus implies something stronger than what the prior knowledge means. The principle of maximum entropy, as a method of statistical inference, is due to Jaynes [87], [88] and [89].

Based on the principle of maximum entropy, to derive the prior probability distribution $\mathbb{P}(X)$ which incorporates (I) but is free from any other knowledge, is to solve the following problem,

$$\begin{aligned} \underset{\mathbb{P}(X)}{\text{maximize}} \quad & -\sum_{X\in\{0,1\}^n} \mathbb{P}(X)\log\mathbb{P}(X) \\ \text{subject to} \quad & \begin{cases} \mathbb{E}_{\mathbb{P}}\big[D_1(X)\big] = d_1 \\ \mathbb{E}_{\mathbb{P}}\big[D_2(X)\big] = d_2 \\ \mathbb{E}_{\mathbb{P}}\big[D_3(X)\big] = d_3 \\ \mathbb{E}_{\mathbb{P}}\big[D_4(X)\big] = d_4 \\ \sum_{X\in\{0,1\}^n}\mathbb{P}(X) = 1. \end{cases} \end{aligned} \tag{II}$$

As is well-known, Lagrange multiplier method leads to the solution

$$\mathbb{P}_\theta(X) = \frac{1}{Z(\theta)}\exp\Big\{-\sum_{j=1}^{4}\theta_j D_j(X)\Big\}, \tag{III}$$

for some constant vector $\theta = (\theta_1,\theta_2,\theta_3,\theta_4)$ and constant $Z(\theta)$. In literature, (III) is called a Gibbs measure with energy function $-\sum_{j=1}^{4}\theta_j D_j(X)$. The normalization constant

$$Z(\theta) = \sum_{X\in\{0,1\}^n}\exp\Big\{-\sum_{j=1}^{4}\theta_j D_j(X)\Big\}$$

is called the partition function. It connects θ with the constants d_i by simultaneous equations

$$\frac{\partial -\log Z(\theta)}{\partial\theta_i} = d_i, \tag{IV}$$

for $i = 1,\dots,4$. Before discussing how to derive $\boldsymbol{\theta}$ from (IV), we first point out three useful properties of (III).

Property 1 (Consistency property). *If $\mathbb{E}\big[\sum_{i=1}^{n} x_i\big] = d$ $(d > 0)$ is the only prior knowledge of positives, then the prior probability distribution determined by the principle of maximum entropy is*

$$\mathbb{P}(X) = \left(\frac{d}{n}\right)^{\sum_{i=1}^{n}x_i}\left(1-\frac{d}{n}\right)^{n-\sum_{i=1}^{n}x_i}.$$

The proof can be found in Jaynes [89]. From this property, we see that, provided proper prior knowledge of positives, the principle of maximum entropy can be used to obtain the prior probability used in Bruno et al [21], Knill [101], Knill et al [100] and Uehara [174], where the DNA library is considered to have been constructed by uniformly and randomly cloning of DNA segments. Hence, maximum-entropy distributions may be reasonable extensions for describing the overlap structure of clone maps.

Denote by N_j the set of neighbors of j. Define $N_1 \triangleq \{2,3\}$, $N_2 \triangleq \{1,3,4\}$, $N_{n-1} \triangleq \{n-3, n-2, n\}$, $N_n \triangleq \{n-2, n-1\}$ and $N_i \triangleq \{i-2, i-1, i+1, i+2\}$, for $3 \leq i \leq n-2$. For any configuration of x_i by assigning $x_i = \sigma_i$ for each $i \in [n]$ according to $\mathbb{P}_\theta(X)$, (III) has the following Markov-type property.

Property 2 (Local Markov property). *For any $j \in [n]$,*

$$\mathbb{P}_\theta(x_j|\sigma_i, i \in [n] \setminus \{j\}) = \mathbb{P}_\theta(x_j|\sigma_i, i \in N_j).$$

We show a sketch of the proof. A more general one can be found in Kindermann and Snell [98] and Pearl [135]. To see this, we rewrite the Gibbs measure into the product form:

$$\begin{aligned}\mathbb{P}_\theta(X) =& \frac{1}{Z(\theta)} \prod_{i=1}^{n} \exp\{-\theta_1 x_i\} \prod_{i=1}^{n-1} \exp\{-\theta_2 x_i x_{i+1}\} \\ & \prod_{i=1}^{n-2} \exp\{-\theta_3 x_i x_{i+2}\} \prod_{i=1}^{n-2} \exp\{-\theta_4 x_i x_{i+1} x_{i+2}\}.\end{aligned}$$

For $\sigma_j = 0, 1$, the condition probability on the left side is,

$$\begin{aligned}&\mathbb{P}_\theta(x_j = \sigma_j|\sigma_i, i \in [n] \setminus \{j\}) \\ =& \frac{\mathbb{P}_\theta(\sigma_1, \ldots, \sigma_{j-1}, x_j = \sigma_j, \sigma_{j+1}, \ldots, \sigma_n)}{\sum_{\sigma_j=0}^{1} \mathbb{P}_\theta(\sigma_1, \ldots, \sigma_{j-1}, x_j = \sigma_j, \sigma_{j+1}, \ldots, \sigma_n)}.\end{aligned}$$

Thus, after canceling out the normalization constant, terms of $\mathbb{P}_\theta(\sigma_1, \ldots, \sigma_n)$ that do not contain σ_j cancel from both the numerator and denominator of the condition probability and therefore this probability depends only on the random value of x_j and those of its neighbors.

This property serves a probabilistic interpretation of multi-2-consecutive property, saying that the state of a clone is only influenced by the states of its previous two and also next two neighbors, if the clone has such neighbors.

Property 3 (Heterogenous property). *Given $X_1, X_2 \in \{0,1\}^n$, if $D_i(X) = D_i(X_2)$, for $i = 1, \ldots, 4$, then $\mathbb{P}_\theta(X_1) = \mathbb{P}_\theta(X_2)$.*

The proof is obvious. Notice that, for any fixed positive pattern $\boldsymbol{d}$, $\mathbb{P}_\theta$ is the heterogenous prior with the greatest uncertainty.

Estimation of lagrange multiplier

The constants θ_j are called Lagrange multipliers. As the same values of d_j's may come from different information sources, different estimation methods may be needed to derive θ from (IV). Here we discuss two cases. In the first case, the values of d_j's

represent the degrees of belief. The belief is needless to be relevant with the outcomes of any random experiment. While in the second case, the values are obtained from the observation of the data $Y = \{Y_1, \ldots, Y_N\}$.

In the former case, we define

$$f(\theta) = -\sum_{j=1}^{4} d_j\theta_j - \log Z(\theta).$$

From (IV), we see that θ can be any stationary point of $f(\theta)$. Therefore, to obtain θ is to solve a gradient-square-minimization problem.

$$\underset{\theta\in\mathbb{R}^4}{\text{minimize}} \quad \|\nabla f(\theta)\|^2. \tag{V}$$

However, in the latter case, we may fit a model chosen from $\{\mathbb{P}_\theta : \theta \in \mathbb{R}^4\}$ to the given data. Assuming that the data $Y_1, \ldots, Y_N$ are i.i.d (independent and identically distributed) random sample drawn from $\mathbb{P}_\theta$ with unknown parameter θ, we estimate θ by applying maximum likelihood estimation method. The log-likelihood is defined by

$$\begin{aligned} &\log \mathbb{P}(\{Y_1, \ldots, Y_N\}|\theta) \\ =& \log \prod_{i=1}^{N} \mathbb{P}(Y_i|\theta) \\ =& \sum_{i=1}^{N} \Big(-\sum_{j=1}^{4} N_j(Y_i)\theta_j - \log Z(\theta) \Big). \end{aligned} \tag{$*$}$$

Let $l_Y(\theta) = \frac{1}{N}\log \mathbb{P}(\{Y_1, \ldots, Y_N\}|\theta)$ and $d_j = \frac{1}{N}\sum_{i=1}^{N} D_j(Y_i)$, for $j = 1, \ldots, 4$. To obtain a maximum likelihood estimate of θ is to maximize $(*)$. Equivalently, we solve

$$\underset{\theta\in\mathbb{R}^4}{\text{maximize}} \quad l_Y(\theta) = -\sum_{j=1}^{4} d_j\theta_j - \log Z(\theta). \tag{VI}$$

Both problems are very difficult to solve, because the partition function $Z(\theta)$ involves exponentially many computations and usually cannot be known thus. Without calculating the partition function, Geyer and Thompson [64] developed a method to numerically solve (VI) by using importance sampling and Monte Carlo simulation. Descombes et al [41] further demonstrated an efficient conjugate gradient algorithm. Motived by their work, we employ their methods to solve (V). Here we sketch that (V) is also solvable (see [64] and [41] for detailed discussions).

The key idea is to estimate, for any fixed θ, $\|\nabla f(\theta)\|^2$ and its gradient by using importance sampling.

By employing

$$\mathbb{P}_\psi(X) = \frac{1}{Z(\psi)} \exp\Big\{ -\sum_{j=1}^{4} \psi_j D_j(X) \Big\},$$

we reformulate $f(\theta)$ as follows:

$$f(\theta) = -\sum_{j=1}^{4} d_j\theta_j - \log \frac{Z(\theta)}{Z(\psi)} - \log Z(\psi).$$

It follows that

$$\begin{aligned} Z(\theta) &= \sum_X \exp\left\{ -\sum_{j=1}^4 (\theta_i - \psi_i) D_j(X)) \right\} \\ &\quad \cdot \exp\left\{ -\sum_{i=j}^4 \psi_j D_j(X) \right\} \\ &= \mathbb{E}_\psi \left[\exp\left\{ -\sum_{j=1}^4 (\theta_i - \psi_i) D_j(X) \right\} Z(\psi) \right], \end{aligned}$$

where $\mathbb{E}_\psi$ refers to the expectation with respect to $\mathbb{P}_\psi$. Then, we obtain

$$\frac{Z(\theta)}{Z(\psi)} = \mathbb{E}_\psi \left[\exp\left\{ -\sum_{j=1}^4 (\theta_j - \psi_j) D_j(X) \right\} \right]. \tag{2.1}$$

The significance of (2.1) lies in that although $Z(\theta)$ is unknown, $\frac{Z(\theta)}{Z(\psi)}$ can be estimated by a sampling of the known probability distribution $\mathbb{P}_\psi$.

Following this observation, we continue to reformulate the partial derivatives of $Z(\theta)$ up to multiplicative constant $\frac{1}{Z(\psi)}$, that is,

$$\frac{1}{Z(\psi)} \frac{\partial Z(\theta)}{\partial \theta_j} = \mathbb{E}_\psi \left[-D_j(X) \exp\left\{ -\sum_{j=1}^4 (\theta_i - \psi_i) D_j(X) \right\} \right]. \tag{2.2}$$

Similarly, higher-order derivatives of $Z(\theta)$ up to multiplicative constant $\frac{1}{Z(\psi)}$ can also be obtained. For example,

$$\frac{1}{Z(\psi)} \frac{\partial^2 Z(\theta)}{\partial \theta_j \partial \theta_k} = \mathbb{E}_\psi \left[D_j(X) D_k(X) \exp\left\{ -\sum_{j=1}^4 (\theta_i - \psi_i) D_j(X) \right\} \right]. \tag{2.3}$$

Next, with (2.1), (2.2) and (2.3) we have,

$$\begin{aligned} &\frac{\partial}{\partial \theta_k} \|\nabla f(\theta)\|^2 \\ =&2 \sum_{j=1}^4 \left(d_j + \frac{Z(\psi)}{Z(\theta)} \frac{1}{Z(\psi)} \frac{\partial Z(\theta)}{\partial \theta_j} \right) \left(-\left(\frac{Z(\psi)}{Z(\theta)} \right)^2 \left(\frac{1}{Z(\psi)} \right. \right. \\ &\left. \left. \frac{\partial Z(\theta)}{\partial \theta_k} \right) \left(\frac{1}{Z(\psi)} \frac{\partial Z(\theta)}{\partial \theta_j} \right) + \frac{Z(\psi)}{Z(\theta)} \left(\frac{1}{Z(\psi)} \frac{\partial^2 Z(\theta)}{\partial \theta_j \partial \theta_k} \right) \right). \end{aligned} \tag{2.4}$$

By using the Gibbs sampler (Geman and Geman [63]), for any fixed θ, by $\mathbb{P}_\psi$ we can theoretically estimate (2.1), (2.2), (2.3) and hence even higher-order partial derivatives of $Z(\theta)$ up to constant $\frac{1}{Z(\psi)}$. This allows us to learn local variation of $\|\nabla f(\theta)\|^2$ at any fixed θ. Those estimations will be helpful to numerically sovle (V) if they can be obtained efficiently. The local markov property of $\mathbb{P}_\psi$ will make the sampling procedure efficient as long as every random variable has few neighbors. We refer to Descombes et al [41] for other algorithmic details.

In fact, (VI) is stronger than (V) in the sense that any optimal of (VI) is also an optimal of (V), but not the reverse. Therefore, we may also formulate the first case in the stronger sense,

$$\underset{\theta\in\mathbb{R}^4}{\text{maximize}} \quad f(\theta) = -\sum_{j=1}^{4} d_j\theta_j - \log Z(\theta). \tag{VII}$$

From this formulation, we can see that both cases are numerically solvable, but the first case is more general than the second one. By estimating a numerical solution of (V) within a given tolerance, the model can be approximately determined, and it can be served as the desired prior probability distribution that approximately incorporates our prior knowledge derived from the available information.

Assumption 2. *Given the values of d_1 through d_4 under Assumption 1, $\widehat{\theta}$ is an approximate optimal of (V). $\mathbb{P}(X)$ is the desired prior probability distribution.*

4.2.3 Stochastic model of pooling results

The response to a screening test is the outcome of the screening test, called pooling result. The following notations concerning pooling experiments will be consistently used. Let $\mathcal{A} = \{p_1, \ldots, p_m\}$ be a pooling design. Each pool p_i is a subset of $\mathcal{C}$ corresponding to the clones in the pool. The pooling design $\mathcal{A}$ constructed by determining each of n clones is put into which of m pools, can be represented by an $m \times n$ binary matrix $A = (a_{ij})$. Each entry a_{ij} is defined as follows:

$$a_{ij} = \begin{cases} 1, & \text{if } c_j \in p_i, \\ 0, & \text{if } c_j \notin p_i. \end{cases}$$

Then, A is called the incidence matrix of pooling design $\mathcal{A}$ and m the size of $\mathcal{A}$. For convenience, $\mathcal{A}$ and A are interchangeably referred to a pooling design.

The pooling result of screening pool p_i is denoted by $r(p_i)$. Sine $r(p_i)$ is automatically measured by a fluorescence sign and an actual observation of $r(p_i)$ is often given in multilevel measurement, taking any value from a set of test outcomes V, such as "negative", "weak positive", "medium positive" or "strong positive". Unfortunately, the pooling results are typically corrupted due to the inclusion of additional clones in a pool or to the failure of screening test. Therefore, we need infer P from erroneous pooling results.

To begin with, we introduce the stochastic model of pooling results given in Knill et al. [101]. The following assumptions will be used.

Assumption 3 (Knill et al. [101])**.** *The distribution of $r(p_i)$ depends only on the number of positives, $|p_i \cap P|$, in p_i.*

Assumption 4 (Knill et al. [101])**.** *Since the PCR was used for screening, we assume that $\Pr(r(p_i)||p_i \cap P|)$ depends only on $|p_i \cap P| = 0$ or $|p_i \cap P| \geq 1$.*

For any integer b, we define

$$\overline{b} = \begin{cases} 0 & \text{if } b = 0, \\ 1 & \text{if } b \geq 1. \end{cases}$$

Notice that, for each i, the number of positives in pool p_i can be represented by the ith coordinate of AX_P, that is, $|p_i \cap P| = (AX_P)_i$. Let R_i be a random variable of pooling result $r(p_i)$,

$$R_i = \begin{cases} 0 & \text{if } r(p_i) \text{ is negative,} \\ 1 & \text{if } r(p_i) \text{ is weak positive,} \\ 2 & \text{if } r(p_i) \text{ is medium positive,} \\ 3 & \text{if } r(p_i) \text{ is strong positive.} \end{cases}$$

Rewrite $\Pr(R_i = r_i|(AX_P)_i)$ into $f(r_i, (AX_P)_i)$ for short, and denote by $\Pr_A(R = r|X_P)$ the likelihood for the pooling results $r(\in \{0,1,2,3\}^m)$ obtained from pooling design A with m pools. By using Assumptions 3 and 4 and the chain rule, it can be expressed as a product:

$$\Pr_A(R = r|X_P) = \prod_{i=1}^{m} f(r_i, (\overline{AX_P})_i). \tag{VIII}$$

Let $f = \{f(j,i) : j = 0,\ldots,3, \text{ and } i = 0,1\}$. In real applications, f can be appropriately estimated. In this application, we are more concerned with how different values of $f(j,i)$ will affect identifiability of pooling experiments. Therefore, we assume that f have been estimated and provided as parameters.

Assumption 5. *The values of $f(j,i)$ are known a priori, but $f(j,i) \neq 0$, for $j = 0,\ldots,3$, and $i = 0,1$.*

Particularly, notice that $f(1,0) + f(2,0) + f(3,0)$ is the likelihood of a false positive, whereas $f(0,1)$ is that of a false negative. Moreover, due to Assumption 3 and Assumption 4, the model does not require any restriction on the structure of positives, and thus is compatible with the overlap structure of consecutive positives.

4.2.4 Bayes inference model

To infer the consecutive positives, Bayes inference model is used to decode the erroneous pooling results. Denoting by $\Pr_A(X_P|r)$ the posterior probability that X_P is the positive set given the pooling results r under the pooling design A, by Bayes' rule we have

$$\Pr_A(X_P|r) \propto \Pr_A(X_P)\Pr_A(r|X_P),$$

where $\Pr_A(X_P)$ denotes the probability that X_P is the positive set given pooling design A is chosen to use. Prior knowledge of positives will affect the construction of pooling design A. However, the reverse does not reasonably hold, because knowing the construction of pooling design A will not add any information to the prior knowledge of positives. Therefore, we introduce the following assumption.

Assumption 6. *Given any $A \in \mathcal{A}$, $\Pr_A(X) = \mathbb{P}_{\hat{\theta}}(X)$, for all $X \in \{0,1\}^n$.*

From Assumption 2, Assumption 5, Assumption 6 and (VIII) , it follows that the posterior probability is computable up to a multiplicative constant, and can be written as,

$$\Pr_A(X_P|r) \propto \mathbb{P}_{\hat{\theta}}(X_P) \prod_{i=1}^{m} f(r_i, (\overline{AX_P})_i).$$

If we are interest in $\Pr_A(c \in P|r)$ for each clone c, this probability can be written as the marginal

$$\Pr_A(x_j = 1|r) \propto \sum_{\substack{P \subseteq \mathcal{C}: \\ c_j \in P}} \mathbb{P}_{\hat{\theta}}(X_P) \prod_{i=1}^{m} f(r_i, (\overline{AX_P})_i). \tag{IX}$$

Substituting Binomial distribution for $\mathbb{P}_{\widehat{\theta}}$, the inference model turns out to be the one used in Knill et al. [101]. Actually, this is least surprise, since they had already pointed out the possibility that other distributions can be used in the inference model.

4.3 Detecting algorithm for consecutive positives

This section presents a positive detecting algorithm for consecutive positives, called modified Markov chain Monte Carlo pool result decoder (MMCPD). MMCPD can be seen as a natural extension of MCPD, depending on the prior knowledge of positives.

Our primary goal is to compute $\Pr_A(x_j = 1|r)$, for each j, and to choose the clones with highest posterior probability of being a positive for confirmatory individual tests. Those clones are called candidate positives. However, as it stands in (IX), every exact computation will involve exponential work in n, making our primary goal impractical when n is very large. Instead, we estimate the posterior probabilities by drawing samples approximately according to $\Pr_A(X|r)$. This idea works as long as a sampling method exists and is able to efficiently produce sufficiently many samples according to $\Pr_A(X|r)$.

To sample from $\Pr_A(X|r)$, we can use the Gibbs sampler, since the full conditional distributions $\Pr_A(x_i|\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_n, r)$ are easy to obtain for each i, due to the local markov property of $\mathbb{P}_{\widehat{\theta}}(X)$ and to the product expression of $\Pr(r|X)$.

In this way, Knill et al. [101]'s method can be extended for detecting consecutive positives. Mimicking their approach, we construct a Markov chain $X_0, X_1, \ldots$ on the family of all configurations of $\mathcal{C}$ with $\Pr_A(X|r)$ as its stationary distribution. Denote by X_S the vector form of the positive set S. Let X_S be the configuration of $\mathcal{C}$ at the end of step $t-1$, that is, state X_{t-1}. Step t of the chain updates X_S by making a random decision for each of the n clones on whether to add it to or remove it from S. At the end of step t, that is, after all clones have been processed, state X_t is the final X_S. In the Gibbs sampler, for clone c_k, the probability of changing from X_S to $X_{S\triangle\{c_k\}}$ equals to

$$\frac{1}{1+\dfrac{\Pr_A(X_S|r)}{\Pr_A(X_{S\triangle\{c_k\}}|r)}}, \tag{X}$$

where $\triangle$ is the symmetric difference, that is, $A\triangle B \triangleq (A \setminus B) \cup (B \setminus A)$. Notice that with this probability of changing from X_S to $X_{S\triangle\{c_k\}}$, X_t and hence X_S tend in distribution to $\Pr_A(X|r)$. From Bayes' Theorem, $\mathbb{P}_{\widehat{\theta}}(X)$ and (VIII) we obtain,

$$\begin{aligned}&\frac{\Pr_A(X_S|r)}{\Pr_A(X_{S\triangle\{c_k\}}|r)}\\ =&\frac{\mathbb{P}_{\widehat{\theta}}(X_S)}{\mathbb{P}_{\widehat{\theta}}(X_{S\triangle\{c_k\}})}\frac{\prod\limits_{\substack{i\in[m]:\\ c_k\in p_i}} f(r_i, \overline{(AX_S)}_i)}{\prod\limits_{\substack{i\in[m]:\\ c_k\in p_i}} f(r_i, \overline{(AX_{S\triangle\{c_k\}})}_i)}.\end{aligned}$$

Notice that, for i such that $c_k \in p_i$, we have

$$(AX_{S\triangle\{c_k\}})_i = \begin{cases}(AX_S)_i + 1, & \text{if } c_k \notin S,\\ (AX_S)_i - 1, & \text{if } c_k \in S.\end{cases}$$

The prior ratio $\frac{\mathbb{P}_{\widehat{\theta}}(X_S)}{\mathbb{P}_{\widehat{\theta}}(X_{S\triangle\{c_k\}})}$ can also be efficiently updated and we will see that it only depends on the associated states of c_k's neighbors. For convenience, let $X_S = (\sigma_1^S, \ldots, \sigma_k^S, \ldots, \sigma_n^S)$ and $\sigma_{-1}^S = \sigma_0^S = \sigma_{n+1}^S = \sigma_{n+2}^S = 0$, for any S. If $c_k \notin S$, then

$$\frac{\mathbb{P}_{\widehat{\theta}}(X_S)}{\mathbb{P}_{\widehat{\theta}}(X_{S\triangle\{c_k\}})} = \exp\{\widehat{\theta}_1^S + \widehat{\theta}_2\sigma_{k-1}^S + \widehat{\theta}_2\sigma_{k+1}^S + \widehat{\theta}_3\sigma_{k-2}^S + \widehat{\theta}_3\sigma_{k+2}^S + \widehat{\theta}_4\sigma_{k-2}^S\sigma_{k-1}^S + \widehat{\theta}_4\sigma_{k-1}^S\sigma_{k+1}^S + \widehat{\theta}_4\sigma_{k+1}^S\sigma_{k+2}^S\}.$$

If $c_k \in S$, then

$$\frac{\mathbb{P}_{\widehat{\theta}}(X_S)}{\mathbb{P}_{\widehat{\theta}}(X_{S\triangle\{c_k\}})} = \exp\{-\widehat{\theta}_1 - \widehat{\theta}_2\sigma_{k-1}^S - \widehat{\theta}_2\sigma_{k+1}^S - \widehat{\theta}_3\sigma_{k-2}^S - \widehat{\theta}_3\sigma_{k+2}^S - \widehat{\theta}_4\sigma_{k-2}\sigma_{k-1}^S - \widehat{\theta}_4\sigma_{k-1}^S\sigma_{k+1}^S - \widehat{\theta}_4\sigma_{k+1}^S\sigma_{k+2}^S\}.$$

From Assumption 4 and that $\mathbb{P}_{\widehat{\theta}}$ is strictly positive (that is, $\mathbb{P}_{\widehat{\theta}}(X) > 0$ for any X), we can see that the Markov chain is aperiodic, since it has a positive probability of remaining in the same state, and that the Markov chain is also irreducible, since it is possible to go from any state to any other state. This assures uniqueness of and convergence to the stationary distribution $\Pr_A(X|r)$ as well as the ergodic property. For discussions of the Gibbs sampler, we refer to Roberts and Smith [149] and Tierney [168]. Hence, starting with any state and running Gibbs sampler algorithm after a suitable warmup period, samples obtained from a realization of the Markov chain can approximately be regarded as the desired ones drawn according to $\Pr_A(X|r)$. By taking sufficiently many samples from the Markov chain, the proportion of the samples with $\sigma_i = 1$ can be thus a Bayesian estimation of $\Pr_A(x_i = 1|r)$.

As an extension of MCPD, MMCPD has two attractive merits. First, only local neighborhood relations and state values are needed to update the prior ratio and likelihood ratio, making the computation efficient. Second, its implementation requires no explicit restriction on the structure of pooling designs. This provides us a considerable freedom to choose and optimize the pooling designs we want to use, rather than the ones we have to use. Particularly, MMCPD is able to decode the pooling results obtained by screening random pooling designs.

4.4 Random k-set design

This section discusses random pooling designs in the presence of consecutive positives and experimental errors.

4.4.1 Motivation and related works

Among all, we are particularly interested in random k-set designs. A is said to be a random k-set design if each column of A is a k-subset of $[m]$ that is generated independently and uniformly at random. k is called the replication number, deciding how many pools each clone will be put into. Since there may exist other positive detecting algorithms for consecutive positives, a practical lower bound of the optimal value of m, independent of any positive detecting algorithm, will be attractive. This motivated us to seek a nontrivial information-theoretic lower bound of m. Information-theoretic bounds have been extensively studied in many fields. For example, related studies can be found in Atia and Saligrama [6], Chan et al [25] and [26], Wang et al. [181] and Wainwright [179]. Our computation method shares some

commonplaces either in purpose or in formulation with prior work such as Bruno et al. [170], Knill et al. [100], Wang et al. [181] and Wainwright [179]. However, the algorithmic consideration of consecutive positives and experimental errors with random k-set designs makes our method different from any of them.

4.4.2 Problem formulation

Notice that the Gibbs sampler (Geman and Geman [63]) allows us to estimate the distribution of the positive patterns of $\mathbb{P}_{\widehat{\theta}}$. Hence, to choose proper values of m and k for $\mathbb{P}_{\widehat{\theta}}$ and f, we begin by fixing a positive pattern $\boldsymbol{d}$ and formulate the subproblem in terms of $\boldsymbol{d}$ and f.

Let $\Omega(\boldsymbol{d}) \subseteq \left\{(\sigma_1, \ldots, \sigma_n) \in \{0,1\}^n : \sum_{i=1}^{n} \sigma_i = |\boldsymbol{d}|\right\}$ be the nonempty subset consisting of all the vectors with positive pattern $\boldsymbol{d}$. If X_P belongs to $\Omega(\boldsymbol{d})$, due to the heterogenous property of $\mathbb{P}_{\widehat{\theta}}$, we think of X_P randomly and uniformly distributed over $\Omega(\boldsymbol{d})$. Denote by $\mathcal{A}_{m,k}$ the set of all k-set designs with size m. Similar to Assumption 6, only knowing the k-set design does not change the distribution of X_P over $\Omega(\boldsymbol{d})$. We formally states this by introducing Assumption 7.

Without causing confusion, when $\boldsymbol{d}$ is fixed and known, we denote by $\Pr(X)$ the probability that X is chosen from $\Omega(\boldsymbol{d})$ as the positive set and by $\Pr_A(X)$ the probability that X is the positive set given k-set design A.

Assumption 7. *When $\boldsymbol{d}$ is given and the values of m and k have been decided, given any k-set design $A \in \mathcal{A}_{m,k}$, $\Pr_A(X) = \Pr(X)$, for all $X \in \Omega(\boldsymbol{d})$.*

Given any instance of random k-set design A, a positive detecting algorithm ϕ with respect to A is a mapping from the m-vector observation r to an estimated vector of positives, say of the form $X_{\widehat{P}} = \phi_A(r)$. Accordingly, based on the k-set design A, the average probability of decoding error of any positive detecting algorithm is defined as

$$p_{err}(A) = \frac{1}{|\Omega(\boldsymbol{d})|} \sum_{X \in \Omega(\boldsymbol{d})} \Pr[\phi_A(r) \neq X | X].$$

We apply Fano's lemma [36] to lower bound the average probability of decoding error. Notice that $X \xrightarrow{A} \overline{AX} \xrightarrow{f} r$ forms a Markov chain, we can arrive at the form used in Wang et al. [181]

$$p_{err}(A) \geq 1 - \frac{H_A(r) - H_A(r|X) + 1}{\log |\Omega(\boldsymbol{d})|}.$$

The average probability of decoding error over $\mathcal{A}_{m,k}$ can be lower bounded as follows:

$$\mathbb{E}_A[p_{err}(A)] \geq 1 - \frac{\mathbb{E}_A\left[\sum_{i=1}^{m} H_A(r_i)\right] - \mathbb{E}_A[H_A(r|X)] + 1}{\log |\Omega(\boldsymbol{d})|}. \tag{XI}$$

4.4.3 A lower bound of $\mathbf{E}_A[p_{err}(A)]$

Provided n, $\boldsymbol{d}$ and f are fixed and known, we begin by stating a set of lemmas and a necessary condition on m and k for $\mathbb{E}_A[p_{err}(A)] = 0$, on the conditions that the positive set X with positive pattern $\boldsymbol{d}$ is randomly and uniformly distributed over

$\Omega(\boldsymbol{d})$ and A is random k-set design, independently and uniformly from $\mathcal{A}_{m,k}$. The lemmas and hence Necessity Theorem are also based on the Assumptions 3, 4 and 7.

Lemma 1. *For any $h \in [m]$,*

$$\mathbb{E}_A[H_A(r_h)] \leq -\sum_{j=0}^{3} f_{\boldsymbol{d},m,k}(j) \log f_{\boldsymbol{d},m,k}(j),$$

where

$$f_{\boldsymbol{d},m,k}(j) = \sum_{i=0}^{1} \lambda_{\boldsymbol{d},m,k}^{1-i}(1-\lambda_{\boldsymbol{d},m,k})^i f(j,i),$$

and

$$\lambda_{\boldsymbol{d},m,k} = (1-\frac{k}{m})^{|\boldsymbol{d}|}.$$

Proof of lemma 1
Notice that $\mathbb{E}_A\left[H_A(r_h)\right]$ can be written into

$$\sum_{A\in\mathcal{A}_{m,k}} \Pr(A)\Big(-\sum_{j=0}^{3}\Pr_A(r_h=j)\log\Pr_A(r_h=j)\Big).$$

Also Notice that $-x\log x$ defined on $\mathbb{R}^+$ is a concave function. Hence, by using Jensen's inequality we can obtain an upper bound of $\mathbb{E}_A\left[H_A(r_h)\right]$, that is,

$$\begin{aligned}
&\mathbb{E}_A\left[H_A(r_h)\right]\\
=&\sum_{j=0}^{3}\sum_{A\in\mathcal{A}_{m,k}} -\Pr(A)\Pr_A(r_h=j)\log\Pr_A(r_h=j)\\
\leq&\sum_{j=0}^{3} -\Big(\sum_{A\in\mathcal{A}_{m,k}}\Pr(A)\Pr_A(r_h=j)\Big)\\
&\qquad\cdot\log\Big(\sum_{A\in\mathcal{A}_{m,k}}\Pr(A)\Pr_A(r_h=j)\Big)
\end{aligned}$$

Hence, to end our proof it suffices to calculate $\sum_{A\in\mathcal{A}_{m,k}}\Pr(A)\Pr_A(r_h=j)$. Since the sample space $\Omega(\boldsymbol{d})$ is fixed and known, due to the fact that $X \xrightarrow{A} \overline{AX} \xrightarrow{f} r$ forms a Markov chain we have

$$\Pr_A(r_h=j) = \sum_{X\in\Omega(\boldsymbol{d})}\Pr_A(X)\Pr_A(r_h=j|X).$$

Due to Assumptions 3 and 4, we have

$$\begin{aligned}
&\Pr_A(r_h = j|X) \\
&= \sum_{i=0}^{1} \Pr_A(r_h = j, (\overline{AX})_h = i|X) \\
&= \sum_{i=0}^{1} \Pr(r_h = j|(\overline{AX})_h = i) \Pr((\overline{AX})_h = i|X) \\
&= \sum_{i=0}^{1} f(j,i) \Pr((\overline{AX})_h = i|X).
\end{aligned} \tag{A. 1}$$

Since $\Pr_A(X) = \Pr(X)$ is assumed in Assumption 7, we further have

$$\begin{aligned}
&\sum_{A \in \mathcal{A}_{m,k}} \Pr(A) \Pr_A(r_h = j) \\
&= \sum_{A \in \mathcal{A}_{m,k}} \Pr(A) \sum_{X \in \Omega(\boldsymbol{d})} \Pr_A(X) \Pr_A(r_h = j|X) \\
&= \sum_{X \in \Omega(\boldsymbol{d})} \Pr(X) \sum_{A \in \mathcal{A}_{m,k}} \Pr(A) \Pr_A(r_h = j|X).
\end{aligned} \tag{A. 2}$$

Since for any fixed X with $|\boldsymbol{d}|$ positives, $\Pr((\overline{AX})_h = i|X) = 1$ if pooling design A contains at least one positive in the hth pool, otherwise 0. When the hth pool is considered, if X is fixed and A is random k-set design constructed by independent column generation of uniform random k-sets and hence A is chosen independently and uniformly from $\mathcal{A}_{m,k}$, then $\Pr((\overline{AX})_h = 0|X) = 1$ with probability $(1 - \frac{k}{m})^{|\boldsymbol{d}|}$. Therefore, we have

$$\begin{aligned}
&\sum_{A \in \mathcal{A}_{m,k}} \Pr(A) \Pr_A(r_h = j|X) \\
&= \sum_{i=0}^{1} f(j,i) \sum_{A \in \mathcal{A}_{m,k}} \Pr(A) \Pr((\overline{AX})_h = i|X) \\
&= \sum_{i=0}^{1} f(j,i) \left(1 - \frac{k}{m}\right)^{|\boldsymbol{d}|(1-i)} \left(1 - \left(1 - \frac{k}{m}\right)^{|\boldsymbol{d}|}\right)^{i}.
\end{aligned} \tag{A. 3}$$

Consequently, we finish the proof by combining the results of (A. 1), (A. 2) and (A. 3).

We introduce some notations for an exact computation of $\mathbb{E}_A[H_A(r_h|X)]$. Let $|\overline{AX}| = |\{h \in [m] : (\overline{AX})_h = 1\}|$. Given an X with positive pattern $\boldsymbol{d}$, denote by $P^{|\boldsymbol{d}|}_{X,m,k}(t)$ the probability that a k-set design A, uniformly chosen at random from $\mathcal{A}_{m,k}$, satisfies $|\overline{AX}| = t$. Recall that the columns of random k-set design are independent and uniformly chosen at random. This suggests that $P^{|\boldsymbol{d}|}_{X,m,k}(t) = P^{|\boldsymbol{d}|}_{X',m,k}(t)$ for any X and X' with $\boldsymbol{d}$. Thus, we can write $P^{|\boldsymbol{d}|}_{m,k}(t)$ for short. For any integers a and b, we define an indicate function of a and b as follows:

$$\delta(a,b) = \begin{cases} 0 & \text{if } a < b, \\ 1 & \text{if } a \geq b. \end{cases}$$

Lemma 2. *$P^{|\boldsymbol{d}|}_{m,k}(t)$ can be iteratively computed, for $t = k, \ldots, |\boldsymbol{d}|k$.*

$$P^{|\boldsymbol{d}|}_{m,k}(t) = \sum_{w=0}^{k} \mu_{m,k,t}(w) P^{|\boldsymbol{d}|-1}_{m,k}(t-k+w),$$

where

$$\mu_{m,k,t}(w) = \delta(t-k+w,k) \frac{\binom{t-k+w}{w}\binom{m-(t-k+w)}{k-w}}{\binom{m}{k}},$$

and in particular

$$P^{1}_{m,k}(k) = 1.$$

Proof of lemma 2

We assume $|\boldsymbol{d}|k \leq m$. For a fixed X_P with $|\boldsymbol{d}|$ positives and a fixed t, let $\mathcal{A}^{|\boldsymbol{d}|}_{X_P,m,k}(t)$ be the subset of all the k-set designs of size m satisfying $|\overline{AX_P}| = t$. Obviously, the value of t can range from k to $|\boldsymbol{d}|k$, and it is determined by the columns corresponding to the positive set P. Since the columns of random k-set designs are randomly and independently generated, it follows that $P^{|\boldsymbol{d}|}_{X_P,m,k}(t)$ only depends on $|\boldsymbol{d}|$, m and k. Therefore, any X_P with $|\boldsymbol{d}|$ positives can be used to derive $P^{|\boldsymbol{d}|}_{m,k}(t)$. Let $u^{|\boldsymbol{d}|}_{X_P,m,k}(t)$ be the number of ways of choosing columns corresponding to X_P such that any k-set design with those columns belongs to $\mathcal{A}^{|\boldsymbol{d}|}_{X_P,m,k}(t)$. Due to construction method of random k-set designs, we have

$$P^{|\boldsymbol{d}|}_{m,k}(t) = \frac{|\mathcal{A}^{|\boldsymbol{d}|}_{X_P,m,k}(t)|}{\binom{m}{k}^n} = \frac{u^{|\boldsymbol{d}|}_{X_P,m,k}(t)}{\binom{m}{k}^{\boldsymbol{d}}}.$$

Hence, to end our proof it suffices to give an iterative expression of $u^{|\boldsymbol{d}|}_{X_P,m,k}(t)$. Without loss of generality and for the sake of simplicity, we express $u^{|\boldsymbol{d}|}_{X_P,m,k}(t)$ by $u^{|\boldsymbol{d}-1|}_{X_{P'},m,k}(t')$s, for some $P' \subset P$ with $|\boldsymbol{d}| - 1$ positives.

Let $v_1, \ldots, v_{|\boldsymbol{d}|-1}$ be the columns of a k-set design A corresponding the positive set P' and $v_{|\boldsymbol{d}|}$ be the column corresponding to $P \setminus P'$. Notice that, for $w = 0, 1, \ldots, k$, if there are $t-k+w$ positive coordinates in the vector $\sum_{j=1}^{|\boldsymbol{d}|-1} v_j$, then we have $\binom{t-k+w}{w}\binom{m-(t-k+w)}{k-w}$ ways to choose $v_{|\boldsymbol{d}|}$ such that $\sum_{j=1}^{|\boldsymbol{d}|-1} v_j + v_{|\boldsymbol{d}|}$ has t positive coordinates. This implies that

$$u^{|\boldsymbol{d}|}_{X_P,m,k}(t) = \sum_{w=0}^{k} \delta(t-k+w,k) \binom{t-k+w}{w} \cdot \binom{m-(t-k+w)}{k-w} u^{|\boldsymbol{d}|-1}_{X_{P'},m,k}(t-k+w).$$

Dividing by $\binom{m}{k}^{|\boldsymbol{d}|}$ on both sides, the proof ends.

Using Lemma 2, we can obtain a closed expression of $\mathbb{E}_A[H_A(r|X)]$.

Lemma 3.

$$\mathbb{E}_A[H_A(r|X)] = \sum_{t=k}^{|\boldsymbol{d}|k} P^{|\boldsymbol{d}|}_{m,k}(t) \left[tH_f(1) + (m-t)H_f(0) \right],$$

where

$$H_f(i) = -\sum_{j=0}^{3} f(j,i)\log f(j,i),$$

for $i = 0, 1$.

Proof of lemma 3
By the definition of conditional entropy and Assumption 7, we have

$$\begin{aligned}
&\mathbb{E}_A\left[H_A(r|X)\right] \\
&= \sum_{A\in\mathcal{A}_{m,k}} \Pr(A) \sum_{X\in\Omega(\boldsymbol{d})} \Pr_A(X) H_A(r|X) \\
&= \sum_{X\in\Omega(\boldsymbol{d})} \Pr(X) \sum_{A\in\mathcal{A}_{m,k}} \Pr(A) \Bigg(\sum_{r\in\{0,1,2,3\}^m} -\Pr_A(r|X) \\
&\quad \cdot \log \Pr_A(r|X) \Bigg).
\end{aligned} \tag{C. 1}$$

Given any $X \in \Omega(\boldsymbol{d})$, we have the partition

$$\mathcal{A}_{m,k} = \bigcup_{t=k}^{|\boldsymbol{d}|k} \mathcal{A}^{|\boldsymbol{d}|}_{X,m,k}(t).$$

Moreover, recall that $\Pr_A(r|X) = \prod_{h=1}^{m} f(r_h, (\overline{AX})_h)$ due to Assumptions 3 and 4. Therefore, this implies when $X \in \Omega(\boldsymbol{d})$ is fixed, for any given $A \in \mathcal{A}^{|\boldsymbol{d}|}_{X,m,k}(t)$,

$$\begin{aligned}
H_A(r|X) &= \sum_{h=1}^{m} H(r_h|(\overline{AX})_h) \\
&= t\sum_{j=0}^{3} -f(j,1)\log f(j,1) \\
&\quad + (m-t)\sum_{j=0}^{3} -f(j,0)\log f(j,0) \\
&= tH_f(1) + (m-t)H_f(0).
\end{aligned} \tag{C. 2}$$

Using the result of (C. 2) and Lemma 3, it follows that

$$
\begin{aligned}
(C.1) &= \sum_{X\in\Omega(\boldsymbol{d})} \Pr(X) \sum_{t=k}^{|\boldsymbol{d}|k} \sum_{A\in\mathcal{A}^{|\boldsymbol{d}|}_{X,m,k}(t)} \Pr(A) \\
&\qquad \cdot \Big(tH_f(1) + (m-t)H_f(0) \Big) \\
&= \sum_{X\in\Omega(\boldsymbol{d})} \Pr(X) \sum_{t=k}^{|\boldsymbol{d}|k} P^{|\boldsymbol{d}|}_{m,k}(t) \Big(tH_f(1) \\
&\qquad + (m-t)H_f(0) \Big) \\
&= \sum_{t=k}^{|\boldsymbol{d}|k} P^{|\boldsymbol{d}|}_{m,k}(t) \Big(tH_f(1) + (m-t)H_f(0) \Big).
\end{aligned}
$$

Respectively substituting the results of Lemma 1 and Lemma 3 for $\mathbb{E}_A[H_A(r)]$ and $\mathbb{E}_A[H_A(r|X)]$ in (XI), we can easily compute an information-theoretic lower bound of the average probability of decoding error, that is,

$$
\mathbb{E}_A[p_{err}(A)] \geq 1 - \frac{I(\boldsymbol{d}, f, m, k) + 1}{\log|\Omega(\boldsymbol{d})|}, \tag{XII}
$$

where

$$
\begin{aligned}
I(\boldsymbol{d}, f, m, k) = &-m \sum_{j=0}^{3} f_{\boldsymbol{d},m,k}(j) \log f_{\boldsymbol{d},m,k}(j) \\
&- \sum_{t=k}^{|\boldsymbol{d}|k} P^{|\boldsymbol{d}|}_{m,k}(t) \big[tH_f(1) + (m-t)H_f(0) \big].
\end{aligned}
$$

For convenience, we denote by $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ the information-theoretic lower bound of (XII). By letting $\mathbb{E}_A[p_{err}(A)] = 0$, (XII) derives a necessary condition on m and k.

Theorem 4 (Necessity Theorem). $\mathbf{E}_A[p_{err}(A)]$ *vanishes to* 0 *only if* m *and* k *satisfy the condition*

$$
\log|\Omega(\boldsymbol{d})| \leq I(\boldsymbol{d}, f, m, k) + 1.
$$

In each subproblem where $\boldsymbol{d}$ is fixed and known, the Necessity Theorem implies that roughly $m = O(\log|\Omega(\boldsymbol{d})|)$ pools are least required for complete identifiability. This casts a light on the usefulness of knowing the overlap structure of clone maps. Prior knowledge of consecutive positives shrinks the size of the sample space of positives, which hence reduces the size of pooling design. This observation is consistent with the original motivation of Balding and Torney [7] and Colbourn [34].

Additionally, $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ can be used to predict the influence of f and k on the positive detectability of random k-set designs. To show this, we re-examine one of the screening problems studied by Knill et al. [101] and Uehara and Jimbo [174]. In the problem, the DNA library consists of 1298 clones without linear orders, among which there are four positives, and thus $|\Omega(\boldsymbol{d})| = \binom{1298}{4}$. Assigning two set of values to f, we fix f_1 and f_2 as shown in TABLE 4.1 and TABLE 4.2, respectively.

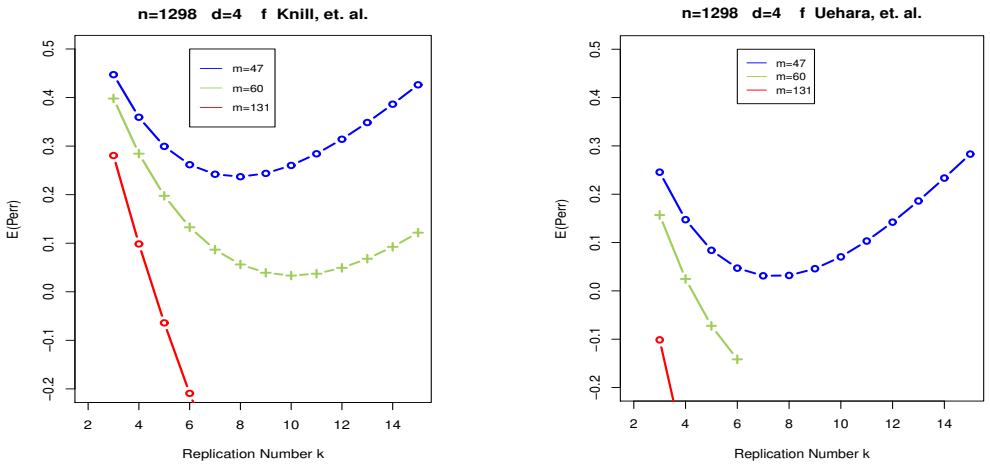
TABLE 4.1: f_1: Knill et al. [101]

$f(0,0) = 0.871$	$f(0,1) = 0.05$
$f(1,0) = 0.016$	$f(1,1) = 0.11$
$f(2,0) = 0.035$	$f(2,1) = 0.27$
$f(3,0) = 0.078$	$f(3,1) = 0.57$

TABLE 4.2: f_2: Uehara and Jimbo [174]

$f(0,0) = 0.856$	$f(0,1) = 0.02$
$f(1,0) = 0.126$	$f(1,1) = 0.155$
$f(2,0) = 0.016$	$f(2,1) = 0.288$
$f(3,0) = 0.002$	$f(3,1) = 0.537$

By fixing the value of m ($m = 47, 60, 131$), Fig. 4.1 shows the variation of $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ as the values of f and k vary.



(a) f1 (b) f2

FIGURE 4.1: Comparison of $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ subject to f_1 and f_2

Similar to the LDPC (Low-Density Parity-Check) codes (see Gallager [58] and Mackay and Neal [112]), random k-set designs demonstrates a degree of error-tolerant ability, which is closely related with the value of k. The U-shaped curve of k is in accordance with our expectation, since either too small or too large a value of k will weaken the positive detectability of random k-set designs. As Uehara and Jimbo [174] mentioned, we also observe that there is a big gap between the values of $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ when $k = 3$ and $k = 4$.

Comparing Fig. 4.1 (a) with Fig. 4.1 (b), the influence of f is also noticeable. Uehara and Jimbo [174] proposed f_2 for repairing the unnatural monotonicity of f_1 in which $f(1,0) < f(2,0) < f(3,0)$. They also simulated the positive detectability of BNPD and MCPD subject to f_2 and showed that the seemingly slight modification remarkably improves the performance of MCPD and BNPD. Interestingly, as is

shown in Fig. 4.1, the variation of $ILB(|\Omega(\boldsymbol{d})|, f, m, k)$ implies that f_1 is more difficult to cope with, which also coincides with their simulation results.

4.4.4 Random pooling designer

Given a positive pattern $\boldsymbol{d}$ it is often over complicated to obtain the exact $|\Omega(\boldsymbol{d})|$. Instead, we estimate a lower bound of $|\Omega(\boldsymbol{d})|$.

Particularly, we consider the positive pattern of the form $\boldsymbol{d} = (d_1, d_2, 0, 0)$ for some positive integer d_1 and nonnegative integer d_2 such that

$$\Omega(\boldsymbol{d}) = \{X \in \{0,1\}^n | D_1(X) = d_1, D_2(X) = d_2, D_3(X) = 0, D_4(X) = 0\}.$$

Letting $|\underline{\Omega}(\boldsymbol{d})| = \binom{d_1-d_2}{d_2} \prod_{i=1}^{d_1-d_2} (n - d_1 + d_2 - 3i + 4)$, we claim that $|\underline{\Omega}(\boldsymbol{d})|$ is a lower bound of $|\Omega(\boldsymbol{d})|$. This can be seen as follows. To begin with, it is obvious that there are $d_1 - 2d_2$ maximum consecutive subset with a single positive and d_2 maximum consecutive subset with 2 consecutive positives, and therefore we can treat each single positive as a red ball and each pair of consecutive positives as a blue ball. To calculate $|\Omega(\boldsymbol{d})|$, it is equivalent to counting how many ways these $d_1 - d_2$ balls with distinguishable colors can be put into $n - (d_1 - d_2) + 1$ urns such that any two ball have at least an urn between. This constraint implies that each ball will occupy at most 3 urns. Thus, to place the ith ball, at least $(n - (d_1 - d_2) + 1 - 3(i-1))$ urns are left to choose from, for $i = 1, 2, \ldots, d_1 - d_2$. Noticing that $\binom{d_1-d_2}{d_2}$ is the number of ways to color these balls without repetition, the desired lower bound is obtained.

With $|\underline{\Omega}(\boldsymbol{d})|$ and f, we give the algorithm called RPD (Random Pooling Designer) for exhaustive search of the least value of m and some value of k that satisfy the Necessity Theorem. $m_{\boldsymbol{d},f}$ and $k_{\boldsymbol{d},f}$ denote by the least value of m and by the value of k with respect to $|\underline{\Omega}(\boldsymbol{d})|$ and f, respectively. With the values of $m_{\boldsymbol{d},f}$ and $k_{\boldsymbol{d},f}$, various strategies can be used to choose the proper values of m and k for $\hat{\theta}$ and f.

Random Pooling Designer

Input: $|\underline{\Omega}(\boldsymbol{d})|$ and f
Output: $m_{\boldsymbol{d},f}$ and $k_{\boldsymbol{d},f}$

```
m_{d,f} ← 1
k_{d,f} ← 1
temp ← ILB(|Ω(d)|, f, m_{d,f}, k_{d,f})
while temp > 0 do
    m_{d,f} ← m_{d,f} + 1
    k_{d,f} ← 1
    while temp > ILB(|Ω(d)|, f, m_{d,f}, k_{d,f}) and k ≤ m do
        temp ← ILB(|Ω(d)|, f, m_{d,f}, k_{d,f})
        k_{d,f} ← k_{d,f} + 1
    end while
end while
```

4.5 Simulation

This section shows the performance of MMCPD with random k-set designs chosen by RPD.

4.5.1 Simulation Method

The simulations are performed as follows.

1. Letting $n = 1298$ be the size of $\mathcal{C}$, the prior knowledge of consecutive positives is set to $d_1 = 6$, $d_2 = 2.4$, $d_3 = 0.01$ and $d_4 = 0.001$.
2. The likelihoods of experimental error f are set to f_1 given in TABLE 4.1.
3. Find an approximate prior probability distribution $\mathbb{P}_\theta$ that incorporates the prior knowledge by solving (V).
4. Estimate the distribution of positive patterns of $\mathbb{P}_\theta$.
5. Choose proper positive patterns, and compute $|\underline{\Omega}(\boldsymbol{d})|$, $m_{\boldsymbol{d},f}$ and $k_{\boldsymbol{d},f}$, for each chosen $\boldsymbol{d}$.
6. Choose $m_{\theta,f}$ and $k_{\theta,f}$ such that $m_{\theta,f}$ is the least value with $ILB(|\underline{\Omega}(\boldsymbol{d})|, f_1, m_{\theta,f}, k_{\theta,f}) < 0$, for all the chosen positive patterns.
7. The positive set P is given in two ways.
   - Fix $\{240, 241, 890, 891, 1001, 1002\}$ as the positive set.
   - Randomly choose a positive set approximately according to $\mathbb{P}_\theta$.
8. Generate a random k-set design A with some proper values of k and m.
9. Compute the number of positives in each pool.
10. Based on 11), determine the pooling results r randomly according to f_1.
11. Implement MMCPD to decode the corrupted pooling results r. The posterior probability of being a positive is estimated for each clone.
12. Clones are sorted in a decreasing order according to their posterior probabilities.

4.5.2 Preprocess of pooling procdedure

Solving (VII) with Descombes et al. [41]'s method, we find that $\hat{\theta} = (6.96, -7.65, 4.955, 2.01)$ is a suitable approximation for the the desired prior distribution. However, the estimation based on MCMC simulation is costly and hence a good initial θ will accelerate the convergence of Descombes et al [41]'s method. In implementation, to find a proper initial θ involves some guesswork and it can be done in a bisection way by using trial and failure method.

Next, we take 10000 samples for estimating the distribution of positive patterns of $\mathbb{P}_{\hat{\theta}}$ and list the positive patterns with sample mean above 0.0025 in TABLE 4.3. It also lists the corresponding $m_{\boldsymbol{d},f}$s and $k_{\boldsymbol{d},f}$s with respect to the positive patterns of our consideration and f_1. Particularly, $\boldsymbol{d} = (0, 0, 0, 0)$ is trivial and we remove it from our consideration. The total positive patterns of our consideration roughly account for 95% of the positive patterns of the samples.

TABLE 4.3: Estimated Distribution of $\boldsymbol{d}$ with respect to $\mathbb{P}_{\hat{\theta}}$ and Values of $m_{\boldsymbol{d},f}$ and $k_{\boldsymbol{d},f}$ with respect to f_1

$\boldsymbol{d}$	**Sample Mean**	$m_{\boldsymbol{d},f}$	$k_{\boldsymbol{d},f}$
(0, 0, 0, 0)	0.0286	-	-
(1, 0, 0, 0)	0.0307	17	7
(2, 0, 0, 0)	0.0192	33	9
(2, 1, 0, 0)	0.065	17	4
(3, 0, 0, 0)	0.0075	48	10
(3, 1, 0, 0)	0.0798	33	6
(4, 1, 0, 0)	0.0470	48	8
(4, 2, 0, 0)	0.0791	31	5
(5, 1, 0, 0)	0.0191	63	8
(5, 2, 0, 0)	0.0925	46	6
(6, 1, 0, 0)	0.006	96	9
(6, 2, 0, 0)	0.0617	61	7
(6, 3, 0, 0)	**0.0618**	**44**	**5**
(7, 2, 0, 0)	0.0232	75	7
(7, 3, 0, 0)	0.0717	58	6
(8, 2, 0, 0)	0.0071	88	8
(8, 3, 0, 0)	0.0459	72	6
(8, 4, 0, 0)	0.0385	55	4
(9, 3, 0, 0)	0.0189	85	7
(9, 4, 0, 0)	0.0446	69	5
(10, 3, 0, 0)	0.0048	98	7
(10, 4, 0, 0)	0.0272	82	6
(10, 5, 0, 0)	0.0181	65	4
(11, 4, 0, 0)	0.0118	95	6
(11, 5, 0, 0)	0.0217	78	5
(12, 4, 0, 0)	**0.0041**	**108**	**6**
(12, 5, 0, 0)	0.0128	91	5
(12, 6, 0, 0)	0.0059	74	4
(13, 5, 0, 0)	0.0040	104	6
(13, 6, 0, 0)	0.0087	86	5
(14, 6, 0, 0)	0.0048	99	5
(14, 7, 0, 0)	0.0028	82	4
(15, 7, 0, 0)	0.0038	95	4
Others	0.0216	-	-

4.5.3 Simulation 1: fixed positive set

Of all the chosen positive patterns, $\boldsymbol{d} = (6, 3, 0, 0)$ attracts our attention because it has relatively high frequency, contains relatively large number of positives, but requires few pools with small value of k. By fixing the positive set $P = \{240, 241, 890, 891, 1001, 1002\}$, 500 simulations are implemented to show MMCPD's positive detectability with random 5-set designs of size 44.

In each simulation, beginning with the state drawn from i.i.d Bernoulli trials with parameter $q = \frac{6}{1298}$, the warmup period includes 5000 steps. After the warmup period, another 15000 steps are run, and the states obtained in every 3 steps are used as samples to estimate the posterior probability of being positive for each clone, that is, approximately the proportion of the obtained states including the given clone. We subsample the Markov chain in hope of weakening potential autocorrelations among the samples. Denote by CP the set of candidate positives. MMCPD outputs CP which consists of the six clones with the highest mean posterior probabilities. TABLE 4.4 shows the number of times among 500 simulations that $|P \cap CP|$ positives can be identified by using MMCPD.

TABLE 4.4: Positive Detectability of MMCPD for Fixed Positive Set

$\lvert P \cap CP \rvert$	Times
6	98
5	39
4	149
3	37
2	112
1	23
0	42

During the simulations, we observed that MMCPD detects the underlying true positives in a pairwise way, which can also be seen from the TABLE 4.4. Successfully decoding one underlying true positive will remarkably improve the performance of detecting the consecutive one. Besides, we can also conclude that the lower bound $ILB(|\underline{\Omega}(\boldsymbol{d})|, f_1, m_{\boldsymbol{d},f}, k_{\boldsymbol{d},f})$, though underestimates the underlying true minimal number of pools required, is nontrivial and useful.

4.5.4 Simulation 2: random positive set

By using the TABLE 4.3, we can verify that $m_{\hat{\theta},f} = 108$ is the least value with

$$ILB(|\underline{\Omega}(\boldsymbol{d})|, f_1, m_{\hat{\theta},f}, k_{\hat{\theta},f}) < 0$$

, for all the chosen positive patterns, where $k_{\hat{\theta},f} = 6$.

1000 simulations are implemented to show MMCPD's positive detectability. In each simulation, a nonempty positive set is first randomly generated approximately according to $\mathbb{P}_{\hat{\theta}}$ and then a random 6-set design of size 108 is independently constructed. The decoding procedure of random positive set is the same with that of the fixed positive set except for the choice of the set of candidate positives. To evaluate the detectability of MMCPD for randomly generated positive set P, we introduce the

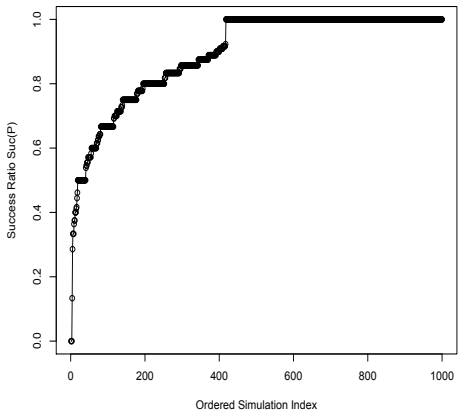


FIGURE 4.2: Detectability of MMCPD for Random Positive set

success ratio $Suc(P) \triangleq \frac{|P \cap CP|}{|P|}$, where CP is the candidate positive set of $|P|$ clones which have the highest mean posterior probabilities. Obviously, the complete identifiability occurs when $Suc(P) = 1$. This measurement is nevertheless strict. Fig. 4.2 shows the simulation results.

In the simulations, MMCPD showed a stable and reliable performance with random designs. Its novel efficiency, as expected, is due to the prior knowledge of consecutive positives and also due to the proper choice of values of m and k suggested by RPD. Actually, there are 593 instances where MMCPD completely identified all the randomly generated positives, which to some extend achieve our expectation and also validates our theoretical analysis.

However, if too many positives or/and errors occur, MMCPD may not perform as well as expected. Denote by E_{FN} and E_{FP} the number of false negative errors and that of false positive errors, respectively. TABLE 4.5 lists all the instances with the success ratio less than 0.4.

TABLE 4.5: Instances Where MMCPD Performed Badly

Index	$\boldsymbol{d}$	E_{FN}	E_{FP}	$Suc(P)$
1	(1, 0, 0, 0)	0	21	0
2	(2, 0, 0, 0)	4	17	0
3	(1, 0, 0, 0)	0	14	0
4	(15, 6, 0, 0)	4	7	0.133333
5	(21, 10, 0, 0)	7	6	0.285714
6	(3, 0, 0, 0)	1	13	0.333333
7	(3, 0, 0, 0)	2	18	0.333333
8	(12, 5, 0, 0)	1	13	0.333333
9	(11, 4, 0, 0)	6	7	0.363636
10	(16, 7, 0, 0)	4	4	0.375
11	(8, 3, 0, 0)	3	17	0.375

Typically, these instances are hard. If too many positives and/or experimental

errors occur, or if the positive pattern of consecutive positives deviates from our prior knowledge too far, or a concurrence of some of these difficulties, will weaken the positive detectability of MMCPD. To overcome this, we may either enlarge the set of candidate positives or use a pooling design with more pools, or both.

4.6 Conclusion

This Chapter studied the problem of screening clone maps in the presence of experimental errors. We propose a probabilistic ranking algorithm for identifying consecutive positives. The rationale behind the method can be summarized as follows. First, we expressed the prior information regarding consecutive structure in a combinatorial way and then transform the prior information into a prior distribution by applying maximum entropy principle. Second, the posterior probability that a clone is positive is estimated and ranked by Markov Chain Monte Carlo method. The MCMC method incorporated the structure of potential positive clones and gave a robust performance even when experimental uncertainty presented and random pooling designs were used. Third, to maximize the detectability of the ranking algorithm with random k-set designs, we give an information-theoretic framework to choose proper size and replication number of random k-set design. This framework supports the conjecture of Balding and Torney [7] that prior information regarding the structure of a DNA library will help improve the efficiency of experiment designs in terms of the number of pools least necessarily needed in order to identify all the positive clones.

Chapter 5

Conclusion

As a methodology of statistical inference, Bayesian approach plays an important role in constructing useful machine learning algorithms that can implement consistent and plausible reasoning, including both modeling for prediction and theorizing for explanation. This dissertation discussed ranking methods and their machine learning algorithms from a cognitive and Bayesian perspective. These methods are applied to consumer data analysis and molecular bioinformatic problems. In each application, we showed (1) how domain-specific knowledge can be transformed and represented for learning purposes, (2) how to construct an effective ranking algorithm that take advantage of the representation of the domain-specific knowledge, and (3) how and why the knowledge representation would improve the performance of the learning process of the ranking algorithm.

5.1 Preference learning machine

In Chapter 2, we discussed the problem of ranking smartphone apps according to their quality assessment from a user's perspective based on large-scale users' behavioral information which can automatically be collected without the requirement of users' active participation. With a commercial dataset, two ranking methods were proposed, one for extracting superior apps and the other for ordering the extracted apps.

In order to extract superior apps, a cognitive process of pointwise comparisons was first discussed and mathematically formulated by introducing the notions: standard of comparison and random user; then, a score function with desirable properties were constructed and used to assign a value of quality assessment measurement to each of the apps to be ranked. In order to arrange the extracted apps into a linear order, a cognitive process of comparisons from an end-user's perspective was first discussed and mathematically formulated by assigning an ordinal structure to the set of usage patterns, and then based on the results of pairwise comparisons between the extracted apps, the aggregate preference into which users' individual preferences merged can be obtained and used as the ranking result.

Because the dataset we used provided neither target observations nor class labels, the ranking algorithm we proposed to learn users' preferences from the data is essentially unsupervised. Data experiment showed that the ranking results given by the pairwise-comparison-based method are different from the ranking result given by the the pairwise-comparison-based method; moreover, they can work together to provide workable ranking results that might be useful for mining knowledge from users' behavioral information.

5.2 Knowledge discovery machine

Chapter 3 discussed a method for discovering the knowledge of the rankings of items by constructing a learning machine that makes prediction and explanation simultaneously. The cognitive ranking model proposed in Section 2.3 of Chapter 2 was examined but from a new perspective. The ranking model was formulated with a class of desirable properties, which produced a class of hypotheses with prior knowledge incorporated. With this model, the knowledge discovery problem was formulated as a single biconvex optimization problem which has a relationship with SVM(Support Vector Machines). To facilitate the process of knowledge discovery, we developed a two-stage learning algorithm. In the first stage, we extracted as much information as possible from the observed data, while in the second stage, the extracted information was further summarized into knowledge. Particularly, the learning algorithm is essential Bayesian, combined with Frequentist learning theory. Frequentist approach is adopted to ease the difficulty that the likelihood function cannot be explicitly formulated.

Given a small number of input data as well as a moderate amount of prior information about the input and the systems parameters, simulation results showed that the two-stage learning algorithm is able to discover the unknown patterns. Moreover, the explanation improved the accuracy of prediction. The predictive model seemed also useful for finding out the best explanations in agreement with the prior and the output.

5.3 Group testing machine

Chapter 4 proposed a Bayesian framework for pooling experiments when consecutive positives present. The prior knowledge of the consecutiveness of positives was transformed into a mathematical prior by using Maximum Entropy Principle. A ranking algorithm was constructed for detecting consecutive positives. The learning process of the ranking algorithm is based on MCMC method, using both prior knowledge and error-prone pooling results generated by random pooling designs. Given prior knowledge of consecutive positives and experimental errors, the necessity theorem of random k-set designs was developed in order to choose the values of the key parameters of pooling design that might most reinforce the performance of the decoding algorithm. Moreover, with the necessity theorem, we gave a theoretical explanation why the prior knowledge of consecutive positives might reduce the number of pools to recover the positives.

Chapter 6

Future Work

6.1 Preference learning machine

When target rankings of smartphone apps are available, it would be interesting to compare the resulting rankings with the target rankings. Moreover, we may be interested in the problem, namely: given a target ranking, which ordinal structure of usage patterns has led to the target ranking. Notice that the problem of linear ordering is computationally hard, in this case, how to solve this inverse problem of preference learning?

Automatically judging the quality of rankings based on observable user behavior holds promising for making ranking evaluation faster, cheaper, and more user centered. In real applications, statistical testing of significance may not be available, and even so, it seems less possible to fully understand the relationship between observable user behavior and ranking quality. Instead, we should seek for practical testing procedures in which significance of practice could be tested. This research direction, involving open information processing and understanding, seems interesting and promising.

6.2 Knowledge discovery machine

A hierarchical structure of choice models would be more desired, with each associated a class of system parameters. In this way, automatic knowledge discovery can be implemented in a larger space of hypotheses with more freedom of parameters. Then, how to arrange the structure of axioms would be a challenging task. Moreover, it seems that there should be a trade-off between prediction and explanation, just as in the cases of bias-variance trade-off or approximation-capacity trade off. If so, how to balance the explaining ability and predicting ability of a learning?

When SRM principle is applied, only the linearly separable case is considered and thus the application would be very limited. Extension to nonlinear cases would involve theorizing problems, computation problems and interpretability problems. Then, how to overcome these difficulties?

The two-stage learning algorithm is solved by half-random-half-deterministic method in the fist algorithmic stage. Although it may be favored by a parallel implementation, a pure deterministic algorithm is desired, to evaluate the performance of the half-random-half-deterministic one. Moreover, it is also interesting to know the relationship between the two-stage learning algorithm and EM algorithms.

6.3 Group testing machine

Fano's inequality plays the key role in formulating the necessity theorem. It is believed that Fano's inequality is found sharp in many cases. Then, is it possible to improve the necessity theorem of random k-sets designs by improving the upper bound of $\mathbb{E}_A\left[H_A(r)\right]$?

EM algorithms are believed faster than MCMC methods in some applications. Is it the case for probabilistic group testing? Can other approximate inference methods be applied to probabilistic group testing?

Motivated by the development of graph-constrained group testing, we may be interested to extend probabilistic group testing algorithms to the graph-constrained cases.

Publication

Journal

Song Luo. "A method for discovering the knowledge of item rank from consumer reviews". In: JSIAM Letters (2018).

***Remark 1** The content of the journal paper is extended and contained in Chapter 3 of this dissertation.

International Conference

Song Luo, Wenbo Ma, and Maiko Shigeno. "Ranking Smartphone Apps Based on Users' Behavior Records". In: Maeno T., Sawatani Y., Hara T. (eds) Serviceology for Designing the Future. Springer, Tokyo (2016), pp. 345-358.

***Remark 2** The content of the conference paper is extended and contained in Chapter 2.

Representation

Megumi Fujimoto, Song Luo, Mutiara Sari Dian, Maiko Shigeno and Miki Zago, "Usage Pattern Analysis for Android Applications: Classification and Measurement", Oral Representation In: The 2th Domestic Conference on Serviceology, Hakodate, Hokkaidou, Japan (2014).

***Remark 3** The content of the oral representation is extended and contained in Chapter 2.

Song Luo. "A nonadaptive probabilistic group testing algorithm for detecting consecutive positives of linear DNA library ". Oral Representation In: The 21st International Symposium on Mathematical Programming, Berlin, Germany (2012).

***Remark 4** The content of the oral representation is extended and contained in Sections 2 and 3 of Chapter 4.

Song Luo and Maiko Shigeno. "On the random k-set designs under experimental uncertainty". Poster Representation In: The Japan Society for Industrial and Applied Mathematics, Fukuoka, Japan (2013).

***Remark 5** The content of the poster representation is extended and contained in Sections 4 and 5 of Chapter 4.

Bibliography

[1] URL: http://www.goodreads.com/list/show/32685.

[2] Shivani Agarwal, Deepak Dugar, and Shiladitya Sengupta. "Ranking chemical structures for drug discovery: a new machine learning approach". In: *Journal of chemical information and modeling* 50.5 (2010), pp. 716–731.

[3] Naseer Ahmad, Muhammad Waqas Boota, and Abdul Hye Masoom. "Smart phone application evaluation with usability testing approach". In: *Journal of Software Engineering and Applications* 7.12 (2014), p. 1045.

[4] Emin Aleskerov, Bernd Freisleben, and Bharat Rao. "Cardwatch: A neural network based database mining system for credit card fraud detection". In: *Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997*. IEEE. 1997, pp. 220–226.

[5] Nikolay Archak, Anindya Ghose, and Panagiotis G. Ipeirotis. "Deriving the pricing power of product features by mining consumer reviews". In: *Management science* 57.8 (2011), pp. 1485–1509.

[6] George K. Atia and Venkatesh Saligrama. "Boolean Compressed Sensing and Noisy Group Testing". In: *IEEE Transactions on Information Theory* 58.3 (2012), pp. 1880–1901.

[7] David J. Balding and David C. Torney. "The design of pooling experiments for screening a clone map". In: *Fungal Genetics and Biology* 21.3 (1997), pp. 302–307.

[8] David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[9] Emmanuel Barillot, Bruno Lacroix, and Daniel Cohen. "Theoretical analysis of library screening using a N-dimensional pooling strategy". In: *Nucleic acids research* 19.22 (1991), pp. 6241–6247.

[10] Raquel Benbunan-Fich and Alberto Benbunan. "Understanding user behavior with new mobile applications". In: *The Journal of Strategic Information Systems* 16.4 (2007), pp. 393–412.

[11] Toby Berger, James W. Mandell, and P. Subrahmanya. "Maximally Efficient Two-Stage Screening". In: *Biometrics* 56.3 (2000), pp. 833–840.

[12] Toby Berger, Nader Mehravari, Don Towsley, and Jack Wolf. "Random multiple-access communication and group testing". In: *IEEE Transactions on Communications* 32.7 (1984), pp. 769–779.

[13] José M. Bernardo and Adrian F.M. Smith. *Bayesian theory*. Wiley, 2006.

[14] Nigel Bevan. "Measuring usability as quality of use". In: *Software Quality Journal* 4.2 (1995), pp. 115–130.

[15] Christopher M. Bishop. *Pattern Recognition and machine learning*. Springer-Verlag New York, 2016.

[16] David C. Blair and Melvin E. Maron. "An evaluation of retrieval effectiveness for a full-text document-retrieval system". In: *Communications of the ACM* 28.3 (1985), pp. 289–299.

[17] Ismael Rodrigo Bleyer and Antonio Leitão. *Novel regularization methods for ill-posed problems in Hilbert and Banach spaces*. Publicações matemáticas do IMPA, 2015.

[18] William M. Bolstad. *Understanding computational Bayesian statistics*. Vol. 644. John Wiley & Sons, 2010.

[19] William M. Bolstad and James M. Curran. *Introduction to Bayesian statistics*. John Wiley & Sons, 2016.

[20] John Bredican and Debbie Vigar-Ellis. "Smartphone Applications-Idea sourcing and app development: Implications for firms". In: *South African Journal of Economic and Management Sciences* 17.3 (2014), pp. 232–248.

[21] William J. Bruno, Emanuel Knill, David J. Balding, David C. Bruce, Norman A. Doggett, Wesley W. Sawhill, Raymond L. Stallings, Craig C. Whittaker, and David C. Torney. "Efficient pooling designs for library screening." In: *Genomics* 26.1 (1995), p. 21.

[22] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. "Learning to rank using gradient descent". In: *Proceedings of the 22nd international conference on Machine learning*. ACM. 2005, pp. 89–96.

[23] C. Richard Cassady, Lisa M. Maillart, and Sinan Salman. "Ranking sports teams: A customizable quadratic assignment approach". In: *Interfaces* 35.6 (2005), pp. 497–510.

[24] Christophe Chamley. *Rational herds: Economic models of social learning*. Cambridge University Press, 2004.

[25] Chun Lam Chan, Pak Hou Che, Sidharth Jaggi, and Venkatesh Saligrama. "Non-adaptive probabilistic group testing with noisy measurements: Near-optimal bounds with efficient algorithms". In: *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*. IEEE. 2011, pp. 1832–1839.

[26] Chun Lam Chan, Sidharth Jaggi, Venkatesh Saligrama, and Samar Agnihotri. "Non-adaptive group testing: Explicit bounds and novel algorithms". In: *IEEE Transactions on Information Theory* 60.5 (2014), pp. 3019–3035.

[27] Esmita Charani, Enrique Castro-Sánchez, Luke SP Moore, and Alison Holmes. "Do smartphone applications in healthcare require a governance and legal framework? It depends on the application!" In: *BMC medicine* 12.1 (2014), p. 29.

[28] Juliana Chen, Janet E. Cade, and Margaret Allman-Farinelli. "The most popular smartphone apps for weight loss: a quality assessment." In: *JMIR Mhealth Uhealth* 3.4 (2015).

[29] Yongxi Cheng and Ding-Zhu Du. "New constructions of one-and two-stage pooling designs". In: *Journal of Computational Biology* 15.2 (2008), pp. 195–205.

[30] Sunjin Choe. "An exploration of the qualities and features of art apps for art therapy". In: *The Arts in psychotherapy* 41.2 (2014), pp. 145–154.

[31] Wei Chu and Zoubin Ghahramani. "Gaussian processes for ordinal regression". In: *Journal of machine learning research* 6.Jul (2005), pp. 1019–1041.

[32] Wei Chu and S. Sathiya Keerthi. "New approaches to support vector ordinal regression". In: *Proceedings of the 22nd international conference on Machine learning*. ACM. 2005, pp. 145–152.

[33] William W. Cohen, Robert E. Schapire, and Yoram Singer. "Learning to order things". In: *Advances in Neural Information Processing Systems*. 1998, pp. 451–457.

[34] Charles J. Colbourn. "Group testing for consecutive positives". In: *Annals of Combinatorics* 3.1 (1999), pp. 37–41.

[35] Enrique Costa-Montenegro, Ana Belén Barragáns-Martínez, and Marta Rey-López. "Which App? A recommender system of applications in markets: Implementation of the service for monitoring users' interaction". In: *Expert systems with applications* 39.10 (2012), pp. 9367–9375.

[36] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[37] Richard T. Cox. "Probability, frequency and reasonable expectation". In: *American journal of physics* 14.1 (1946), pp. 1–13.

[38] Giulio D'Agostini. *Bayesian reasoning in data analysis: A critical introduction*. World Scientific, 2003.

[39] Marco De Gemmis, Leo Iaquinta, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. "Preference learning in recommender systems". In: *Preference Learning* 41 (2009).

[40] Krzysztof Dembczyński, Wojciech Kotłowski, Roman Słowiński, and Marcin Szeląg. "Learning of rule ensembles for multiple attribute ranking problems". In: *Preference Learning*. Springer, 2010, pp. 217–247.

[41] Xavier Descombes, Robin Morris, Josiane Zerubia, and Marc Berthod. "Maximum likelihood estimation of Markov random field parameters using markov chain Monte Carlo algorithms". In: *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer. 1997, pp. 133–148.

[42] Nico d'Heureuse, Felipe Huici, Mayutan Arumaithurai, Mohamed Ahmed, Konstantina Papagiannaki, and Saverio Niccolini. "What's app?: a wide-scale measurement study of smart phone markets". In: *ACM SIGMOBILE Mobile Computing and Communications Review* 16.2 (2012), pp. 16–27.

[43] Pierre Dillenbourg. *Collaborative learning: Cognitive and computational approaches. advances in learning and instruction series*. ERIC, 1999.

[44] Trinh-Minh-Tri Do and Daniel Gatica-Perez. "By their apps you shall understand them: mining large-scale patterns of mobile phone usage". In: *Proceedings of the 9th international conference on mobile and ubiquitous multimedia*. ACM. 2010, p. 27.

[45] Robert Dorfman. "The detection of defective members of large populations". In: *The Annals of Mathematical Statistics* 14.4 (1943), pp. 436–440.

[46] A D'yachkov, Frank Hwang, Antony Macula, Pavel Vilenkin, and Chih-wen Weng. "A construction of pooling designs with some happy surprises". In: *Journal of Computational Biology* 12.8 (2005), pp. 1129–1136.

[47] Arkadii G. D'yachkov, Vyacheslav Rykov, and A. M. Rashad. "Superimposed distance codes". In: *Problems of Control and Information Theory* 18.4 (1989), pp. 237–250.

[48] Nathan Eagle, Alex Sandy Pentland, and David Lazer. "Mobile phone data for inferring social network structure". In: *Social computing, behavioral modeling, and prediction*. Springer, 2008, pp. 79–88.

[49] Martin Farach-Colton, Sukeshwar Kannan, Emanuel Knill, and Sri Murugarasan Muthukrishnan. "Group testing problems with sequences in experimental molecular biology". In: *Compression and Complexity of Sequences 1997. Proceedings*. IEEE. 1997, pp. 357–367.

[50] Kraig Finstad. "The usability metric for user experience". In: *Interacting with Computers* 22.5 (2010), pp. 323–327.

[51] Peter Flach. *Machine learning: the art and science of algorithms that make sense of data*. Cambridge University Press, 2012.

[52] Josua Flath, Christian Lütkemeyer, Samet Mercan, and Bui Le Dzung. "The Facebook acquisition of Instagram: A Case Study". In: (2013).

[53] Brendan J. Frey. *Graphical models for machine learning and digital communication*. MIT press, 1998.

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

[55] Johannes Fürnkranz and Eyke Hüllermeier. "Preference learning". In: *Encyclopedia of Machine Learning*. Springer, 2011, pp. 789–795.

[56] David G. Taylor and Michael Levin. "Predicting mobile app usage for purchasing and information-sharing". In: *International Journal of Retail & Distribution Management* 42.8 (2014), pp. 759–774.

[57] Mohamed M. Gaber. *Scientific data mining and knowledge discovery*. Springer, 2009.

[58] Robert Gallager. "Low-density parity-check codes". In: *IRE Transactions on information theory* 8.1 (1962), pp. 21–28.

[59] Dani Gamerman and Hedibert F. Lopes. *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*. Chapman and Hall/CRC, 2006.

[60] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1983.

[61] Damianos Gavalas, Paolo Bellavista, Jiannong Cao, and Valérie Issarny. *Mobile applications: Status and trends*. 2011.

[62] Gennian Ge, Ying Miao, and Xiande Zhang. "On block sequences of Steiner quadruple systems with error correcting consecutive unions". In: *SIAM journal on discrete mathematics* 23.2 (2009), pp. 940–958.

[63] Stuart Geman and Donald Geman. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images". In: *Readings in Computer Vision*. Elsevier, 1987, pp. 564–584.

[64] Charles J. Geyer and Elizabeth A. Thompson. "Constrained Monte Carlo maximum likelihood for dependent data". In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1992), pp. 657–699.

[65] Anna C. Gilbert, Mark A. Iwen, and Martin J. Strauss. "Group testing and sparse signal recovery". In: *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*. IEEE. 2008, pp. 1059–1063.

[66] Jeremy Goecks and Jude Shavlik. "Learning users' interests by unobtrusively observing their normal behavior". In: *Proceedings of the 5th international conference on Intelligent user interfaces*. ACM. 2000, pp. 129–132.

[67] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. "Deep learning (adaptive computation and machine learning series)". In: *Adaptive Computation and Machine Learning series* (2016), p. 800.

[68] Jochen Gorski, Frank Pfeuffer, and Kathrin Klamroth. "Biconvex sets and optimization with biconvex functions: a survey and extensions". In: *Mathematical methods of operations research* 66.3 (2007), pp. 373–407.

[69] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. "A cutting plane algorithm for the linear ordering problem". In: *Operations research* 32.6 (1984), pp. 1195–1220.

[70] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. "Facets of the linear ordering polytope". In: *Mathematical Programming* 33.1 (1985), pp. 43–60.

[71] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. "On the acyclic subgraph polytope". In: *Mathematical Programming* 33.1 (1985), pp. 28–42.

[72] Arjun K. Gupta and Saralees Nadarajah. *Handbook of beta distribution and its applications*. CRC press, 2004.

[73] Peter Haddawy, Vu Ha, Angelo Restificar, Benjamin Geisler, and John Miyamoto. "Preference elicitation via theory refinement". In: *Journal of Machine Learning Research* 4.Jul (2003), pp. 317–337.

[74] Kelli Hale, Sandra Capra, and Judith Bauer. "A framework to assist health professionals in recommending high-quality apps for supporting chronic disease self-management: illustrative assessment of type 2 diabetes apps". In: *JMIR mHealth and uHealth* 3.3 (2015).

[75] Lutz H. Hamel. *Knowledge discovery with support vector machines*. Vol. 3. John Wiley & Sons, 2011.

[76] Sariel Har-Peled, Dan Roth, and Dav Zimak. "Constraint classification: A new approach to multiclass classification and ranking". In: *In Advances in Neural Information Processing Systems 15*. Citeseer. 2002.

[77] Sariel Har-Peled, Dan Roth, and Dav Zimak. "Constraint classification for multiclass classification and ranking". In: *Advances in neural information processing systems*. 2003, pp. 809–816.

[78] Rachel Harrison, Derek Flood, and David Duce. "Usability of mobile applications: literature review and rationale for a new usability model". In: *Journal of Interaction Science* 1.1 (2013), p. 1.

[79] Nicholas J.A. Harvey, Mihai Patrascu, Yonggang Wen, Sergey Yekhanin, and Vincent W.S. Chan. "Non-adaptive fault diagnosis for all-optical networks via combinatorial group testing on graphs". In: *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. IEEE. 2007, pp. 697–705.

[80] Robert H. Hayes and Steven C. Wheelwright. "Link manufacturing process and product life cycles". In: *Harvard business review* 57.1 (1979), pp. 133–140.

[81] Ralf Herbrich, Thore Graepel, Peter Bollmann-Sdorra, and Klaus Obermayer. "Learning preference relations for information retrieval". In: *ICML-98 Workshop: text categorization and machine learning*. 1998, pp. 80–84.

[82] Adrian Holzer and Jan Ondrus. "Mobile application market: A developer's perspective". In: *Telematics and informatics* 28.1 (2011), pp. 22–31.

[83] Kit Huckvale, Mate Car, Cecily Morrison, and Josip Car. "Apps for asthma self-management: a systematic assessment of content and tools". In: *BMC medicine* 10.1 (2012), p. 144.

[84] Frank K. Hwang and Ding-zhu Du. *Combinatorial group testing and its applications*. Vol. 12. World Scientific, 2000.

[85] Frank K. Hwang and Ding-zhu Du. *Pooling designs and nonadaptive group testing: important tools for DNA sequencing*. Vol. 18. World Scientific, 2006.

[86] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[87] Edwin T. Jaynes. "Information theory and statistical mechanics". In: *Physical review* 106.4 (1957), p. 620.

[88] Edwin T. Jaynes. "Information theory and statistical mechanics. II". In: *Physical review* 108.2 (1957), p. 171.

[89] Edwin T. Jaynes. "Prior probabilities". In: *IEEE Transactions on systems science and cybernetics* 4.3 (1968), pp. 227–241.

[90] Edwin T. Jaynes. *Probability theory: the logic of science*. Cambridge university press, 2003.

[91] Min Jiang, Yukun Chen, Mei Liu, S Trent Rosenbloom, Subramani Mani, Joshua C. Denny, and Hua Xu. "A study of machine-learning-based approaches to extract clinical entities and their assertions from discharge summaries". In: *Journal of the American Medical Informatics Association* 18.5 (2011), pp. 601–606.

[92] Thorsten Joachims. "Optimizing search engines using clickthrough data". In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2002, pp. 133–142.

[93] Krusche John. *Doing Bayesian data analysis: a tutorial introduction with R*. 2010.

[94] Justie Su-Tzu Juan and Gerard J. Chang. "Adaptive group testing for consecutive positives". In: *Discrete Mathematics* 308.7 (2008), pp. 1124–1129.

[95] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. "An efficient k-means clustering algorithm: Analysis and implementation". In: *IEEE transactions on pattern analysis and machine intelligence* 24.7 (2002), pp. 881–892.

[96] Maurice G. Kendall. "A new measure of rank correlation". In: *Biometrika* 30.1/2 (1938), pp. 81–93.

[97] Azeem J. Khan, Vigneshwaran Subbaraju, Archan Misra, and Srinivasan Seshan. "Mitigating the true cost of advertisement-supported free mobile applications". In: *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*. ACM. 2012, p. 1.

[98] Ross Kindermann and J. Laurie Snell. *Markov random fields and their applications*. Vol. 1. American Mathematical Society, 1980.

[99] Steven Klepper. "Entry, exit, growth, and innovation over the product life cycle". In: *The American economic review* (1996), pp. 562–583.

[100] Emanuel Knill. "Lower bounds for identifying subset members with subset queries". In: *SODA*. 1995, pp. 369–377.

[101] Emanuel Knill, Alexander Schliep, and David C. Torney. "Interpretation of pooling experiments using the Markov chain Monte Carlo method". In: *Journal of Computational Biology* 3.3 (1996), pp. 395–406.

[102] Yves Kodratoff. *Introduction to machine learning*. Elsevier, 2014.

[103] Igor Kononenko. "Inductive and Bayesian learning in medical diagnosis". In: *Applied Artificial Intelligence an International Journal* 7.4 (1993), pp. 317–337.

[104] Jochen Kruppa, Alexandra Schwarz, Gerhard Arminger, and Andreas Ziegler. "Consumer credit risk: Individual probability estimates using machine learning". In: *Expert Systems with Applications* 40.13 (2013), pp. 5125–5131.

[105] Theodore Levitt et al. *Exploit the product life cycle*. Vol. 43. Graduate School of Business Administration, Harvard University, 1965.

[106] Zhung-Xun Liao, Yi-Chin Pan, Wen-Chih Peng, and Po-Ruey Lei. "On mining mobile apps usage behavior for predicting apps usage in smartphones". In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM. 2013, pp. 609–618.

[107] Aristidis Likas, Nikos Vlassis, and Jakob J. Verbeek. "The global k-means clustering algorithm". In: *Pattern recognition* 36.2 (2003), pp. 451–461.

[108] Duen-Ren Liu and Ya-Yueh Shih. "Integrating AHP and data mining for product recommendation based on customer lifetime value". In: *Information & Management* 42.3 (2005), pp. 387–400.

[109] Tie-Yan Liu. *Learning to rank for information retrieval*. Springer Science & Business Media, 2011.

[110] Wanli Ma, Dat Tran, and Dharmendra Sharma. "A novel spam email detection system based on negative selection". In: *Computer Sciences and Convergence Information Technology, 2009. ICCIT'09. Fourth International Conference on*. IEEE. 2009, pp. 987–992.

[111] David J.C. MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[112] David J.C. MacKay and Radford M. Neal. "Near Shannon limit performance of low density parity check codes". In: *Electronics letters* 32.18 (1996), p. 1645.

[113] Anthony J. Macula. "Probabilistic nonadaptive group testing in the presence of errors and DNA library screening". In: *Annals of Combinatorics* 3.1 (1999), pp. 61–69.

[114] Jean-Michel Marin and Christian Robert. *Bayesian core: a practical approach to computational Bayesian statistics*. Springer Science & Business Media, 2007.

[115] Stephen Marsland. *Machine learning: an algorithmic perspective*. CRC press, 2015.

[116] Marc Mézard and Cristina Toninelli. "Group testing with random pools: Optimal two-stage algorithms". In: *IEEE Transactions on Information Theory* 57.3 (2011), pp. 1736–1745.

[117] Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell. *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[118] Stuart E. Middleton, David C. De Roure, and Nigel R. Shadbolt. "Capturing knowledge of user preferences: ontologies in recommender systems". In: *Proceedings of the 1st international conference on Knowledge capture*. ACM. 2001, pp. 100–107.

[119] Rada Mihalcea. "Graph-based ranking algorithms for sentence extraction, applied to text summarization". In: *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics. 2004, p. 20.

[120] Tom M. Mitchell. *Machine learning*. McGraw-Hill Education, 1997.

[121] Jun Mo Kwon, Jung-in Bae, and Shane C. Blum. "Mobile applications in the hospitality industry". In: *Journal of Hospitality and Tourism Technology* 4.1 (2013), pp. 81–92.

[122] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[123] Laszlo Monostori and Janos Prohaszka. "A step towards intelligent manufacturing: Modelling and monitoring of manufacturing processes through artificial neural networks". In: *CIRP Annals-Manufacturing Technology* 42.1 (1993), pp. 485–488.

[124] Meinard Müller and Masakazu Jimbo. "Consecutive positive detectable matrices and group testing for consecutive positives". In: *Discrete mathematics* 279.1-3 (2004), pp. 369–381.

[125] Meinard Müller and Masakazu Jimbo. "Cyclic sequences of k-subsets with distinct consecutive unions". In: *Discrete Mathematics* 308.2-3 (2008), pp. 457–464.

[126] Giuseppe Munda and Michela Nardo. "On the methodological foundations of composite indicators used for ranking countries". In: *Ispra, Italy: Joint Research Centre of the European Communities* (2003).

[127] Balas K. Natarajan. *Machine learning: a theoretical approach*. Morgan Kaufmann, 2014.

[128] Hung Q. Ngo and Ding-Zhu Du. "A survey on combinatorial group testing algorithms with applications to DNA library screening". In: *Discrete mathematical problems with medical applications* 55 (2000), pp. 171–182.

[129] Mi Jin Noh and Kyung Tag Lee. "An analysis of the relationship between quality and user acceptance in smartphone apps". In: *Information Systems and e-Business Management* 14.2 (2016), pp. 273–291.

[130] Daniel E. O'Leary. *Enterprise resource planning systems: systems, life cycle, electronic commerce, and risk*. Cambridge university press, 2000.

[131] George S. Oreku. "Mobile technology interaction to e-Commerce in promising of u-Commerce". In: *African Journal of Business Management* 7.2 (2013), p. 85.

[132] Do-Hyung Park and Sara Kim. "The effects of consumer knowledge on message processing of electronic word-of-mouth via online consumer reviews". In: *Electronic Commerce Research and Applications* 7.4 (2008), pp. 399–410.

[133] Do-Hyung Park, Jumin Lee, and Ingoo Han. "The effect of on-line consumer reviews on consumer purchasing intention: The moderating role of involvement". In: *International journal of electronic commerce* 11.4 (2007), pp. 125–148.

[134] Raja Patel, Lucy Sulzberger, Grace Li, Jonny Mair, Hannah Morley, Merryn Ng-Wai Shing, Charlotte O'Leary, Asha Prakash, Nicholas Robilliard, Merrin Rutherford, Caitlin Sharpe, Caroline Shie, Logitha Sritharan, Julia Turnbull, Imogen Whyte, Helen Yu, Christine Cleghorn, William Leung, and Nick Wilson. "Smartphone apps for weight loss and smoking cessation: Quality ranking of 120 apps". In: *The New Zealand Medical Journal (Online)* 128.1421 (2015), p. 73.

[135] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[136] Kathryn E. Pedings, Amy N. Langville, and Yoshitsugu Yamamoto. "A minimum violations ranking method". In: *Optimization and engineering* 13.2 (2012), pp. 349–370.

[137] Fernando Peruani and Lionel Tabourier. "Directedness of information flow in mobile phone communication networks". In: *PloS one* 6.12 (2011), e28860.

[138] Réjean Plamondon and Sargur N. Srihari. "Online and off-line handwriting recognition: a comprehensive survey". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.1 (2000), pp. 63–84.

[139] George Pólya. *Mathematics and Plausible Reasoning: Patterns of plausible inference*. Vol. 2. Princeton University Press, 1990.

[140] Lawrence R. Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Vol. 14. PTR Prentice Hall Englewood Cliffs, 1993.

[141] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. "Getting to know you: learning new user preferences in recommender systems". In: *Proceedings of the 7th international conference on Intelligent user interfaces*. ACM. 2002, pp. 127–134.

[142] Al Mamunur Rashid, George Karypis, and John Riedl. "Learning preferences of new users in recommender systems: an information theoretic approach". In: *Acm Sigkdd Explorations Newsletter* 10.2 (2008), pp. 90–100.

[143] Deepak Ravichandran and Eduard Hovy. "Learning surface text patterns for a question answering system". In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics. 2002, pp. 41–47.

[144] Siddheswar Ray and Rose H. Turi. "Determination of number of clusters in k-means clustering and application in colour image segmentation". In: *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques*. Citeseer. 1999, pp. 137–143.

[145] Catharine Reese Bomhold. "Educational use of smart phone technology: A survey of mobile phone application use by undergraduate university students". In: *Program* 47.4 (2013), pp. 424–436.

[146] Charmian Reynoldson, Catherine Stones, Matthew Allsop, Peter Gardner, Michael I. Bennett, S. José Closs, Rick Jones, and Peter Knapp. "Assessing the quality and usability of smartphone apps for pain self-management". In: *Pain medicine* 15.6 (2014), pp. 898–909.

[147] David R. Rink and John E. Swan. "Product life cycle research: A literature review". In: *Journal of business Research* 7.3 (1979), pp. 219–242.

[148] Christian Robert. *Machine learning: a probabilistic perspective*. 2014.

[149] Gareth O. Roberts and Adrian F.M. Smith. "Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms". In: *Stochastic processes and their applications* 49.2 (1994), pp. 207–216.

[150] Steve G Romaniuk and Lawrence O Hall. "Decision making on creditworthiness, using a fuzzy connectionist model". In: *Fuzzy Sets and Systems* 48.1 (1992), pp. 15–22.

[151] Gavriel Salomon. *Distributed cognitions: Psychological and educational considerations*. Cambridge University Press, 1997.

[152] Bernhard Scholkopf and Alexander J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[153] Anne Schuth, Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. "Lerot: An online learning to rank framework". In: *Proceedings of the 2013 workshop on Living labs for information retrieval evaluation*. ACM. 2013, pp. 23–26.

[154] Shahana Sen and Dawn Lerman. "Why are you telling me this? An examination into negative consumer reviews on the web". In: *Journal of interactive marketing* 21.4 (2007), pp. 76–94.

[155] Pak Sham, Joel S. Bader, Ian Craig, Michael O'Donovan, and Michael Owen. "DNA pooling: a tool for large-scale association studies". In: *Nature Reviews Genetics* 3.11 (2002), p. 862.

[156] Claude Elwood Shannon. "A mathematical theory of communication". In: *ACM SIGMOBILE Mobile Computing and Communications Review* 5.1 (2001), pp. 3–55.

[157] Kent Shi and Kamal Ali. "Getjar mobile application recommendations with very sparse datasets". In: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2012, pp. 204–212.

[158] Scott Spangler. *Accelerating Discovery: Mining Unstructured Information for Hypothesis Generation*. Vol. 37. CRC Press, 2015.

[159] Suvrit Sra, Sebastian Nowozin, and Stephen J. Wright. *Optimization for machine learning*. Mit Press, 2012.

[160] Andreas Stippig, Ulrich Hübers, and Markus Emerich. "Apps in sleep medicine". In: *Sleep and Breathing* 19.1 (2015), pp. 411–417.

[161] Stoyan R. Stoyanov, Leanne Hides, David J. Kavanagh, Oksana Zelenko, Dian Tjondronegoro, and Madhavan Mani. "Mobile app rating scale: a new tool for assessing the quality of health mobile apps". In: *JMIR mHealth and uHealth* 3.1 (2015).

[162] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation by joint identification-verification". In: *Advances in neural information processing systems*. 2014, pp. 1988–1996.

[163] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. "Personalizing search via automated analysis of interests and activities". In: *ACM SIGIR Forum*. Vol. 51. 3. ACM. 2018, pp. 10–17.

[164] My T. Thai. *Group Testing Theory in Network Security: An Advanced Solution*. Springer Science & Business Media, 2011.

[165] My T. Thai, Ying Xuan, Incheol Shin, and Taieb Znati. "On detection of malicious users using group testing techniques". In: *Distributed Computing Systems, 2008. ICDCS'08. The 28th International Conference on*. IEEE. 2008, pp. 206–213.

[166] Sergios Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic Press, 2015.

[167] Robert Tibshirani, Martin Wainwright, and Trevor Hastie. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

[168] Luke Tierney. "Markov chains for exploring posterior distributions". In: *the Annals of Statistics* (1994), pp. 1701–1728.

[169] Ian Tonks. "Bayesian learning and the optimal investment decision of the firm". In: *The Economic Journal* 93 (1983), pp. 87–98.

[170] David C. Torney, F. Sun, and William J. Bruno. "Optimizing nonadaptive group tests for objects with heterogeneous priors". In: *SIAM Journal on Applied Mathematics* 58.4 (1998), pp. 1043–1059.

[171] Evangelos Triantaphyllou. *Data mining and knowledge discovery via logic-based methods: theory, algorithms, and applications*. Vol. 43. Springer Science & Business Media, 2010.

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

[173] Catherine Tucker. "The economics value of online customer data". In: *Background Paper* 1 (2010).

[174] Hiroaki Uehara and Masakazu Jimbo. "A positive detecting code and its decoding algorithm for DNA library screening". In: *IEEE/ACM transactions on computational biology and bioinformatics* 6.4 (2009), pp. 652–666.

[175] Heli Väätäjä. "User experience evaluation criteria for mobile news making technology: findings from a case study". In: *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*. ACM. 2010, pp. 152–159.

[176] Mathias Van Singer, Anne Chatton, and Yasser Khazaal. "Quality of smartphone apps related to panic disorder". In: *Frontiers in psychiatry* 6 (2015), p. 96.

[177] Vladimir Vapnik. *Statistical learning theory. 1998*. Wiley, New York, 1998.

[178] Hannu Verkasalo, Carolina López-Nicolás, Francisco J. Molina-Castillo, and Harry Bouwman. "Analysis of users and non-users of smartphone applications". In: *Telematics and Informatics* 27.3 (2010), pp. 242–255.

[179] Martin Wainwright. "Information-theoretic bounds on sparsity recovery in the high-dimensional and noisy setting". In: *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*. IEEE. 2007, pp. 961–965.

[180] Jun Wang. "Artificial neural networks versus natural neural networks: A connectionist paradigm for preference assessment". In: *Decision Support Systems* 11.5 (1994), pp. 415–429.

[181] Wei Wang, Martin J. Wainwright, and Kannan Ramchandran. "Information-theoretic limits on sparse signal recovery: Dense versus sparse measurement matrices". In: *IEEE Transactions on Information Theory* 56.6 (2010), pp. 2967–2979.

[182] Jon Williamson. "The philosophy of science and its relation to machine learning". In: *Scientific Data Mining and Knowledge Discovery*. Springer, 2009, pp. 77–89.

[183] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. "Listwise approach to learning to rank: theory and algorithm". In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 1192–1199.

[184] Qiang Xu, Jeffrey Erman, Alexandre Gerber, Zhuoqing Mao, Jeffrey Pang, and Shobha Venkataraman. "Identifying diverse usage behaviors of smartphone apps". In: *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM. 2011, pp. 329–344.

[185] Gongjun Yan, Danda B. Rawat, Hui Shi, and Awny Alnusair. "Developing and applying smartphone apps in online courses". In: *Journal of Information Systems Education* 25.2 (2014), p. 149.

[186] Jun Yan, Ning Liu, Gang Wang, Wen Zhang, Yun Jiang, and Zheng Chen. "How much can behavioral targeting help online advertising?" In: *Proceedings of the 18th international conference on World wide web*. ACM. 2009, pp. 261–270.

[187] Jun Yan, Ning Liu, Gang Wang, Wen Zhang, Yun Jiang, and Zheng Chen. "How much can behavioral targeting help online advertising?" In: *Proceedings of the 18th international conference on World wide web*. ACM. 2009, pp. 261–270.

[188] Yi Yang, Feiping Nie, Dong Xu, Jiebo Luo, Yueting Zhuang, and Yunhe Pan. "A multimedia retrieval framework based on semi-supervised ranking and relevance feedback". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.4 (2012), pp. 723–742.

[189] Peifeng Yin, Ping Luo, Min Wang, and Wang-Chien Lee. "A straw shows which way the wind blows: ranking potentially popular items from early votes". In: *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM. 2012, pp. 623–632.

[190] Theodora Zarmpou, Vaggelis Saprikis, Angelos Markos, and Maro Vlachopoulou. "Modeling users' acceptance of mobile services". In: *Electronic Commerce Research* 12.2 (2012), pp. 225–248.

[191] Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. "A regression framework for learning ranking functions using relative relevance judgments". In: *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2007, pp. 287–294.

[192] Feng Zhu and Xiaoquan Zhang. "Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics". In: *Journal of marketing* 74.2 (2010), pp. 133–148.

[193] Hengshu Zhu, Enhong Chen, Hui Xiong, Kuifei Yu, Huanhuan Cao, and Jilei Tian. "Mining mobile user preferences for personalized context-aware recommendation". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 5.4 (2015), p. 58.

[194] Jan Zyt, Will Klosgen, and Jan Zytkow. *Handbook of data mining and knowledge discovery*. Oxford university press, 2002.