Predicting Customer Preferences in Non-experimental Retail Settings

Simon Chan

Supervisor: Prof Philip Treleaven
Supervisor: Dr Licia Capra

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University College London.

Department of Computer Science

University College London

April 2, 2015

I, Simon Chan, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

This thesis investigates the application of computational statistics and Machine Learning in consumer preference prediction, with specific reference to the challenges imposed by real world operational retail environments. Some retailers base their competitiveness on Machine Learning. For instance, Dunnhumby analyzes more than 400 million online consumer records for retailers, such as Tesco, to optimise business decisions.

The experiments in this thesis investigate three main challenges commonly presented in operational scenarios that hinder the application of Machine Learning in retail environments:

1. The measurement of correlation of feature factors for Machine Learning in a noisy setting;
2. The exploration and exploitation balance for predicting purchase preferences on new products;
3. The model adaptability to the changing dynamics over time.

A design of a distributed Machine Learning framework for building practical applications of consumer preference prediction is also presented.

Experiment 1: Correlation between Contextual Information and Purchase Behaviours under a Non-experimental Retail Environment

The first experiment applies statistical methodologies, namely odds ratio and Mantel-Haenszel method, to analyze contextual information in a retail business. More specifically, it investigates the correlation between customers' recent online browsing behaviours on Boots.com and their in-store purchase behaviours at Boots' retail stores nationally in a non-experimental noisy setting. Methodologies such as stratified analysis with K-means clustering are proposed to detect and eliminate confounding factors that affect the evaluation of the correlation. The dataset for this experiment, provided by Boots UK, is the first year of a 2-year anonymised real in-store and online purchase records data. It contains profiles of 10,217,972 unique consumers who are Advantage Card holders and 2,939 unique selected products under 10 different brands.

Experiment 2: Resources Allocation of Exploration and Exploitation for New Products under Retail Constraints

The second experiment provides a two-stage batch solution based on matrix factorization and binary integer programming to optimise the customer response rate to new products of a simulated group buying system. This experiment investigates how the balance between the exploration of new products and the exploitation of existing known model affects overall business gains through purchase prediction and recommendation. In this experiment, the products are new with no prior profile and the number of new products a retailer can recommend to each customer is limited. The effectiveness of one of the traditional experimental design techniques in improving the learning efficiency during the exploration process is evaluated.

Experiment 3: Continuous Model Selection for a Changing Retail Environment

The third experiment investigates, using root-mean-square error and mean average precision measures, the adaptability of data model for consumer purchase prediction in a non-static retail environment. In particular, it analyzes the prediction accuracy of data models with static parameters over time. A continuous model selection approach by using an automatic hyperparameter tuning technique, namely random search, is proposed and is evaluated. The results challenge the traditional assumption that a one-off initial model selection is sufficient. The dataset for this experiment is a 2-year anonymised real in-store and online purchase records data provided by Boots UK.

System Design: A Distributed Machine Learning Framework with Automated Modeling

This system design outlines the concept and system architecture. It also demonstrates scenarios of a distributed Machine Learning framework for (i) evaluating, comparing and deploying scalable learning algorithms, (ii) tuning hyperparameters of algorithms manually or automatically and (iii) evaluating model training status. The design has become the foundation of a popular open-source software project - PredictionIO. The project is followed by over 5000 data scientists and practitioners on Github.

Contributions to Science

The major contribution of this thesis is to offer robust research-based methodologies to handle prediction challenges in real world operational environment for retail businesses. Computational statistics and Machine Learning methodologies are proposed to 1) identify contextual factors that are relevant to consumer preference in noisy non-experimental setting; 2) determine the importance of exploration and exploitation for new products under real-world constraints; 3) adjust data model continuously to adapt to changes in retail environments.

This thesis contributes to the existing literature in a number of ways. First, this research proposes a novel statistical method to isolates the influence of confounding factors in correlation analysis for consumer preference prediction. It is a topic that received little attention in empirical literature. Second, this research proves the existence of the correlation between consumer online browsing and in-store purchase behaviours in a real retail dataset. This is a significant finding for the retail industry to improve prediction accuracy in the future. Third, this research examines the influence of the balance between exploration and exploitation of new product profiles on maximising business gains. Forth, this research proves that random selection surprisingly outperforms D-optimal experimental design in some retail cases. Fifth, this research challenges the existing assumption that model selection is needed only once at the initial stage. It proves that prediction accuracy can be improved significantly by continuous model selection. Sixth, this thesis presents the implementation of a continuous model selection approach by using automatic hyperparameter tuning techniques. Finally, this thesis presents a design of a distributed system that can be used for building predictive retail applications.

Acknowledgements

It would be impossible to complete this thesis without the support of intelligent and kind people around me, only some of whom can be mentioned here. First, I would like to thank my PhD supervisors Prof. Philip Treleaven and Dr. Licia Capra for their wise, consistent, and indispensable guidance throughout these years. They make my research work possible with not only their scholarly support, but also with their patience, encouragement, thoughtful feedback and inspiring questions. I would also like to thank my industrial supervisor Dr. Alan Payne for offering incredible insights on real-world scientific challenges in the industry. Without his help, the access to real-world data for the experiments in this thesis would never be possible. A special thanks goes to Dr. Jun Wang and Dr. Mark Sinclair who kindly broadened my understanding in the Information Retrieval and Machine Learning disciplines. I am also thankful to the examiners of the previous viva exams, Prof. Ingemar Cox and Dr. Robert Smith, who have given invaluable feedback on my work. Furthermore, I would like to acknowledge the academic, financial and industrial support of the University College London, Boots UK and Procter & Gamble.

Above all, I thank my wonderful wife, Esther, for the constant love and support. I also owe a great deal of gratitude to my parents and brother for everything throughout my entire life. I dedicate this thesis to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter presents an overview of the thesis. First, the motivation for this research is introduced along with the research topic. Furthermore, the objectives of this research are discussed. Subsequently, this chapter describes the research and the scientific contributions of this thesis. This chapter ends by describing the structure of the whole thesis and showing a list of publications relating to this thesis that have been done so far.

1.1 Motivation

The commercial application of Machine Learning has become increasingly popular in recent years. Some retailers base their competitiveness on prediction through consumer data analysis [1]. For instance, they try to predict what consumer will purchase accurately in a timely manner. Many online retailers, such as Amazon [2] and Netflix [3], are well-known for aiming to provide better shopping experience based on consumer analytics. The demand for consumer analytics for traditional retailers is also high. For instance, Dunnhumby analyzes more than 300 million online consumer records for retailers, such as Tesco, to optimise business decisions. Boots UK, similarly, keeps track of the purchase behaviours of its 18 million loyalty card holders.

A main reason for retailers to conduct consumer analytics is to predict consumer preference. By gaining an insight on future consumer preference, businesses hope to make better management decisions on various areas, such as inventory management, price management, sales and marketing strategy and in-store product placement [4]. Another benefit is that businesses can recommend personalized information or products to consumers. Some motivations include increasing revenue for businesses, improving

the shopping experience and enhancing consumer satisfaction [5]. The prediction of future consumer preference can be done by a computer system learning from previous data to predict unknown or future events. This kind of computation is called Machine Learning in the research discipline.

An example application of Machine Learning in retail is recommender systems. Literature shows that Recommender systems can provide satisfactory information seeking or discovering experience to consumers by predicting their preference [6]. Various Machine Learning and statistics techniques can be applied to build the predictive models in recommender systems [7, 8]. With different underlying assumptions, predictive models try to predict future consumer preference towards items (such as products or other information), which are usually represented by numeric scores, probability or ratings, as accurate as possible. In most scenarios, top-ranked N items will then be recommended to consumers according to the result of prediction. A large number of literature focuses on improving the accuracy of prediction, which is commonly evaluated by root-mean-squared error (RMSE), mean absolute error (MAE) or recall and precision metrics [9].

With the advancement in data storage technology, the cost of storing big data is decreasing [10]. The rise of schemaless database design has also facilitated the storage of unstructured data [11]. Therefore, the size and type of data being stored is increasing rapidly in some businesses. Although it is generally true that the more data being used as training data the better prediction accuracy can be produced, consumer preference prediction in real world retail settings poses unique challenges.

The first challenge is that the shopping context of consumers changes rapidly. For example, the same consumer may be shopping for different reasons at different stores during different time. The same consumer may also be shopping in a physical store, through a mobile app or on the web. The question is, among so much contextual information that the system can collect, how the system can determine the relevancy of a context to the prediction efficiently and accurately. A real world operational environment, which is in a non-experimental setting, imposes unique challenges on identifying the relevant factors or contextual information. It is not uncommon that consumer behaviours are affected by multiple activities from different channels at the same time. The challenge is to identify relevant factors for building data model in such noisy en-

vironment and it is non-trivial. It is important to determine which type of contextual information is relevant to, i.e. correlated to, consumer preference. If a prediction model is built based on a wrong assumption, the model will become invalid. It is also important to determine the weight, i.e. the importance, of each relevant contextual information in the prediction model. Unlike research experiments where controlled groups can be setup, consumer behaviours happen, and are recorded, naturally in business environment without any experimental influence. In the field of statistics, this kind of data analysis is called observational study [12]. The significance of the this condition is that confounding factors exist which affect the validity of the conclusion drawn by standard data analysis and Machine Learning techniques.

Another challenge retailers are facing is that the interaction between retailers and consumers is constrained. For instance, retailers can only recommend a limited number of products to each consumer by email in a certain time period, or they can sell a product within a certain period of time only. These constraints create dilemmas such as whether retailers should use the precious interaction opportunity to improve data model by gaining more new knowledge or to maximize their profits by utilizing existing knowledge. In the field of statistics, this is commonly called an exploration and exploitation problem. The balance between exploration and exploitation affects how well a retail business goal, such as maximizing the number of purchase, can be achieved potentially.

In addition to the first two challenges aforementioned, the third challenge is that retail business dynamics rapidly change over time. For instance, consumer behaviours may change due to external factors such as social changes; consumer base may change; and the products the retailers are selling may also be changed. It is possible that these changes invalidate existing predictive model. Although the problems of model fitting and hyperparameter optimisation have been studied for decades in various disciplines, the adaptiveness of the initially selected model is not as well understood. To my best knowledge, as of the time this study is conducted, existing research in consumer preference prediction in retail analytics does not commonly evaluate whether re-selecting hyperparameters for model selection repeatedly over time would have any impact on the prediction accuracy of recommender systems in retail settings.

Finally, applying Machine Learning techniques to existing retail software applica-

tions is challenging: the learning curve of Machine Learning techniques is steep; the selection of algorithm and hyperparameter tuning is time-consuming; and the construction of distributed system infrastructure for processing large amount of transactional data is non-trivial. State-of-the-art research results cannot be implemented to solve real-world settings unless this technical barrier is removed.

The main motivation of this research is to improve the accuracy of consumer preference prediction in retail settings by addressing various Machine Learning challenges in real-world scenarios.

1.2 Objective of this Research

Our research objective is to improve the accuracy of consumer preference prediction by addressing the practical real-world challenges of Machine Learning in retail businesses. The main hypothesis of this research states that:

> By applying various Machine Learning and statistical techniques selectively and carefully, the accuracy of consumer preference prediction can be improved by taking operational retail-specific challenges into consideration when building a prediction system.

To validate this hypothesis, there are four main tasks for this research:

1. Analyze the correlation between retail-related contextual information and consumer preference in non-experimental environments.

The first task is to prove that the accuracy of consumer preference prediction can be improved by incorporating retail-specific contextual information into the data model. It can be done easily by showing that there is a correlation between a contextual factor and future consumer preference. This research, therefore, proposes a robust and automated methodology for this kind of correlation analysis for data collected in non-experimental noisy retail environments.

2. Investigates the balance between exploration and exploitation of new products and how this balance affects overall business gains.

The second task is to prove that the accuracy of consumer preference prediction for new products can be improved by making a good balance of resources allocation between

exploration and exploitation. Resources means the opportunity to recommend products to consumers, which is limited by real-world retail constraints such as the maximum number of products that can be displayed in an user interface. This research investigates different approaches to allocate the limited resources for exploration, i.e. to learn more about new products, and exploitation, i.e. to make accurate prediction.

3. To investigates the adaptability of data model for consumer purchase prediction in a non-static retail environment.

The third task is to prove that the accuracy of consumer preference prediction can be improved by continuously adjusting the hyperparameter settings of a selected algorithm upon the collection of new data. Previous literature assumes that the process of hyper-parameter optimisation, or model selection, is needed only once at the initial stage. In this research, I investigate whether re-selecting hyperparameters for model selection repeatedly after the introduction of new data would have any impact on the accuracy of consumer preference prediction in a dynamic retail setting.

4. To propose a system design for building scalable predictive applications.

The fourth task is to propose a system design for building real-world predictive retail applications. Some retailers collect huge amount of data about consumers and products. A software framework that can process big data and build predictive models efficiently is needed. This research presents a developer-friendly design of a distributed system for implementing research-based Machine Learning methodologies in practical software applications.

1.3 Data used in this Research

The dataset for this thesis, provided by Boots UK and Procter & Gamble, is a 2-year anonymised records of Advantage Card (the loyalty card of Boots UK) holders who have browsed the selected products online on Boots.com and of those who have purchased the selected products in any Boots retail store in UK. It contains 10,217,972 unique consumers who are Advantage Card holders and 2,939 unique selected products under 10 different brands. There are 17,834,762 in-store purchase transaction records in the first-year dataset. There are 21,668,137 in-store purchase transaction records in the second-year dataset. We can associate consumers' online browsing and in-store

purchasing behaviours with unique Advantage Card numbers.

Boots.com is the digital face of Boots UK, attracting approximately 1.5m unique visitors per week. Boots UK operates 2,500 stores nationwide. The Boots Advantage Card system is a loyalty scheme that tracks individual customer transactions in both of these online and offline channels.

1.4 Structure of the Thesis

Structure of this thesis is as follows,

- Chapter 2 reviews background information on a number of key concepts in the areas of Machine Learning, recommender systems and consumer preference prediction in retail and briefly describes the literature related to my research.

- Chapter 3 describes a robust correlation analysis methodology for data collected in non-experimental environments. It has been illustrated as a case study that analyses the correlation between consumer's online browsing behaviours and their in-store purchasing behaviours. Boots UK and P&G dataset has been used in the study.

- Chapter 4 describes the exploration stage and exploitation stage for predicting consumer preferences on new products. It further describes possible operational constraints of retail businesses followed by introducing a two-stage approach to maximize overall prediction gain using various methodologies.

- Chapter 5 describes a continuous model selection methodology to improve the accuracy of consumer preference prediction as more data is collected while the environment is changing.

- Chapter 6 describes the system design of a distributed Machine Learning framework for developers to build practical predictive retail applications.

- Chapter 7 summarises this research and outlines possibilities for future work.

1.5 Publications Related to This Thesis

The following publications are related to this thesis:

- Simon Chan and Licia Capra. From Online Browsing to Offline Purchases: Analyzing Contextual Information in the Retail Business. In *Proceedings of the 4th Workshop on Context-Aware Recommender Systems (CARS) in ACM Recsys. Dublin, Ireland, September 2012*.

- Simon Chan. Constrained by Limited Information and Recommendation Opportunities: An Exploration and Exploitation Problem for Recommender Systems. In *Proceedings of the 14th IEEE International Conference on Information Reuse and Integration. CA, USA, August 2013*.

- Simon Chan, Philip Treleaven and Licia Capra. Continuous Hyperparameter Optimization for Large-scale Recommender Systems. In *Proceedings of the IEEE International Conference on Big Data. CA, USA, October 2013*.

- Simon Chan, Thomas Stone, Kit Pang Szeto and Ka Hou Chan. PredictionIO: A Distributed Machine Learning Server for Practical Software Development. In *Proceedings of the ACM Conference of Information and Knowledge Management (CIKM). CA, USA, October 2013*.

- Simon Chan. A good nudge trumps a good prediction. In *O'Reilly Radar - http://radar.oreilly.com/2014/07/a-good-nudge-trumps-a-good-prediction.html, July 18, 2014*.

- Simon Chan. Machine Learning and the Jargons. In *PredictionIO Blog - http://blog.prediction.io/machine-learning/, April 8, 2013*.

- Simon Chan and Philip Treleaven. Continuous Model Selection for Large-scale Recommender Systems. In *Proceedings of the Volume 33, Handbook of Statistics: Big Data Analytics*.

Chapter 2

Background

This chapter introduces the general context of consumer preference prediction in the retail industry. I present the motivation for retail businesses to predict customer preference and draw a line between pure prediction and nudge. In addition, the difference between conducting correlation and causation analysis in experimental environment and in non-experimental retail environment is also discussed. I further give a brief introduction of the type of data retail businesses can use to predict customer preference. An introduction and a detailed review of some latest research of recommender systems, a common application of consumer preference prediction in the retail industry, is presented at last.

Literature review on related work will be presented separately in the following chapters of contributions and experiments.

2.1 Consumer Preference Prediction for Retail

The advancement of Machine Learning techniques used to predict customers' preferences has enabled the development of a wide variety of predictive applications in the retail industry. Some retailers base their competitiveness on consumer analytics [1] using Machine Learning and they aim to offer the right products to customers accurately in a timely manner with Recommender Systems. Many online retailers such as Amazon [2] and Netflix [3] are well-known for providing superior shopping experience using Item-based Collaborative Filtering, k-Mean clustering and other Machine Learning techniques . A main reason for retailers to develop Machine Learning techniques is to predict consumer preference. By having an insight on future consumer preference,

businesses can make better retail decisions on areas, such as:

- Establish association rules - if you buy this, you have a high chance of buying these [13]
- Predict customer shopping lists from Point-of-sale purchase data [14]
- Inventory management
- Price management
- Sales and marketing strategy
- In-store product placement [4]
- Dynamic pricing

Another benefit is that businesses can recommend personalized information or products to consumers. Good recommendation increase revenue for businesses and it improves the shopping experience, thus satisfaction, for consumers [5].

2.2 The Retail Motivation: With Pure Prediction and Nudge

A working predictive model is a powerful business tool for retail businesses. By translating data into insights and subsequent actions, businesses can offer better customer experience, retain more customers, and increase revenue. For these reasons, retailers are interested in developing, or purchasing, Machine Learning solutions that can predict future or unknown customer behaviours. The implementation of Machine Learning in businesses is, however, non-trivial. During this research investigation, the problem often is not merely the quality of data or algorithms, but also the evaluation metrics that aligns with the desired business motivation. Oftentimes, a misguided evaluation approach will lead retailers to adopt ineffective Machine Learning solutions. To set the right evaluation metrics, a good understanding behind the motivation for the prediction is needed. Broadly speaking, they are motivated to predict customer preferences by either or both of the two objectives: 1) improve customers' experience as they navigate through the e-commerce website, mobile application or physical retail store; 2) increase business revenue, be it short-term or long-term.

Literatures show that Recommender Systems that predict user preferences increase direct and indirect revenue for retailers such as LeShop - the No.1 e-grocer in Switzerland since 1998 [15] and Nova Pontocom - the second largest Latin American online retailer [16]. Some retailers, therefore, believe that as long as they can accurately predict their customers future purchasing preferences, they can increase revenue. This supposedly flawless assumption that accurate predictions can increase revenue in all situations is easily overthrown when the accuracy evaluation metrics are implemented in reality. As a terminology in the Machine Learning discipline, a metric measures how well a predictive model performs, usually based on some pre-defined scoring rules. Different models can then be compared. For instance, in Recommender Systems research, metrics such as recall/precision, Root Mean Square Error (RMSE), and Mean Average Precision (MAP) [17, 18, 9, 19] are frequently used to evaluate how well the models perform. Roughly speaking, these metrics assume that a good model can accurately predict those products a customer will purchase or give the highest rating to. The problem with this assumption can be illustrated by a simple example. For example, a customer wants to buy cereal and milk from an e-commerce website. Once the customer arrives at the website, the Recommender System accurately predicts they will buy cereal and milk and shows them on the screen. The customer clicks and buys those products. In this case, the Recommender System has probably improved the customer's experience, but it has not increased revenue for the retailer because the customer was going to buy cereal and milk, regardless of the accuracy of the prediction. If the aim is to increase revenue, the metric should, for example, focus on how well the model could predict and recommend products that will nudge the customer to buy other products. A recent research study has reviewed this problem [20]. In this study, the experiments compare the recommendation list of a traditional Recommender System that minimises prediction error with respect to users' choices with a modified recommendation list that takes into account the revenue the business could have made if the user accepts the recommendation. The study shows that the modified list that is re-arranged according to potential revenue gain improves business revenue significantly.

As shown above, if the motivation is to improve customers' experience, pure prediction of what customers like to buy may be sufficient. On the other hand, if the motivation is to increase business revenue, the Machine Learning implementation needs to

take one step further to, not only predict, but also to influence, customers' preferences. This motivation is sometimes called nudge. Researchers [21] and businesses have a vested interest in nudging. For instance, Lowes grocery store in El Paso [22] successfully conducted an experiment that nudged customers to buy more produce by simply adding huge green arrows on the floor that pointed toward the produce aisle. Another successful example of nudging is checkout charity [23]: retailers raise millions of dollars for charity by asking customers for small donations at the checkout screens.

Applying the predictive power of Machine Learning techniques to nudge, if done correctly, can be extremely valuable. Many bright statistical minds are, however, confused by the subtle difference between a good prediction and a good nudge. If the two in the evaluation process are mixed up, an unsuitable model that does not help fulfill the motivation may be selected. Facebook's emotional contagion experiment [24], despite its controversy, not only illustrates how data influence users emotions, but also gives us a vivid example of when the goal of a metric should be to measure influence (or nudge) rather than predict. The best metric is one that is consistent with business motivation.

2.3 Correlation and Causation in Experimental and Non-experimental Environments

2.3.1 Correlation and Causation

Correlation is a statistical representation of the extent to which two or more variables fluctuate each other. It enables predictive models to use the value of one variable to predict the value of another. One of the main challenges of predictive modeling in Machine Learning is to identify a representative set of features from the data. Correlation study is important for predicting customers' preferences because a good representative set often contains features that are highly correlated with the predicting value, yet uncorrelated with each other [25].

To formulate correlation, the understanding of a similar concept called covariance is needed. Covariance represents how two variables are related. A positive covariance means the variables are positively related, while a negative covariance means the

variables are inversely related. The formula for calculating covariance is:

$$cov_{x,y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1} \tag{2.1}$$

where x is the independent variable, y is the dependent variable, N is the number of data points in the sample, $\bar{x}$ is the mean of the independent variable x and $\bar{y}$ is the mean of the dependent variable y.

Correlation, which takes one step further compared to covariance, also determines how the two variables are related. Besides determining whether two variables are positively or inversely related, correlation also measures the degree to which the variables tend to change together. Covariance does not use a standard unit of measurement, so it cannot measure the degree to which the variables change together. Correlation solves this problem by standardizing the unit of measurement of two variables, so it can measure how closely the two variables change. The formula of correlation is:

$$r_{x,y} = \frac{cov_{x,y}}{s_x s_y} \tag{2.2}$$

$r_{x,y}$ is the correlation of the variables x and y, $cov_{x,y}$ is the covariance of the variables x and y, s_x is the sample standard deviation of the random variable x and s_y is the sample standard deviation of the random variable y.

The value of correlation, i.e. the correlation coefficient, ranges between 1 and -1 inclusive. If $r_{x,y} = 0$, the two variables are uncorrelated. If $r_{x,y} = 1$, then the two variables are completely correlated in the same direction. This means that when the value of a variable is changed, the value of the second variable should have also changed proportionally in the same direction. If $1 > r_{x,y} > 0$, the two variables have a less than perfect positive correlation The strength of the correlation increases as $r_{x,y}$ increases. If $r_{x,y} = -1$, the variables are completed correlated in the opposite direction. When the value of a variable is changed, the value of the other variable should have changed proportionally in the opposite direction. If $0 > r_{x,y} > -1$, then the two variables have a less than perfect negative correlation, with the strength of the correlation increases as $r_{x,y}$ decreases.

Causation is different from correlation. Causation represents that the change of the value of one variable causes the value of another to change. Intuitively, if retailers can identify causality between a variable and a desired customer action, i.e. the change of

one, or more, adjustable variable causes the change of a desired customer action, they can nudge customers to achieve certain business objectives, e.g. to increase revenue.

One of the common problems of Machine Learning implementation in the industry and even research disciplines is the confusion between correlation and causation. When two variables are correlated, positively or negatively, it does not guarantee causality between them. For any two correlated variables, x and y, the following relationships are possible:

1. x causes y;
2. y causes x;
3. x causes y AND y causes x;
4. x and y are consequences of a common cause, but do not cause each other;
5. There is no connection between x and y; the correlation is coincidental.

It is, therefore, logically incorrect to state that if x and y are correlated, x must cause y. This type of logical fallacy is sometimes referred as *cum hoc ergo propter hoc* logical fallacy, which is Latin that means "after this, therefore because of this". The relationship between correlation and causation has been discussed in multiple disciplines [26, 27, 28, 29].

2.3.2 Experimental and Non-Experimental Studies

Many Machine Learning models, for instance, the classic and widely-applied association rules mining [30, 31] and Market basket analysis [32, 33] are built based upon a correlation of variables to predict customer preferences. By observing correlated variables, or events, retailers can predict what products customers will likely purchase and thus improve their shopping experience accordingly.

Correlation is useful for customer preference prediction, but is it useful for nudge? To implement Machine Learning to nudge customers towards some choices or actions, it is important to understand that correlation does not necessarily mean causation, that is, the *cum hoc ergo propter hoc* logical fallacy. For instance, research [34] shows that a brand's display-advertising campaign on Yahoo! Front Page (x) is correlated to the keyword searches of the name of that brand (y), however, x does not cause y. It means

that, unlike what the business expected, the display-advertising campaign of the brand does not increase the keyword searches about it.

It is, however, also important not to fall into the opposite fallacy. Correlation *may imply* causation. While there are a number of established statistical methods to identify correlation between two correlated variable or events, it is generally very challenging to determine causality. The industry relies on human judgement and experimental studies to verify causality after correlation is identified. Broadly speaking, there are two types of studies for identifying correlation: observational studies and experimental studies.

In observational studies, under a non-experimental environment, customers' behaviours are observed in the real world situation without intervention. This kind of study is less intrusive to customers and thus can potential analyze a very large number of customers. The problem is that variables, such as external confounding factors, cannot be controlled by the researchers. Positive correlation found by such studies may be a result of statistical coincidence only. Practically, observational studies serve as good starting points to eliminate variable pairs that are not correlated with each other and to identify pairs that are potentially correlated. After correlation is found between two variables, although causal relationship and even correlation should not be assumed at this point, researchers can follow up with experimental studies for these variables.

In experimental studies, the experiment environment as well as the population of the customers can be carefully selected, and controlled. Researchers can control, or isolate, the effect of irrelevant variables when the correlation between two targeted variables is being analyzed. If the study is designed carefully and x is noticed to causes better outcome for business objective y than the placebo, then it is generally good enough in practice to assume the cause and effect.

2.4 Data for Consumer Preference Prediction

The prediction of future preference of consumer is usually made possible by analyzing data of historical consumer behaviours and attributes of consumers and products that, directly or indirectly, represents consumer preference at different context. Such data can be categorized as explicit feedback, implicit feedback [35, 36, 37] and contextual information [38, 39].

Explicit Feedback

The data of historical consumer behaviours is called explicit feedback when consumers express their preference explicitly. Retail sites, such as Amazon.com, encourage consumers to give rating to products. A new form of expression influenced by social networking sites such as Facebook is also adopted: The 'Like' and 'Dislike' buttons. Rating and Like/Dislike are both explicit feedback from consumers. This kind of feedback is subjected to a number of problems, such as the bias of rating scale, e.g. some users always tend to give higher rating than the others, and the intrusiveness to consumers' shopping experience.

Implicit Feedback

Data that imply consumer preference indirectly, such as web visit log, link clicks and purchase transactions, is called implicit feedback. This type of feedback does not require extra interaction with consumers. Implicit feedback is relatively unbiased and is less intrusive.

Contextual Information

Contextual information, such as time, location, sales promotion, extra information about the products and additional attributes of consumers, is often available in the database of retail businesses. Research has shown that relevant contextual information can be used to improve the performance of Recommender Systems.

2.5 Recommender Systems

Recommender Systems is a mainstream research topic and it has been widely adopted in the retail industry [6, 7]. Recommender Systems generally represent systems that recommend items to users. Recommender Systems can be implemented by techniques from disciplines such as statistics, information retrieval and Machine Learning [40].

Generally speaking, the goal of a Recommender System is to provide satisfactory information seeking or discovering experience by predicting consumer preference. Various Machine Learning and statistics techniques that can be used for building Recommender Systems [7, 8]. Despite the difference between their underlying assumptions and mathematical models, they all try to predict future consumer preference towards

items (such as products or other information), which are usually represented by numeric scores, probability or ratings, as accurate as possible. In some scenarios, top-ranked N number of items will be recommended to consumers according to the result of prediction. A large number of literatures focus on improving the accuracy of prediction, which is commonly evaluated by RMSE, MAE or recall and precision metrics [9].

Since Recommender System is one of the most popular application of Machine Learning to predict customer preferences in retail industry, this chapter introduces some basic concepts and state-of-the-art techniques of Recommender Systems. A review of Context-aware Recommender Systems (CARS), an emerging research discipline that incorporates additional contextual information into the system in order to improve accuracy, is also presented. Commonly used metrics for evaluating the recommendation performance are also introduced.

Classification of Algorithms for Recommender Systems

Algorithms for implementing Recommender Systems are classified into the following categories [7]:

- Content-based Filtering
- Collaborative Filtering
- Hybrid

Content-based filtering

Content-based filtering algorithms predict users' preferences towards items according to the content information of item profiles and the content information of user profiles [41]. A user profile contains information representing a user's preferences. It can be elicited explicitly. For example, the system can ask a targeted user to fill in questionnaires. A user profile can also be learned implicitly using historical behaviour data of a targeted user. An item profile contains information representing the characteristics of items. The ranking of items for each user is based on the similarity between item profiles and the user profile of the targeted user. Usually, user profiles and item profiles are represented by a set of keywords. The similarity is represented by a score which is estimated by Information Retrieval and Information Filtering techniques, such as term

frequency-inverse document frequency (TF-IDF). A general formulation of similarity score between user u and item i can be written as:

$$score(u,i) = IR_similarity(UserProfile(u), ItemProfile(i)) \tag{2.3}$$

Other Machine Learning techniques can also be used, such as the Bayesian classifier, clustering, decision trees and artificial neural networks.

Collaborative filtering

Unlike content-based filtering algorithms, collaborative filtering algorithms predict users' preferences towards items according to the historical preference of *other users*. Collaborative filtering (CF) algorithms are further categorized into two types [42]:

- Memory-based CF (or heuristic-based CF)
- Model-based CF

Memory-based algorithms predict a user's preference towards an item based on the entire collection of historical preferences of users towards items. One approach of memory-based algorithms, called the user-based approach, relies on the similarity between users. In user-based approach, the estimation of the unknown preference $S_{u,i}$ for user u and item i is computed as an aggregate of the historical preferences of other users for the same item i, which can be formulated as:

$$S_{u,i} = \underset{u' \in C_u}{\text{aggr}}\, S_{u',i} \tag{2.4}$$

where C_u is the set of M users that are the most similar to user u and who have historical preferences to item i, where $1 \leq M \leq$ the total number of users.

Another approach of memory-based algorithms, called the item-based approach, relies on the similarity between items to estimate unknown preferences. In this case, $S_{u,i}$ is computed as an aggregate of the historical preferences of the targeted users u for other items, which can be formulated as:

$$S_{u,i} = \underset{i' \in D_i}{\text{aggr}}\, S_{u,i'} \tag{2.5}$$

where D_i is the set of N items that are the most similar to item i with historical preferences by user u, where $1 \leq MN \leq$ the total number of items.

Model-based algorithms learn a model for each user or for each item using historical data, which can then make prediction directly without keeping track of previous data. It can be done by various Machine Learning and statistics techniques, such as clustering, artificial neural network and bayesian classifiers. Among the model-based algorithms in collaborative filtering, matrix factorization technique [43] is popularized by the Netflix competition. This technique transforms both items and users to the same latent factor space. The latent space is then used to make prediction by characterizing both users and items in term of factors automatically inferred from user feedback. Latent factor models is commonly induced by a lower rank Singular value decomposition (SVD) [44] of the prediction matrix.

Hybrid

Hybrid algorithms combine content-based filtering and collaborative filtering implementations. Literature [45] suggests that hybrid algorithms perform better than pure collaborative filtering algorithms and pure content-based algorithms in some situations.

2.5.1 Context-aware Recommender Systems

Introduction

Context-aware Recommender Systems (CARS) is a relatively new research area [38]. A traditional Recommender System tries to predict users' preferences towards items, which can be a rating, a probability or any numerical score, of user-item pairs that have not been observed before:

$$U \times I \rightarrow S \tag{2.6}$$

where U is the user feature matrix, I is the product feature matrix and S is the matrix of estimation.

This model relies solely on historical interaction between consumers and products. It neglects any additional information that may be useful for the estimation. Context-aware Recommender Systems (CARS), on the other hand, take contextual information into account for the prediction. The general model of Context-aware Recommender Systems can be written as:

$$U \times I \times C \rightarrow S \tag{2.7}$$

where C is the context matrix.

What is Context

The word "context" has been used by many research disciplines, such as computer science, psychology and linguistics etc. Sometimes, its meaning is defined differently. Reference [46] describes 150 different definitions of context and conclude that there is no unifying definition for the term. Survey [38] also concluded that the definition of context in Recommender Systems is still open to further studies and discussion. In Recommender Systems, different type of context can possibly be relevant to the prediction, such as the context of the user, the context of the item, the context of the behaviour or the interaction and the context of the situation.

A comprehensive view was introduced by Dourish [47], which classify context into the representational view and the interactional view.

Representational View

Most existing literature in Context-aware Recommender Systems focused on context of the representational view. Context is defined as a predefined set of static observable attributes. According to the definition of [47], the representational view must fulfill four assumptions: 1) context is a form of information; 2) context is delineable; 3) context is stable; 4) context and activity are separable.

Interactional View

The interactional view [48, 39] is the opposite of the representational view. The scope of contextual information is defined dynamically, and it is occasional rather than static. A cyclical relationship between context and activity is assumed, where the activity gives rise to context and the context influences activity.

Integrating Context into Recommender Systems

Existing approaches to integrate context into Recommender Systems, such as [39, 49], can be generalized into three categories: contextual pre-filtering, contextual post-filtering and contextual modeling. They all take the representational view of context.

Contextual prefiltering

First, historical data is selected based on the current context. Prediction is then made using any traditional prediction model according to the selected data only.

Contextual postfiltering

Contextual information is initially ignored. Prediction is made as usual using any traditional prediction model according to all available historical data of user preferences. Then, the resulting set of recommendations is re-ranked or adjusted for each user using the contextual information.

Contextual modeling

Contextual information is used directly as part of the prediction model.

2.5.2 Context-aware Recommender Systems for Retail

The emerging research in Context-aware Recommender Systems provide a solid ground for further development on incorporating additional information into traditional prediction models to improve accuracy. Existing state-of-the-art research faces some limitations. First, the representational view of context cannot fully represent the reality of contextual information in retail businesses. The business environment is changing rapidly, consumer preferences are dynamics. The interactional view that assumes contextual information to be dynamic and occasional is much more realistic. Second, similar to many Machine Learning techniques, current Context-aware Recommender Systems techniques require manual expert decisions on weighting the importance of each type of contextual information on the prediction model. This process is time consuming and challenging. There is space for improvement for the methods, such as correlation-based filter, to automatically integrate contextual information into the model, especially for contextual information that is dynamic.

2.5.3 Prediction with Matrix Factorization

The prediction model of Recommender Systems can be implemented by a number of Machine Learning and statistics techniques. For instance, the problem can be formulated using the matrix factorization technique [50]. For a retailer with M consumers and N products, I represent the purchase history with $\boldsymbol{s}$, a $M \times N$ matrix where $s_{mn} = 1$ if consumer m has purchased product n in the past, the value is *unknown* otherwise. Please note that it is a sparse matrix with a lot of unknown values. This is a prediction problem to replace an unknown value in $\boldsymbol{s}$ with an estimation of the probability of purchase given a consumer and a product, i.e. $p(purchase = 1|u, i)$. The higher the

probability, the more likely the consumer will purchase the product.

Matrix factorization is a popular technique in Recommender Systems research which transforms both items and users to the same latent factor space. The latent space is then used to explain prediction by characterizing both consumers and products in term of factors automatically inferred from user feedback. Latent factor model is usually induced by an SVD-like lower rank decomposition of the prediction matrix.

We set the number of features as k. We define $\boldsymbol{p}$ as a $k \times M$ matrix representing factors of k features of M consumers. We also define $\boldsymbol{q}$ as a $k \times N$ matrix representing factors of k features of N products. $\boldsymbol{e}$ is a $M \times N$ matrix of experimental errors which I will further explain below. The basic latent factor model for user-item ranking score can be written as:

$$\boldsymbol{s} = \boldsymbol{p^T q} + \boldsymbol{e}$$

where

$$\boldsymbol{s} = \begin{bmatrix} s_{11} & s_{12} & \dots & s_{1N} \\ s_{21} & \ddots & & s_{2N} \\ \vdots & & \ddots & \vdots \\ s_{M1} & s_{M2} & \dots & s_{MN} \end{bmatrix}, \boldsymbol{p^T} = \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1k} \\ p_{21} & \ddots & & p_{2k} \\ \vdots & & \ddots & \vdots \\ p_{M1} & p_{M2} & \dots & p_{Mk} \end{bmatrix}$$

$$\boldsymbol{q} = \begin{bmatrix} q_{11} & q_{12} & \dots & q_{1N} \\ q_{21} & \ddots & & q_{2N} \\ \vdots & & \ddots & \vdots \\ q_{k1} & q_{k2} & \dots & q_{kN} \end{bmatrix}, \boldsymbol{e} = \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1N} \\ e_{21} & \ddots & & e_{2N} \\ \vdots & & \ddots & \vdots \\ e_{M1} & e_{M2} & \dots & e_{MN} \end{bmatrix}$$

Estimation (or prediction) of the probability of purchase of a consumer to a product is calculated by the rule: $s_{u,i} = \boldsymbol{p_u}^T \boldsymbol{q_i}$. Standard SVD decomposition techniques [51] are unable to handle matrix with missing data. One way to decompose the purchase history matrix $\boldsymbol{s}$ to $\boldsymbol{p}$ and $\boldsymbol{q}$ is to first impute missing data with an initial value, such as the average probability of purchase or even zero. Another way is to use advanced optimisation techniques, such as Alternating Least Squares [52], to obtain $\boldsymbol{p}$ and $\boldsymbol{q}$ directly using only available data. In this case where $s_{u,i}$ contains only implicit data, i.e. consumer purchase behaviour, the technique [35] that transforms implicit customer observations into two paired magnitudes: preferences and confidence levels is commonly applied. With this technique, for each customer-product pair, the system

derives from the input data an estimate to whether the consumer would purchase or not purchase the product (preference) and couple this estimate with a confidence level.

2.5.4 Context-aware Prediction Model

Traditional prediction models, such as the matrix factorization model mentioned above, do not take contextual information into consideration. Our research hypothesis is that by incorporating contextual information of retail business into the prediction model, the accuracy of the purchase prediction can be improved. Here I use the matrix factorization technique for the illustration again. The prediction model combined with contextual information can be written as:

$$s_{u,i,c_1....c_j} = \boldsymbol{p_u}^{\boldsymbol{T}} \boldsymbol{q_i} + \omega_1 b_1(c_1) + ... + \omega_j b_j(c_j) + \boldsymbol{e_{u,i}}$$

where j is the number of contextual information I integrate to the model. $b_1....b_j$ are j contextual bias functions for contextual inputs $c1....cj$ respectively which return the increase or decrease of probability of purchase given the context. $\omega_1....\omega_j$ are the weighting parameters for these context.

We can formulate it as an optimisation problem:

$$\underset{\boldsymbol{p},\boldsymbol{q},\omega_1....\omega_j}{\operatorname{argmin}} \sum_{(u,i,c_1....c_j \in R)} \left(s_{u,i,c_1....c_j} - \boldsymbol{p_u}^{\boldsymbol{T}} \boldsymbol{q_i} - \omega_1 b_1(c_1) - .... - \omega_j b_j(c_j)\right)^2$$

Please note that the matrix factorization model $\boldsymbol{p_u}^{\boldsymbol{T}} \boldsymbol{q_i}$ can be replaced by any other prediction models.

Chapter 3

Correlation between Contextual Information and Purchase Behaviours

This chapter introduces a computationally inexpensive approach to conduct preliminary relevance analysis for contextual information in retail business. Odds ratio, a statistical technique from the medical research field, is applied to measure the relationship between consumers' online exposure to retailer's e-commerce website i.e. a contextual factor, and their offline in-store purchase decisions, i.e., the outcome to be predicted based on a retail dataset provided by a large UK retail business, Boots UK, with both online and offline presence. Unlike machine learning approaches, this analysis can be done even before a recommender system is built by using the proposed approach. This research further shows that the relevance of a contextual factor depends on extraneous attributes, such as consumers' ages and gender. This chapter serves as a preliminary step to analyze relevant contextual factors for building context-aware recommender systems.

3.1 Background

3.1.1 Motivation

Recommender systems are increasingly important for retail businesses. Retailers commonly provide personalized product recommendations to consumers through various channels, for instance, web advertisement, email vouchers advertisement and in-store location-based mobile advertisement. Recommender systems try to predict the out-

comes accurately so that they can recommend products according to the best expected scenario. Research shows that the use of proper contextual information in recommender systems can improve the accuracy of prediction in some situations [49].

For instance, the use of purchase intent as a contextual factor can improve prediction accuracy [53]. The challenge is whether retailers can obtain such contextual information of a consumer who just enters a retail store without requiring additional user interaction. In an offline in-store environment, other than time and location context, the type of user context retailers can obtain without intruding consumers' experience is limited. To address this issue, this chapter explores the possibility of using consumers' recent online behaviours on the retailer's e-commerce website to derive contextual factors for their in-store purchase decisions. The type of consumer behaviour that can be collected non-intrusively online is usually richer than that which could be collected in-store.

In this chapter, I consider consumers' recent online exposure to brands at a retailer's website as a potential, but controversial, contextual factor. Consumers' online exposure could imply consumers' tendency to purchase products online, so they are less likely to purchase them in-store. Oppositely, it could be seen as a user context that represents consumers' recent preference on brands, which implies a higher chance of in-store purchase. The understanding of the relevance of this contextual factor is important for a number of recommendation scenarios. For simplicity, I focus on a specific scenario: A large retailer that has both an online e-commerce website and an offline store presence wants to predict those product brands the customers are going to purchase when they enter stores. For a brand that a targeted consumer has browsed online recently, a recommender system can possibly consider three cases: 1) If the relevance is strongly positive, the consumer will likely to purchase products of this brand in-store anyway, so no recommendation is needed; 2) If the relevance is negative or if there is no relevance, the chance of purchase in-store is not high; 3) If the relevance is mildly positive , the system may try to nudge the consumer to purchase products of this brand in-store.

From the point of view of utilizing contextual information in recommender systems, this chapter investigates the relevance between consumers' online exposure to brands at the retailer's website i.e. a contextual factor, and the probabilities of their

in-store purchasing decisions i.e. the outcome to be predicted, for product brands, stratified by different consumer groups. I propose the use of statistical techniques, namely Odds Ratio [54, 55, 56] , Mantel-Haenszel (MH) method [57] and Stratified Analysis [58], to analyze contextual factor. Unlike machine learning techniques, this approach is independent from the prediction model of the system. Besides, unlike traditional correlation analysis techniques, such as sign test and chi-squared test, that can only evaluate the relevance of a contextual factor, my approach can further estimate the influence of the contextual factor on the probability of in-store purchase on products of brands. This information can possibly be used to improve the prediction model directly in my future work. A challenge of using statistical techniques, namely the issue that basic probabilistic measurement is sensitive to external noise, is presented by a numerical example in this chapter. The dataset for experiments used in this research is provided by Boots UK - a large UK retail business. Ten product brands are analyzed. This experiment makes use of the first year data of a 2-year anonymised record of loyalty card holders who have browsed products of the selected brands online at the retailer's website and of those who have purchased products of the selected brands in any store of the retailer in the UK. All data is collected in a real non-experimental setting. In my experiments, whether the consumers have browsed any product page of a targeted brand at the retailer's website within a month is regarded as a binary contextual factor and the outcome to be predicted is consumers' in-store purchase decisions. Results show that the relevance of this contextual factor on the outcome to be predicted varies with consumers' attributes such as age and gender. The rest of the chapter is organized as follows. Section 3.1.2 is a review of related literature about techniques that identify relevant contextual information in recommender systems. Section 3.2 describes the challenges of data analysis in a retail scenario. Solutions are then proposed to analyze contextual factors. Section 3.4 presents experiments conducted based on real retail dataset and section 3.5 is the assessment of the work.

3.1.2 Related Work

The literature review of this chapter focuses on techniques that identify and evaluate relevant contextual information in recommender systems. The technical goal of a recommender system can be seen as the problem of predicting ratings, or any other outcomes,

for the items that have not been seen by a user [7]. The outcome to be predicted in retail recommender systems, for instance, may be consumers' purchase decisions instead of ratings. The use of contextual information in recommender systems improves the accuracy of prediction in some situations [49]. Context is a multifaceted concept that is defined differently in multiple research disciplines [38]. Various kind of attributes can be defined as context. For instance, there are context of users, context of items and context of interactions or situations [46]. Regardless of the definition, the selection of relevant contextual factors to be used in recommender systems is a critical issue. To deal with this issue, some literature applies machine learning techniques to identify relevant factors automatically. Decision trees and feature selection techniques are used to rank the relevance of user preferences and system settings in a news recommender system for accurate recommendations [59]. Another feature selection technique, Las Vegas Filter algorithm, has been applied in a more recent work to identify relevant factors [60]. A pre-filtering algorithm that pre-processes and selects contextual segments offline has also been described [38]. In [61], users are clustered based on the value of some contextual factors. The predictive accuracy of each cluster is then compared with the one of the whole dataset which is non-contextual in order to understand whether and where performance has been improved. The advantage of this type of algorithm is that factors are considered only in situations where contextual method outperforms the standard non-contextual algorithm. In current literature, relevance of contextual factors is measured based on their effects in the system's predictive accuracy. Recent research, however, shows that recommendation accuracy of context-aware recommender systems can be affected by conditions other than the contextual factors themselves, such as the task requirement and the overall number of items in the recommended list [62]. As a result, contextual factors may be omitted simply because they are not integrated into the system or the prediction model properly.

Another approach to evaluate the relevance of a contextual factor is to measure the dependence between the factor and the value to be predicted statistically. There are various statistical methods to measure the dependence between random variables, such as covariance, correlation and odd ratios. In contrast to a machine learning approach, a statistical approach to evaluate the relevance of a contextual factor is fast to compute and is independent from the prediction model implemented by the system. Not all

type of data fulfills the assumptions of these statistical models though. For instance, Pearson Correlation Coefficient, or its binary form, Phi Coefficient, expects a linear relationship between the two variables. Paired t-test discussed in previous literature [49] is not suitable for binary data with binomial distribution. They are, therefore, not suitable for my scenario. Although other statistical methods, such as Sign Test and Chi-squared Test, could be suitable for binary data, this chapter presents an alternative statistical methodology that, not only identify the relevance of a contextual factor, but also estimates the strength of the relevance as a probability score. By representing the relevance in a probability score, it is possible to make use of this contextual information to improve the prediction models directly in my future work.

3.2 Research Challenge

Some retailers aim to improve the accuracy of customer preference prediction in Recommender Systems. In retail and many other business environments, extra contextual factors are sometimes available. As mentioned in the previous section, the use of some contextual factors can improve the accuracy of prediction in some situations. The relevance of some factors, however, is controversial in the industry. For instance, consumers' recent online exposure to products can decrease the chance of in-store purchase as consumers may choose to purchase products online. On the other hand, online exposure can be seen as an evidence of consumers' preference on products, which implies a higher chance of in-store purchase. The understanding of true relevance is important for product recommendation in-store in this case.

The research challenge is how to evaluate the relevance of potential factors for prediction.

3.3 Statistical Analysis

3.3.1 Problem Formulation

In this section, I consider a retailer that operates both an online e-commerce website and physical retail stores. Consumers' recent online exposure at the retailer's website is the contextual factor to be evaluated. In particular, I define whether the consumers have browsed at least a page of a targeted brand at the retailer's website within a month as the contextual condition, $browse = 1$ if the condition exists, 0 otherwise. For illustration

purposes, I assume that the outcome to be predicted is the purchase decision of any product of the targeted brand at any physical store (in-store purchase), which is a binary variable: $purchase = 1$ if purchased, 0 otherwise. In order to evaluate the relevance between this contextual factor and the outcome, I need to compare the probability of in-store purchase of consumers who have browsed and of those who have not, i.e. $p(purchase = 1|browse = 1)$ and $p(purchase = 1|browse = 0)$. In a population of N potential consumers, I construct a table to represent the online browsing and in-store purchasing situation of the dataset:

	purchase=1	**purchase=0**	
browse=1	a_{11}	a_{10}	a_{1*}
browse=0	a_{01}	a_{00}	a_{0*}
	a_{*1}	a_{*0}	N

where a_{11},a_{10},a_{01} and a_{00} are the number of consumers for the corresponding purchasing and browsing situations. $a_{*1} = a_{11} + a_{01}$ is the number of consumers who have purchased in-store, $a_{*0} = a_{10} + a_{00}$ is the number of consumers who have not purchased in-store, $a_{1*} = a_{11} + a_{10}$ is the number of consumers who have browsed online and $a_{0*} = a_{01} + a_{00}$ is the number of consumers who have not browsed online. A direct way to express the relationship is to compare the two probabilities with *relative correlation* (RC), where

$$RC = \frac{p(purchase = 1|browse = 1)}{p(purchase = 1|browse = 0)} = \frac{a_{11}/a_{1*}}{a_{01}/a_{0*}} \tag{3.1}$$

There is no correlation if $RC = 1$, the relevance is positive if $RC > 1$ and negative if $RC < 1$. This approach suffers from two problems when the data is collected from a non-experimental retail environment. First, RC is sensitive to the total number of consumers who have purchased and also to the total number of consumers who have not purchased. These two numbers can be affected by external irrelevant factors, such as marketing campaigns that should be isolated from this analysis. This problem can be illustrated with a numerical example. Suppose the data looks like the following table when there is no sales promotion:

Let us assume that a sales promotion successfully attracts new consumers to purchase and the number of purchase increases 10 times as shown in the following table:

	purchase=1	**purchase=0**	
browse=1	5	500	505
browse=0	60	25,000	25,060
	65	25,500	25,565

Table 3.1: Sample data before sales promotion

	purchase=1	**purchase=0**	
browse=1	50	500	550
browse=0	600	25,000	25,600
	650	25,500	26,150

Table 3.2: Sample data after sales promotion

When all things being equal, a temporary sales promotion that affects the purchase of customers regardless whether they have browsed online or not should not affect the relationship between the contextual factor and the outcome. In reality, however, $RC = \frac{5/505}{60/25060} = 4.14$ in the first case while $RC = \frac{50/550}{600/25600} = 3.88$ in the second one. In another words, RC is sensitive to the change of number of consumers who purchase (a_{*1}). This problem exists in many real-world environments since businesses may attract new consumers to stores or website dynamically, which affects N, and thus a_{*1} and a_{*0} can be manipulated. An odds ratio technique to estimate RC that is insensitive to the change of N is proposed later in this chapter. The second problem is the existence of extraneous attributes, such as age and gender. They may affect the relevance of the targeted contextual factor on the outcome to be predicted. This problem occurs when an attribute is associated with the contextual factor and at the same time such attribute affects the outcome dependently or independently. This kind of extraneous attribute is called a *confounder* in the statistics discipline. Stratified analysis is proposed to evaluate the impact of possible confounding attributes.

3.3.2 Odds Ratio

Odds ratio is commonly used as an alternative of RC in medical and epidemiological research for case-control studies where disease cases are not easy to be obtained [54, 55, 56]. Similar to the current problem, N is also adjustable in medical studies because

the number of people with and without diseases in the dataset are determined by the design of the case-control studies artificially. In this case, OR can be calculated as:

$$OR = \frac{a_{11}/a_{01}}{a_{10}/a_{00}} = \frac{a_{11}a_{00}}{a_{10}a_{01}} \tag{3.2}$$

Identical to RC, there is no correlation if $OR = 1$, the relevance is positive if $OR > 1$ and negative if $OR < 1$. Unlike RC, OR is insensitive to the row and column scaling operations of the data table. Using the same example above, $OR = \frac{5X25000}{60X500} = 4.17$ when there is no sales promotion, $OR = \frac{50X25000}{600X500} = 4.17$ as well when there is sales promotion. OR is a good alternative statistically if a condition is fulfilled. The condition is that, for the two groups of consumers i.e. those who have browsed online and those who have not browsed online, the number of consumers who have purchased in-store must be a small percentage (less than 10%) of the total number of consumers in the group. This condition is reasonably fulfilled in most retail situations. Confidence interval (CI) is used to determine the reliability of the results. The larger the range of CI, the less reliable the result is. The CI of odds ratio [63] can be approximated with:

$$CI = \frac{a_{11}a_{00}}{a_{10}a_{01}} \exp\left(\pm z\sqrt{\frac{1}{a_{11}} + \frac{1}{a_{10}} + \frac{1}{a_{01}} + \frac{1}{a_{11}}}\right) \tag{3.3}$$

where z is the score of the standard normal distribution associated with the confidence level. $z = 1.96$ for a 95% confidence interval.

3.3.3 Stratified Analysis

Extraneous attributes, such as consumers' age and gender, potentially affect the relevance of the contextual factor on the outcome. Stratified analysis is a computationally inexpensive solution to reveal their effects. This technique is commonly used in medical research when setting up control group experiments is not feasible and so the existence of extraneous factors is common [58]. It analyzes subgroups (strata) of the study population separately according to the attributes. For instance, two strata are created for the gender attribute: female consumers and male consumers. Odds rate is measured for each strata separately. Stratified analysis provides an independent view for each strata each with its own odds ratio. The difference is then comparable among these strata. In addition, a common strata-adjusted odds ratio is estimated by Mantel-Haenszel (MH) method [57]. This adjusted value represents a weighted average of

the stratum-specific odds ratio which is an approximation to the maximum likelihood estimation. According to [64], the formula of approximation can be written as:

$$OR_{MH} = \frac{\sum_{i=1}^{k} \frac{a_{11i}a_{00i}}{N_i}}{\sum_{i=1}^{k} \frac{a_{01i}a_{10i}}{N_i}} \tag{3.4}$$

where k is the total number of strata in an analysis and i represents one of them. For this Mantel-Haenszel method of estimation to be accurate, the overall sample size must be large. [65] provides a more robust but complicated approximation method for data with small sample size. Confidence interval (CI) can again be used to indicate the reliability of the result:

$$95\% \text{ CI for } OR_{MH} = Exp[(lnOR_{MH} \pm SE(lnOR_{MH})] \tag{3.5}$$

where

$$SE(lnOR_{MH}) = \sqrt{(\frac{\sum_{i=1}^{k}(\frac{a_{10i}a_{01i}}{N_i})^2 v_i}{\sum_{i=1}^{k}(\frac{a_{10i}a_{01i}}{N_i})^2})}$$

and

$$v_i = \frac{1}{a_{11i}} + \frac{1}{a_{10i}} + \frac{1}{a_{01i}} + \frac{1}{a_{00i}}$$

3.4 Experiment

3.4.1 Dataset

The dataset, which is provided by Boots UK, is the first year data of a 2-year anonymised records of loyalty card holders who have browsed the selected products online on the retailer's website and of those who have purchased the selected products in any store of the retailer in the UK. It contains 10,217,972 unique loyalty card holders and 2,939 unique products under 10 selected brands. There are 21,668,137 in-store purchase transaction records and 299,070 online browsing records. I associate consumers' online browsing and in-store purchasing behaviours with unique loyalty card numbers. All the data is collected in a real non-experimental setting.

3.4.2 Experimental Design

This experiment investigates the relevance between consumers' recent online browsing behaviours and the probabilities of their in-store purchase decisions for ten product brands carried by a large UK retail business nationally. These ten brands were selected

randomly, some of them are luxury brands while the others are mid-range brands. I define whether a consumer has browsed at least a page of a targeted brand at the retailer's website within a month as the context of the consumer, $browse = 1$ if the condition exists, 0 otherwise. Odds ratio is used to compare the relevance of this contextual factor on the probabilities of consumers' binary purchase decision of any product of the targeted brand at any physical store (in-store purchase). I pre-process the dataset to filter out consumers who have not visited any page at the retailer's website at least once in the past year. This process ensures that the remaining N consumers have at least successfully accessed the retailer's website recently. I start with a hypothesis that age and gender are two attributes of consumers that may confound the relevance. I conduct monthly strata-specific measurement of odds ratio based on these two attributes for each brand. Practically, age and gender information is missing in *some* records. In each analysis, therefore, I analyze a population size of N_{age} or N_{gender} which represent the total number of consumers *with* age information or *with* gender information respectively. In these experiments, I calculate the monthly crude (unadjusted) odds ratio for each strata for each brand. If the odds ratio for a strata of a brand is X, it means that, in this strata and in this particular month, the probability to purchase at least one product of this brand in-store by consumers who have browsed at least a webpage of this brand online is X times higher than the probability for those who have not browsed so. I also calculate the monthly common strata-adjusted odds ratio as well as the 95% confidence interval (CI).

3.4.3 Results

Stratified odds ratios analysis of six product brands have been conducted. All figures show that odds ratio measurements are well above 1, i.e., the relevance is positive for all brands. It means that the probability to purchase at least one product of a selected brand in-store by consumers who have browsed at least a webpage of that brand online at the retailer's website is higher than the probability for those who have not browsed so. The values and patterns are different for each brand though, which means that the impact of this contextual factor of online exposure varies with brands.

Figure 3.1 represents gender-stratified analysis of brand A. The relevance between recent online browsing behaviours and in-store purchase behaviours on female

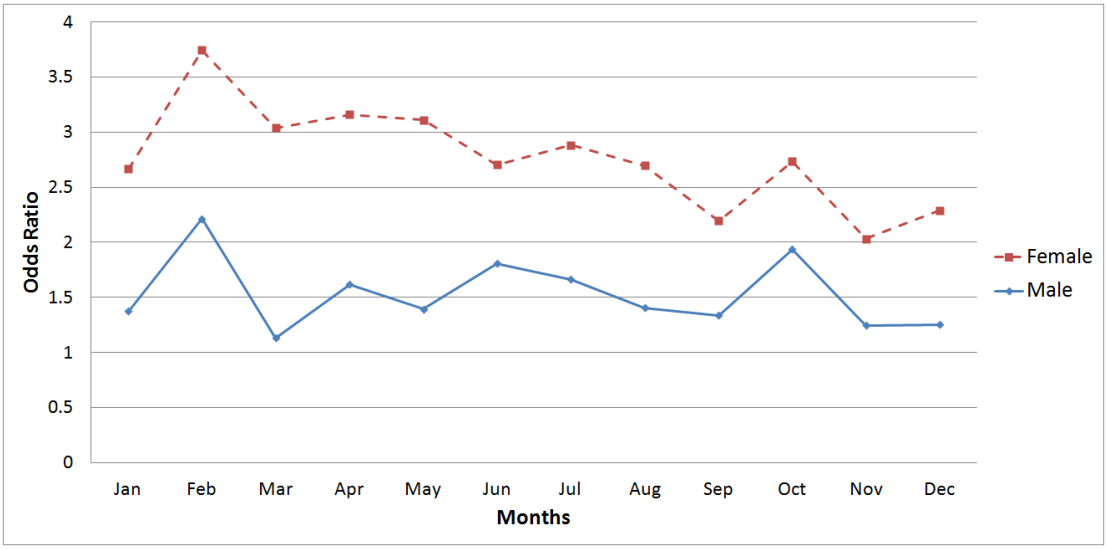

Figure 3.1: Odds Ratio by Gender (Brand A)

consumers is much stronger than the one on male consumers. An interesting discovery is that the odds ratio measurements for both genders follow a very similar up and down monthly pattern. Both strata have peaked odds ratio in February in the data year period. This suggests that time is a contextual factor that should also be considered in future work. The 95% confidence interval values of Odd Ratios of brand A is also shown in Figure B.19 as an example. Figure 3.2 shows that the odds ratio range of different age groups for brand B are separated clearly. The relevance for consumers of age 18-25 is the highest while the one for consumers of age 26-35 is the lowest. It means that the probability for consumers of age 18-25 is higher than the one for consumers of age 36-45 and both of them are higher than the one for consumers of age 26-35. This observation suggests that, for brand B, the age attribute itself can be correlated to consumers' in-store purchase decisions. Figure 3.3, on the other hand, draws a different conclusion for brand A. In this case, the odds ratio measurements of these age groups mixed together in a close range. There is no clear monthly pattern either. It means that age, gender and month are not confounding factors for this brand.

Appendix 2 contains a complete list of all figures.

3.5 Assessment

This chapter derives a contextual factor from consumers' recent online browsing behaviours on the retailers' website for the prediction of their offline in-store purchase.

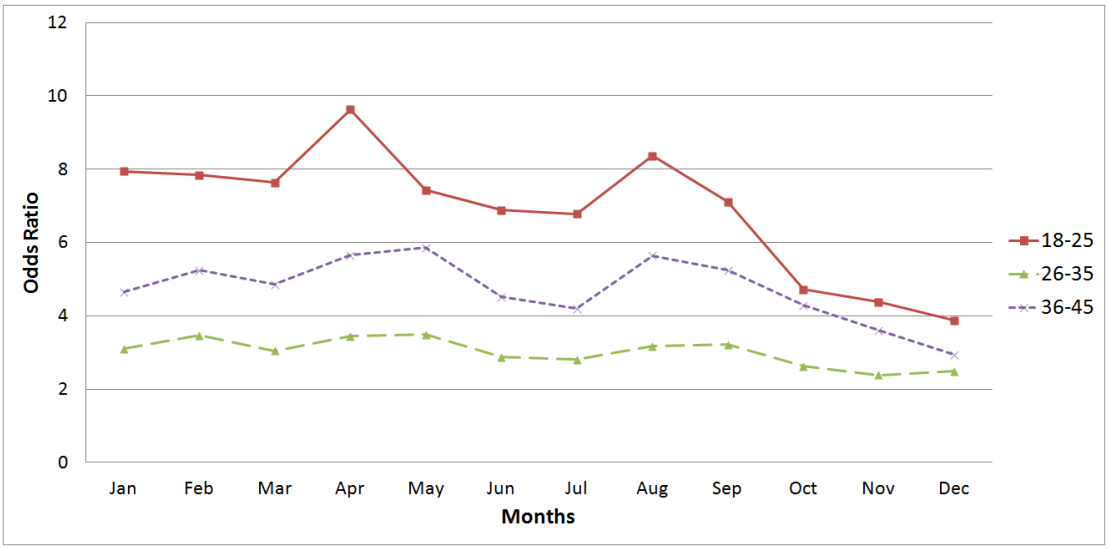

Figure 3.2: Odds Ratio by Age (Brand B)

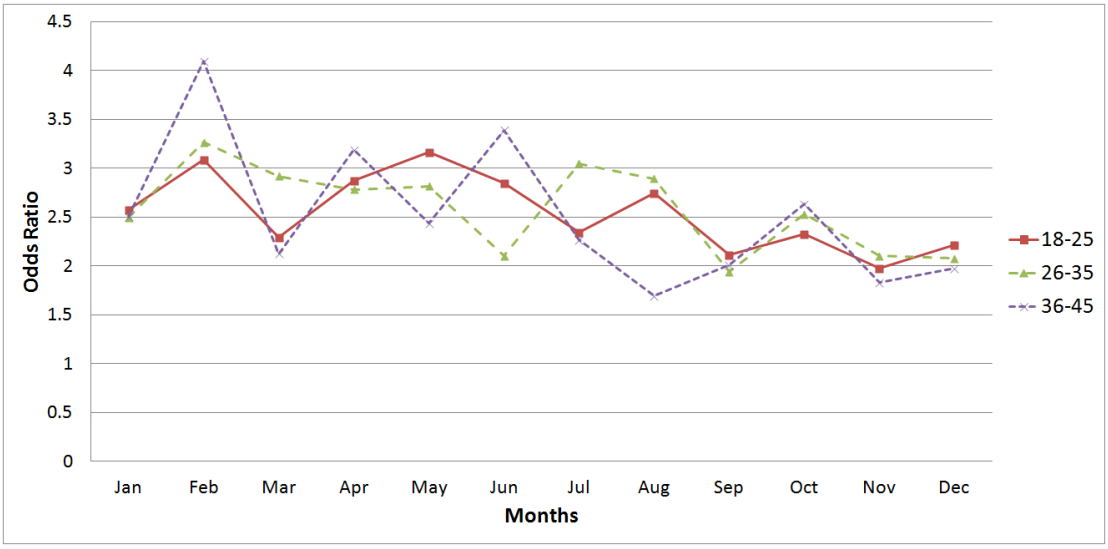

Figure 3.3: Odds Ratio by Age (Brand A)

A statistical approach is presented to conduct a preliminary analysis on the relevance between this factor and the offline purchase decisions on brands using odds ratio and stratified analysis techniques. The initial uncertainty that consumers who browse online on retailer's website tend to purchase online and therefore they have lower chance to purchase in-store has been proven untrue for the brands I have analyzed. In addition, as expected, the relevance of online exposure on offline purchases varies with brands and consumers' ages and gender. In my future work, the analysis for non-binary contextual factor will be illustrated. Besides the factors I have evaluated in this chapter, it is interesting to see whether other relevant contextual factors can be derived from consumers' recent online behaviours for their in-store purchase decisions. Future work is to build a context-aware recommender system for in-store product recommendation based on these findings. OR value can potentially be used to improve the accuracy of prediction. Also, a comparison of predictive performance of recommender systems using contextual factors selected by this approach and by existing machine learning techniques is part of my future work.

Chapter 4

Exploration and Exploitation for New Products

In this chapter, I investigate a resource allocation problem in recommendation systems where unknown new items keep coming to the system at different time. The task is to recommend (allocate) each user a limited number of new items. The objective is to maximise the overall positive response rate. I propose a two-stage batch solution to approximately optimise the objective, using group buying as a working example. My experiments indicate that the proposed approach significantly improves the positive response rate.

4.1 Background

4.1.1 Motivation

Product cold-start problem [66], which means that the system has not gathered sufficient information about a new product to make prediction, is very common in the retail industry. I investigate the situation where new products that are unknown to the system keep being available to customers. The task is to recommend (allocate) each user a limited number of new products. The objective is to maximise the overall positive response rate. This problem is non-trivial because, on one hand, retailers need to allocate these news items to users who are helpful for learning new item feature profiles using the limited recommendation opportunities (resources) in order to improve prediction accuracy; on the other hand, allocate these items to users who would most likely purchase

them on the basis of the new item information gathered so far in order to maximise positive response rate. In this chapter, I propose a two-stage batch solution to approximately optimise the objective, using group buying as a working example. During the first stage, I estimate the user purchase decisions towards new items by allocating some resources for exploration. During the second stage, I optimally allocate the remaining resources for exploitation according to the prediction and the operational constraints using the binary integer programming technique. I also want to evaluate the effectiveness of one of the traditional experimental design techniques in improving the learning efficiency during the exploration process.

The chapter is organised as follows. I discuss the related work in Section 4.1.2. I then present the problem formulation and its solution in Section 4.3. I present empirical investigation in Section 4.4 and finally conclude in Section 4.5.

4.1.2 Related Work

The research of this chapter is considered as part of collaborative filtering research [67, 50]. The prediction challenge for new users or new items is commonly known as the cold-start problem [68]. Content-based approaches and active learning approaches are the two main methods to address the cold-start problem. Content-based approach [69, 70] utilizes the known attributes of new users or new items to make prediction. There are also hybrid methods that combine content-based technique with collaborative filtering technique [71]. In this case, however, I assume there is no prior knowledge about the attributes of the new items, and they are represented by merely unique IDs. The benefit of having this assumption is that my model can be generalized for any type of items, such as non-text images and video or items with very little descriptive texts. Another reason is that new products of group buying systems may belong to different domains and their attributes can be very different and difficult to match up. Active learning approaches select a subset of user-item pairs, known as queries, and collect feedback from them. These feedback will be used as training data for new items or new users. Although Active Learning techniques, such as D-Optimal design and its variations [72], have been proven effective to improve prediction accuracy in CF by some literatures [73, 74] previously, these efforts are limited to enhancing the exploration stage. The balance between exploration and exploitation is not covered. Furthermore,

many techniques [75, 76] assume an iterative interview process for sequential query selection i.e. only one query is chosen at a time for one user, after the response is collected, another query will be chosen according to the user response of the previous query. In this case, as well as many other practical situations, exploration of multiple new items for multiple users has to be done together in one round. Therefore, iterative techniques are not applicable.

Recently, a social networks approach to address the cold-start problem has been studied [77]. This approach automatically select a set of users who are mentors or leaders in the network to rate the new items in the early stage. The drawback of this approach, especially when retailers have a limited quota of resources (recommendation) for each user, is that the learning of new items is heavily relied on a small pool of users. The recommendation quota of these users will run out very soon.

Despite the similarity in wordings, this chapter focus on recommendation for individuals, not groups. Group recommendation research concerns about the situation where the users of the recommender systems operate not individually but in groups [78]. It would be the case if the system has to recommend an item, or a same set of items, to a group of users. In my scenario, the system has the flexibility to recommend different items to each user individually.

4.2 Research Challenge

For the last two decades or so, information retrieval technologies have fundamentally transformed the way in which people seek and work with information. Roughly speaking, there are two types of information retrieval systems [79]; On one hand, there are *ad hoc* information retrieval, e.g., web search [80], which deals with a relatively fixed collection of information items (webpages, documents, images, product descriptions etc) and dynamically changing users information requests. On the other hand, information filtering systems, such as content recommender systems, address the situation where users information requests (as user profiles) stay rather static whilst new items keep arriving. Nevertheless, in either case a fundamental problem remains the same, which is to compute and find the *match* between the information items and users information requests [81].

Recently, a new type of information filtering application emerges in the Internet,

namely *group buying* [82]. Group buying, also called collective buying, makes use of the Internet to collectively find relevant buyers (users) and offers them with products and services at significantly reduced prices on the condition that a minimum number of buyers would make the purchase. Examples of such application include Groupon and LivingSocial. While its dynamical pricing mechanism is of interest to economists [82], it also raises a challenging information retrieval problem: Given new services or products keep coming to the system at different time, how to find the relevant users (or user types) who are likely to purchase them and consequently recommend them to those users *quickly* (normally within a day)? Since they are new items, although their content descriptions may be provided, retailers do not know the users preference about them. Estimating the match between the information requests (the specific interests from the potential buyers in this case) and those new items (information about services and products in this case) is non-trivial. It has an out-of-sample problem, known in collaborative filtering as the cold-start problem [68] and one of the common approaches is through *active learning* - ask potential buyers to provide preference judgments so as to get the prediction model as accurately as possible in a cost-effective way [83]. This is also known as Experimental Design in statistics [84]. Alternatively, one can find a solution by extracting content descriptions [71]. The problem I am solving is more complicated than that. Existing research on active learning approach for the cold-start problem mainly focus on the *new users problem*, but retailers are dealing with the *new items problem*. In the proposed case, there is a series of new items and only a limited number of items can be recommended to a single user at a time. For example, in a group buying case, retailers are only allowed to send every potential buyer a fixed number of recommendations per day. Thus, within this limited number of recommendation opportunities (resources), the question is how to balance the exploration (estimating the prediction model) and exploitation (utilizing the model) in order to maximise the overall effectiveness of the system, e.g., the overall positive response rate. The new challenge of *new items problem* can be illustrated by Fig. 4.1. Furthermore, there are some business operational constraints. For instance, a deal of an item would become valid only if the minimum purchase requirement is fulfilled.

In this chapter, I formulate the problem as the *recommendation resource allocation problem* and use group buying as my working example. Note that in literature [85]

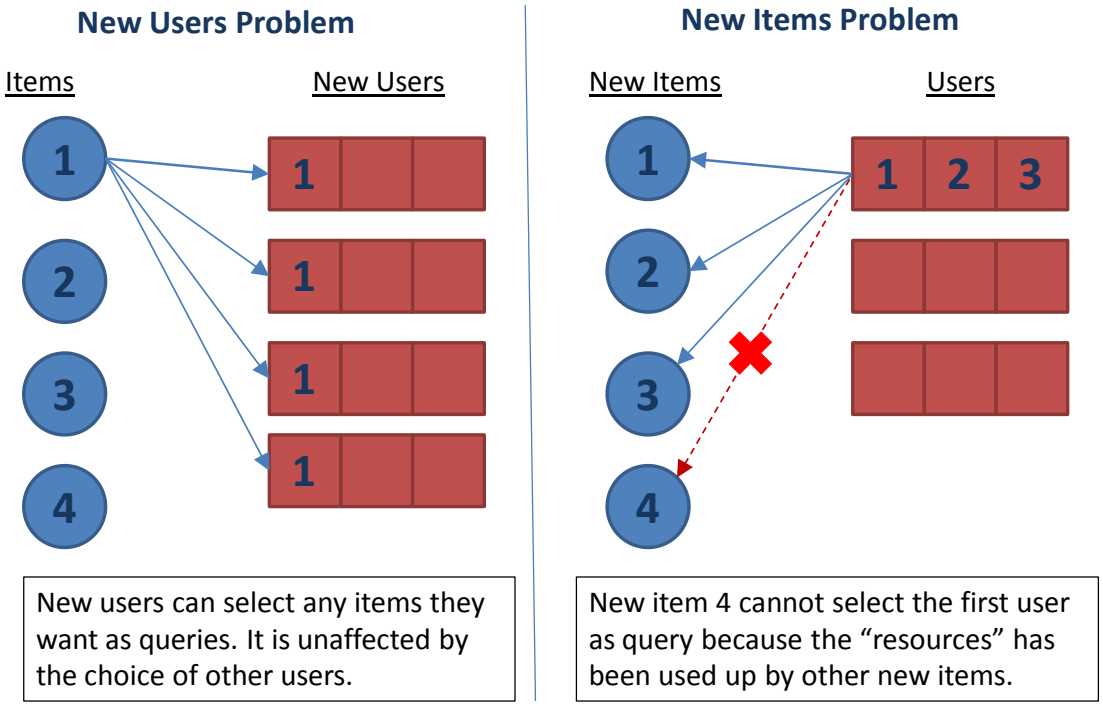

Figure 4.1: Difference between New Users Problem and New Items Problem

the usual analogy of classic statistical resource allocation problems is that of a casino with multiple one-armed bandit slot machines (the Multi-Armed Bandit Problem), each with a different probability distribution of rewards, and the problem is to find the best strategy for exploring the distributions of the bandits, whilst also exploiting the highest paying bandit [86]. The solution is usually sequential - try each arm (user) sequentially and obtain feedback one by one. This solution is not suitable because, in my settings, retailers need to make a decision in a short period of time. I thus propose a two-stage solution. During the initial exploration stage, I use a portion of resources to estimate the new item models. After that, during the exploitation stage, I recommend these items to users using the remaining resources. The recommendations are calculated by the prediction obtained from the exploration stage, and also by the binary integer programming technique [87] to optimise the objective function according to all the constraints. My simulation shows that the two-stage personalization method indeed has better performance than the one without personalization, which is commonly used in practice. Furthermore, I also found that unlike the conventional belief in active learning, the comparative evaluation of the D-optimal design [88] and the random selection approach on my specific dataset shows no statistically significant difference

between them when both approaches are at their own best exploration and exploitation ratio settings due to the resource constraints of the problem.

4.3 Recommendation Allocation

4.3.1 The formulation

Let us define there are M users in the system, whose response to a system's recommendation is denoted as binary random variable $R \in \{1,\ 0\}$, where $R = 1$ denotes a purchase action while $R = 0$ means otherwise. Up to now, the system has observed their purchase actions across N_0 number of the existing items. The definition of an item is a general one: it could be a piece of information, or a specific product or service. A user-item matrix can be used to specify the responses over the set of items. The matrix is extremely sparse and many of the elements are not observed. For a unobserved pair of user and item, I assign a probability $p(R = 1|u, i)$ to summarize the belief about how likely user u would like to purchase item i, where $i \in [1, N_0]$ and $u \in [1, M]$. The estimation of probability $p(R = 1|u, i)$, i.e. $\hat{p}(R = 1|u, i)$, can be estimated by using collaborative filtering (CF) models[50] through the correlations between users and/or between items, either implicitly or explicitly.

Suppose now there are N new items coming to the system. To make the discussion general, I assume there is no contextual information about them and they are represented by unique item IDs. The system is ready to recommend the N new items to M users. Suppose, the system is allowed to recommend K number of items to each user in the system. What is the optimal recommendation allocation strategy of the N new items so that a certain objective, e.g., the expected total number of positive responses (purchases) or the total profit gain, is maximised? There is no obvious solution because the system has no knowledge about the new items and thus cannot make personalized recommendation. Nonetheless, we should be able to learn the characteristics of the items by recommending them to some users for feedback collection. This is called *exploration* as the goal is to reach out certain types of users to build a personalization model as accurately as possible. Suppose, some user feedback has been obtained and the system has built the personalization model for the new items accordingly, the system should now be able to *exploit* the current model to recommend items

to the users who are likely to purchase them. However, the recommendation opportunities ($A = KM$) is limited. To maximise the overall positive response rate, the system needs to balance the use of these resources for exploration and exploitation. In fact, the trade-off between them has been well studied in the context of multi-armed bandit machine [86], but the multi-armed bandit machine formulation is mainly designed for the sequential learning case. The presented problem is different as the decision process of the system usually lasts in a very short period of time and the system cannot wait for the responses from users sequentially one by one. In this chapter, I propose a two-stage batch process where the system first simultaneously sends all new items to a subset of users in order to construct the recommendation model. Once users feedback are obtained, the system can estimate the item model and provide personalized recommendation for the remaining users based on prediction. Moreover, an extra challenge is that, in group buying systems, a purchase is successful only if the following operational constraints are fulfilled:

1. *minimum purchase requirement* A deal of an item i would become valid only if the minimum purchase requirement j_i i.e. the total number of purchase of the item, is fulfilled. If it is not fulfilled, all purchase orders of the item by users will become invalid.

2. *maximum available inventory* An item i may have a maximum number of availability l_i. No purchase can be made once an item is out of stock.

In short, the system needs to make three critical decisions: 1) Which users should it selects for exploration for each new item?; 2) How to balance the use of the limited resources for exploration and exploitation?; 3) How to optimise the recommendations given the operational constraints based on the purchase prediction?. Mathematically, for the exploration, the system wants to select b portion of resources A and construct a matrix $\boldsymbol{d} = [\boldsymbol{d}_1, ..., \boldsymbol{d}_i, ..., \boldsymbol{d}_M]$, which is a $M \times N$ binary system decision matrix where the user-item pair binary random variable d_{ui} is 1 when the system recommends item i to user u, otherwise it is 0. Similarly, I denote $\boldsymbol{d'}$ as the binary system decision matrix for the exploitation.

One possible way to formulate a real world scenario of the allocation of resources as an optimisation problem with some sample constraints is as follows:

The system sends recommendations in two stages. In stage 1, the system selects a portion of the resources, i.e. Ab, to learn the personalization model of new items where the parameter is denoted as $\hat{\theta}$. Before this stage, the best guess of the purchase decision is non-personalized, i.e. $\hat{p}(R=1|u,i) \equiv \hat{p}(R=1|u)$, which is obtained from the historical data of u. In stage 2, the system uses the resulting item model from stage 1 and the remaining $A(1-b)$ resources to make prediction $\hat{p}(R=1|u,i;\hat{\theta})$, which is controlled by the decision matrix $\boldsymbol{d'}$.

$$\underset{\boldsymbol{d},\boldsymbol{d'},b,\hat{\theta}}{\operatorname{argmax}} \quad \sum_{u=1}^{M}\sum_{i=1}^{N}\hat{p}(R=1|u)d_{ui} + \sum_{u=1}^{M}\sum_{i=1}^{N}\hat{p}(R=1|u,i;\hat{\theta})d'_{ui}$$

subject to:

minimum purchase:

$$\sum_{u=1}^{M}(\hat{p}(R=1|u)d_{ui} + \hat{p}(R=1|u,i)d'_{ui}) \geq j_i, \text{ for all } i = 1 \dots N$$

maximum inventory:

$$\sum_{u=1}^{M}(\hat{p}(R=1|u)d_{ui} + \hat{p}(R=1|u,i)d'_{ui}) \leq l_i, \text{ for all } i = 1 \dots N$$

maximum resources limit:

$$\sum_{i=1}^{N}(d_{ui} + d'_{ui}) \leq K, \text{ for all } u = 1 \dots M \tag{4.1}$$

exploration vs. exploitation split:

$$\sum_{u=1}^{M}\sum_{i=1}^{N} d_{ui} = Ab, \text{ and } \sum_{u=1}^{M}\sum_{i=1}^{N} d'_{ui} = A(1-b)$$

no repeat recommendation:

$$d_{ui} + d'_{ui} \leq 1 \text{ for all } i = 1 \dots N, u = 1 \dots M$$

4.3.2 The Two-stage Approach

Directly solving the objective function in Eq. (4.1) is computationally expensive as the item model is not known a priori. A two-stage approach can be applied instead. First, the system estimates the model by active learning techniques. Second, it solves the equation by binary integer programming technique. A matrix factorization technique

in [50] is used to construct the basic model, namely:

$$\hat{p}(R = 1|u, i; \theta) = \hat{R}_{u,i} \equiv \hat{\boldsymbol{p}}_u^T \hat{\boldsymbol{q}}_i \tag{4.2}$$

where the model parameter θ contains $\hat{\boldsymbol{p}}_{\boldsymbol{u}}$ and $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$, two f-dimensional column vectors, representing a set of estimated features for user u and item i respectively. The focus is on $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$ while the $\vec{P} = [\hat{\boldsymbol{p}}_{\boldsymbol{1}}^T, ..., \hat{\boldsymbol{p}}_{\boldsymbol{u}}^T, ..., \hat{\boldsymbol{p}}_{\boldsymbol{M}}^T]^T$ of all existing M users has already been estimated from the historic data using the matrix factorization techniques. Although not explicitly indicated in Eq. (4.1), to have an accurate estimation of $p(R = 1|u, i)$ requires a robust estimation of $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$ for each new item i for $i \in [1, N]$. Thus, instead of directly optimising Eq. (4.1), the system first finds the allocation decision to minimise the variance of $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$.

For mathematical convenience, instead of vector d_i, I use a $M \times M$ diagonal matrix $\boldsymbol{D}_i$ to denote the decision during the exploration, i.e., $\boldsymbol{D}_i \equiv diag(d_i)$. I denote the model in terms of the observations from collected feedback as $R_{u,i} = \hat{\boldsymbol{p}}_u^T \hat{\boldsymbol{q}}_i + e_i$, where e_i is random error. The sum of the squared errors, e_i, using the selected samples can be written as:

$$\sum_{i=1}^{v} e_i^2 \;\; = \;\; \boldsymbol{e}^T \boldsymbol{e} = (\boldsymbol{R_i} - \boldsymbol{P}\hat{\boldsymbol{q}}_{\boldsymbol{i}})^T \boldsymbol{D}_i (\boldsymbol{R_i} - \boldsymbol{P}\hat{\boldsymbol{q}}_{\boldsymbol{i}})$$

Minimising the error to find the optimal $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$ as:

$$\hat{\boldsymbol{q}}_{\boldsymbol{i}}|\boldsymbol{D}_i = (\boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{P})^{-1} \boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{R_i}$$

where if $\boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{P}$ is not invertible, pseudo-inverse or Moore-Penrose-inverse [89] can be used. The respective variance of $\hat{\boldsymbol{q}}_{\boldsymbol{i}}$ is obtained by following the standard linear regression as:

$$\begin{aligned} Var[\hat{\boldsymbol{q}}_{\boldsymbol{i}}|\boldsymbol{D_i}] &= Var[(\boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{P})^{-1} \boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{R_i}] \\ &= \sigma^2 (\boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{P})^{-1} \end{aligned}$$

where the variance of $\boldsymbol{R_i}$ is assumed to be $\sigma^2 I$ and I is an identity matrix. In this chapter, I consider a solution from D-optimal design [88] of active learning, where the objective is to minimise the determinant of the inverse matrix. D-optimal criterion suggests that the system needs to select users (through $\boldsymbol{D}_i$) such that the determinant of $(\boldsymbol{P}^T \boldsymbol{D}_i \boldsymbol{P})$ is the largest.

Once it has been done, the second stage is to find the decision matrix of exploitation $\boldsymbol{d'}$ such that $\sum_{u=1}^{M}\sum_{i=1}^{N}\hat{\boldsymbol{p}}_u^T\hat{\boldsymbol{q}}_i d'_{ui}$ is maximised along with all necessary operational constraints in Eq. (4.1).

4.4 Experiment

In this section, I intend to evaluate the proposed two-stage online recommendation framework. The chief aim is to:

- investigate whether the two-stage approach is better than non-personalized recommendation in terms of improving the positive response rate;
- study the performance and mechanism of D-Optimal design and random selection approaches for exploration; and
- study the importance of the balance of resources allocation for exploration and exploitation and how that affects the outcome.

4.4.1 Data set

To verify the algorithms effectiveness, ideally I would like to have a real-time live system to work with. However, since the data or access of commercial group buying systems is not publicly available, in this chapter I evaluate the framework using a widely used rating dataset - MovieLens 1M data set [90]. The simulations over the real data allow me to verify the theoretical limitations and achieve the valuation goals. The dataset contains 3533 movies and 6041 users represented by unique movie IDs and user IDs respectively. I do not use any of the meta data. It is assumed that users' ratings towards movies can imply their purchase decisions. Movies are items in this context. To simulate purchase decisions with the available ratings between users and items, I convert the ratings to $0, 1$ binary decision values. Rating score 4 or 5 are considered as *purchase* (or essentially "like it"), i.e. 1, while all other rating scores are *not purchase* (or essentially "dislike it"), i.e. 0. Like most information retrieval evaluations, I also assume missing ratings for testing as *not purchase*, making the estimated total number of successful purchases the lower bound of the real ones.

4.4.2 Exploration and Exploitation

In this experiment, I evaluate the effect of exploration and exploitation ratio on the positive response rate, along with the D-Optimal design and the random selection approach for exploration. To make the results clear, no constraint except the limitation of resources, i.e. recommendation opportunities, is taken into account. I will handle the other operational constraints in the next section. There are 6041 users in the system ($M = 6041$). I separate the data set into two sets: One represents all users' purchase decision records towards new items for testing, which is constructed by randomly selecting 20 items from the data set as new items ($N = 20$). I separate them, along with all associated purchase decisions, from the original data set and store them into the new items data set. The other one represents all users' purchase decision towards existing items for training, which consists the remaining 3513 items in the original data set.

The system represents each user by a user model which is a vector with 3 estimated factors ($f = 3$). PureSVD technique [17] of CF is implemented to build these models from the data set of existing items. PureSVD is based on latent-factor, i.e., users and items are represented into a low dimensional space. To simulate the resources limitation, the system only recommends 1 item to each user ($K = 1$). So the total resources the system has, i.e. the total number of recommendations the system makes in both exploration and exploitation stages combined, is $6041 = M \times K$. I evaluate the positive response rate of the following approaches:

Random Allocation

I define the baseline result as using the non-personalized Random Allocation technique, which means that resources are randomly allocated in both exploration and exploitation stages. Simply speaking, the system randomly recommends 1 unique new item to each of the 6041 users.

Random+CF: Random Exploration + CF Exploitation

In the exploration stage, for each new items, sequentially, the system randomly selects a percentage of unique users as queries according to b (the portion of exploration) to collect purchase decisions for item model estimated by ordinary least squares. No user can be selected by more than 1 item. When this stage is completed, I predict the

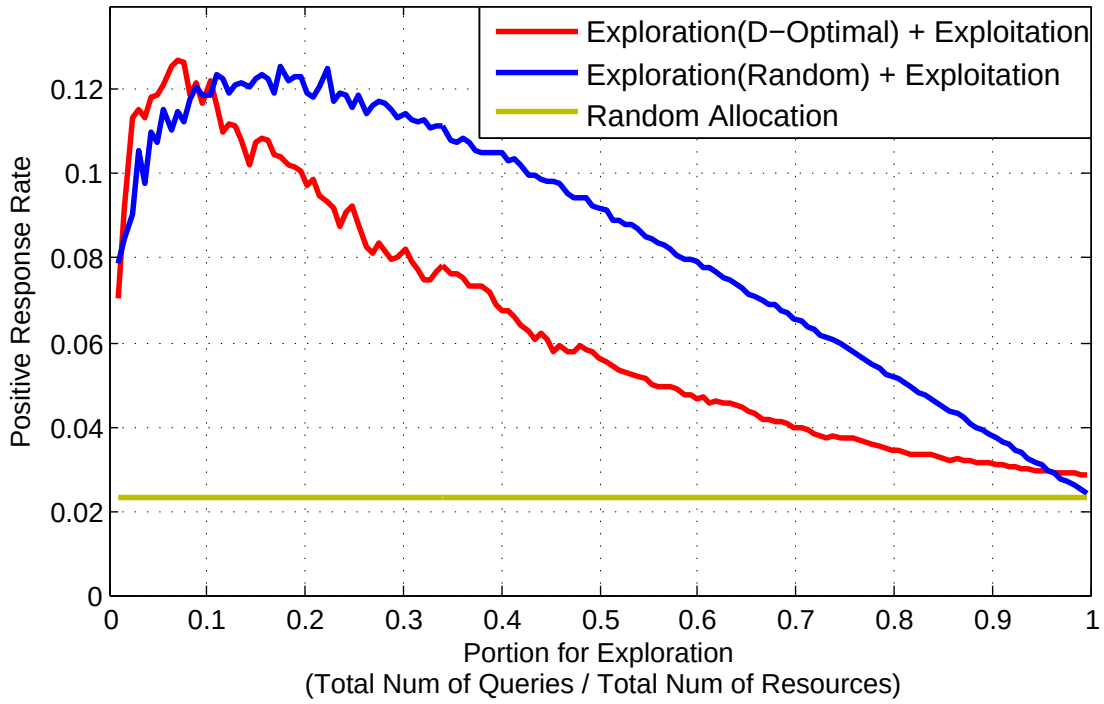

Figure 4.2: Mean of Positive Response Rate

probability of purchase, i.e. $\hat{p}(R = 1|u, i)$, for all users and new items pairs, excluding pairs that have been recommended during exploration. In the exploitation stage, each user u of the 6041 users has a remaining $k_u = 1 - \tilde{k_u}$ resources quota, where $\tilde{k_u}$ is the number of items recommended to user u in the exploration stage. At last the system recommends top k_u items, which have the highest $\hat{p}(R = 1|u, i)$, for each user.

D-Optimal+CF: D-Optimal Exploration + CF Exploitation

This approach is identical to Random+CF, except the system does not select the unique users as queries by random. The system selects the queries for each item based on the D-optimal design.

For each run of the experiment, the system selects 20 new items randomly and run Random Allocation, Random+CF and D-Optimal+CF ten times each against these new items. The system runs the whole experiment 10 times repeatedly with different set of 20 new items.

Fig. 4.2 shows the combined positive response rate of exploration and exploitation against different b (the portion of exploration) level, while Fig. 4.3 shows their standard deviation. Four main observation are to be noticed: 1) The two-stage approach, regardless Random+CF or D-Optimal+CF, is better than Random Allocation; almost 6 times

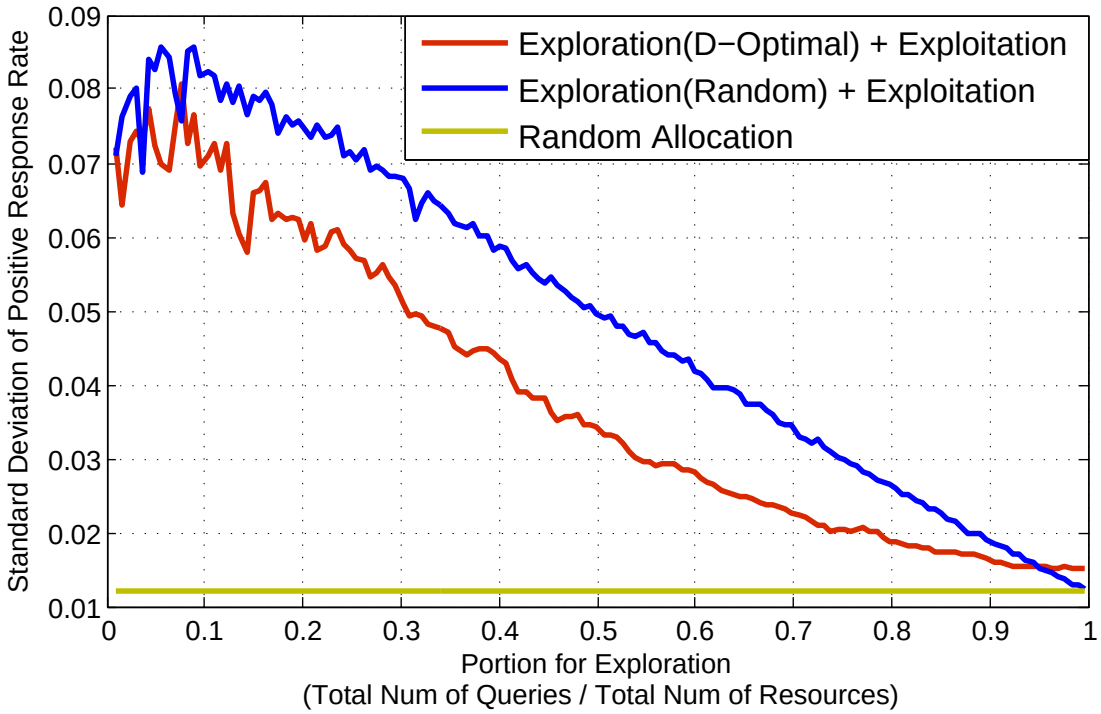

Figure 4.3: Standard Deviation

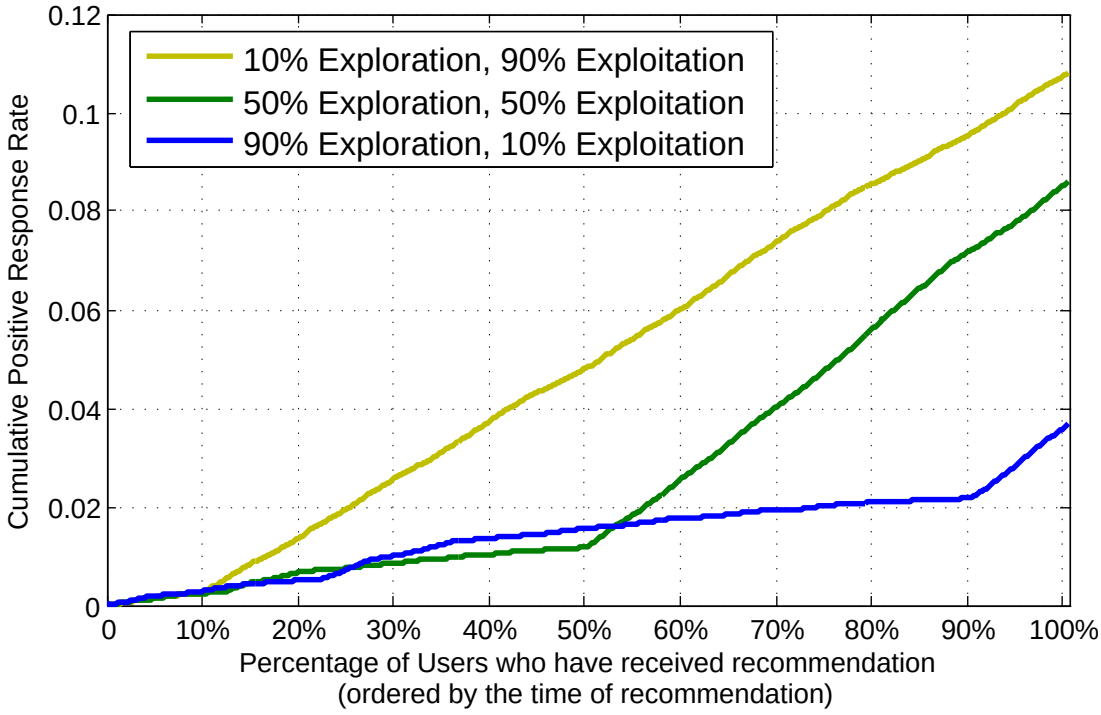

Figure 4.4: Random+CF Ratio

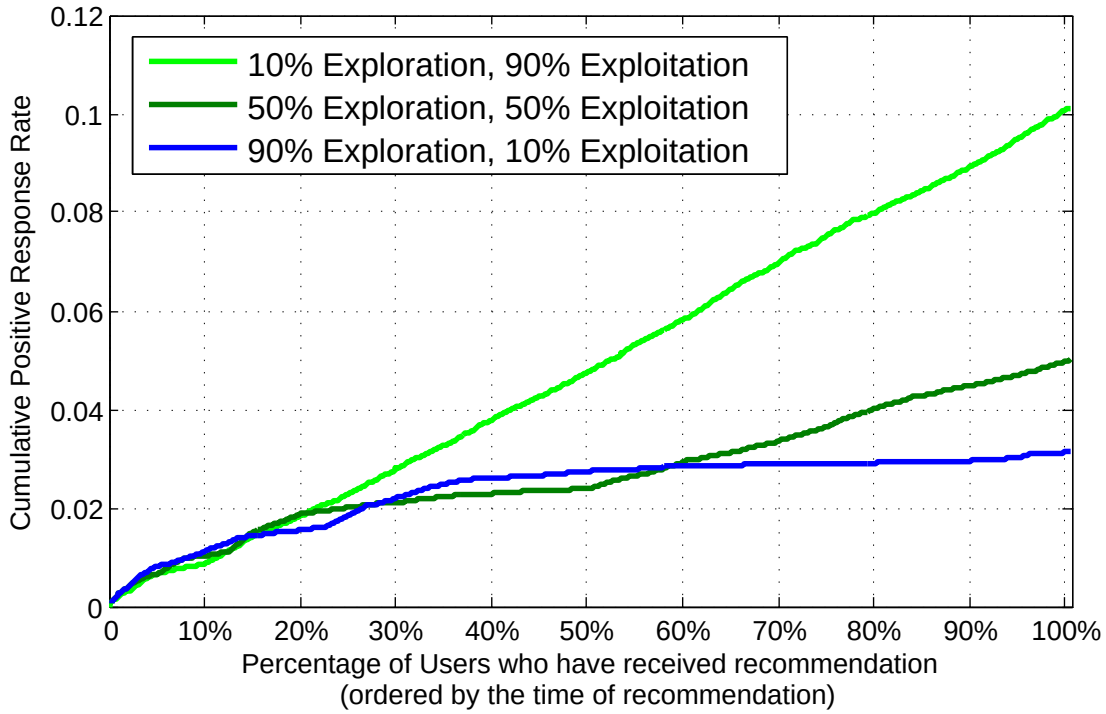

Figure 4.5: D-Optimal+CF Ratio

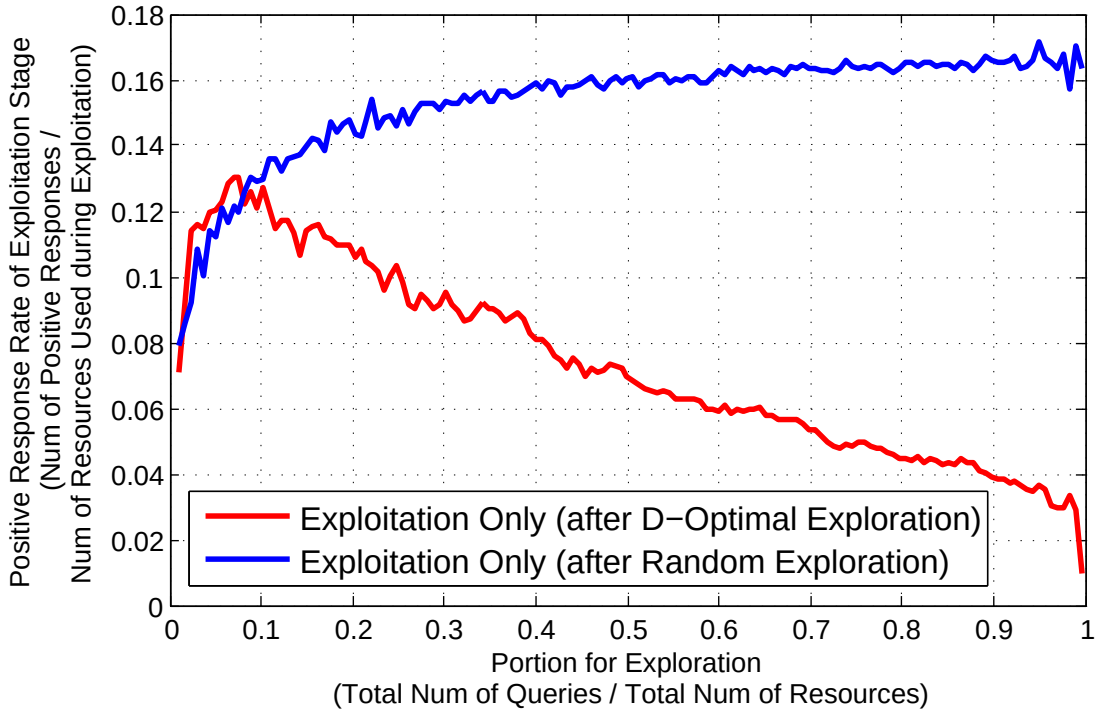

Figure 4.6: Response Rate of Exploitation

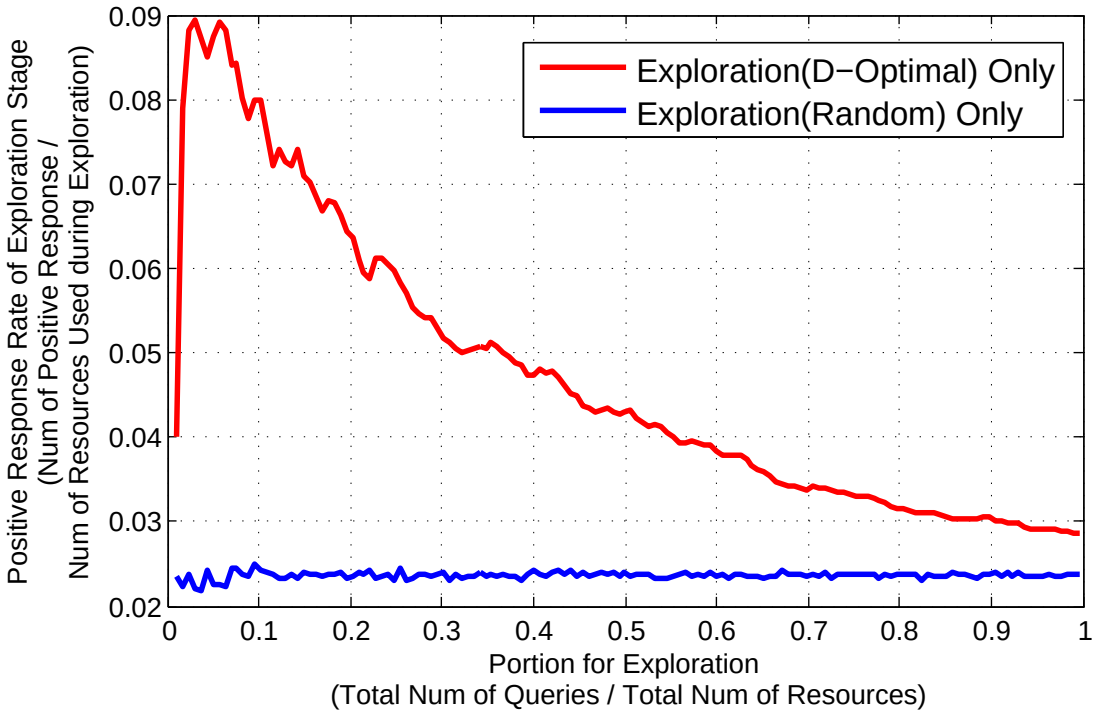

Figure 4.7: Response Rate of Exploration

Approach A	Approach B	p-value	
		without constraints	with constraints
D-Optimal + CF	Random Allocation	5.4604e-005	2.8789e-005
Random + CF	Random Allocation	4.2946e-006	9.5566e-007
D-Optimal + CF	Random + CF	0.7160	0.9812

Table 4.1: p-values for the significance tests between Approach A and B

better than the baseline algorithm. 2) The combined positive response rate of both Random+CF and D-Optimal+CF increase at the beginning, then peak, then start to drop as b increases. 3) Although the combined positive response rate of D-Optimal+CF peaks earlier than Random+CF, the maximum positive response rates of these two approaches are similar, so it seems that the setting of b is more important. 4) The combined positive response rate of D-Optimal+CF drops much faster than Random+CF after peak. It does not seem to agree with the conventional belief in active learning that using D-Optimal design for learning should give better results than using random selection for learning. I further study these points with more figures.

I look closer to effect of the change of b with Fig. 4.4 for Random+CF and Fig. 4.5 for D-Optimal+CF. In Fig. 4.4 (Random+CF), the cumulative positive response rate increases at a faster rate in CF exploitation than in random exploration. In the exploitation part, the slope of the line with 9:1 exploration and exploitation ratio is deeper than the one with 5:5 ratio, which is deeper than the one with 1:9 ratio. It means that, for Random+CF, the more exploration the system does, the better accuracy of prediction for exploitation. As expected, the positive response rate increases as the system increases the resources for exploration at the beginning, which improves the accuracy of prediction for exploitation. When the system starts to use more and more resources for exploration, the system has less remaining resources for exploitation which is harmful for the positive response rate, so it should drop gradually after a certain exploration and exploitation ratio even though the accuracy of prediction is still improving. On the other hand, Fig. 4.5 (D-Optimal+CF) is quite different. Although the cumulative positive response rate also increases at a faster rate in CF exploitation than in D-Optimal exploration, it seems that the accuracy of prediction of the exploitation stage does not improve when I change the exploration and exploitation ratio from 1:9 to 5:5, and from 5:5 to 9:1.

Fig. 4.6 shows that the positive response rate of Random+CF keeps increasing in exploitation stage when b increases, but the increment slows down gradually. Intuitively, after the system has collected more and more data about new items, adding just a few more data does not improve the accuracy of prediction so much as when the system only has very little data. This figure also shows that the positive response rate of D-Optimal+CF improves slightly faster than the one of Random+CF when $b \leq 0.07$.

Surprisingly, however, the positive response rate of D-Optimal+CF starts to drop after this point of b.

The unexpected behavior of D-Optimal+CF, at least for this dataset, can be explained with Fig. 4.7 which shows that the positive response rate during exploration stage of D-Optimal+CF is consistently better than the one of Random+CF. It implies that D-Optimal+CF gives priority to high $p(R = 1|u)$ users for query selection. In other words, it selects more active buyers than Random+CF for exploration. As b increases, more users are selected as queries. D-Optimal+CF then starts to select users with relatively lower $p(R = 1|u)$ when higher $p(R = 1|u)$ users have already been selected. Obviously, if the quota of resources of active buyers, i.e. users with high $p(R = 1|u)$, are used up during exploration stage, it would be much harder to obtain good positive response rate in the exploitation stage, even if the system has better the accuracy of prediction. In conclusion, the seemingly unexpected behavior of D-Optimal+CF is due to existence of the quota of resources.

4.4.3 Optimisation with Operational Constraints

I repeat the previous experiment setting again, but this time, I use the binary integer programming technique to optimise the objective function according to the prediction, with and without the operational constraints (see Section 4.3.1). Under these constraints, the gain of the purchase of items that do not fulfill the minimum purchase requirement is all zero, and the gain of the purchase of items after their maximum inventory availability have been reached is also zero. In this experiment, I use the open-source package OpenSolver [91] to handle the computation of integer binary programming. I calculate the final positive response rate of Random Allocation, Random+CF and D-Optimal+CF. For D-Optimal+CF and Random+CF, I set the exploration and exploitation ratio b at their best level respectively according to my previous experiments. I then conduct paired t-tests similar to one of [92] to compare the performance of these three approaches using the p-value.

At each time, two approaches are compared: A and B. I repeat the experiment n times, each with a different set of N new items. Each time, the final positive response rate for both approaches, denoted as a_i and b_i respectively, are computed, for $i = 1, 2, \ldots, n$. The null hypothesis is that the expected difference $\frac{\sum_{i=1}^{n}(a_i - b_i)}{n}$ is 0.

The alternative hypothesis is that the expected difference > 0, which means that A performs better than B. The p-value is computed using the t-distribution with the degree of freedom $n - 1$ for $T \geq \frac{\text{expected difference}}{\text{s.e. of expected difference}}$. A small p-value ($\leq 0.01$) means that A performs better than B with statistical significance.

Table 4.1 shows that, with or without operational constraints of group buying system, Random+CF and D-Optimal+CF both perform significantly better than Random. However, I cannot conclude that D-Optimal+CF performs better than Random+CF with statistical significance.

4.5 Assessment

I have presented a novel recommendation problem and provided a two-stage batch solution to approximately optimise a practical objective function of group buying systems with operational constraints for new items under the cold-start situation. I show that my solution significantly increases the response rate comparing to the non-personalized recommendation. I conclude that using the popular D-Optimal design approach for exploration does not provide significant advantage over using the random selection approach in some cases. It is explained by my experiments which show that the D-Optimal design prefers to use the quota of resources of active buyers during the exploration stage, which affects the performance of the exploitation stage negatively.

A limitation of this chapter is that I was unable to obtain real group buying dataset from the commercial industry for the experiments at the time of the experiments. A movie recommendation dataset is used instead to simulate purchasing behaviors, which is far from ideal. For future work, I hope to evaluate the proposed approach on real group buying or retail datasets. I are also interested in evaluating the use of other experimental design and active learning techniques in the exploration stage. A drawback of my current method is that the optimal trade-off parameter b was obtained empirically. Given the unique nature of the new item problem, new techniques are required to directly optimise the objective function in Eq. (4.1) by using, for instance, dynamical programming.

Chapter 5

Continuous Model Selection for Changing Business Dynamics

This chapter considers a practical scenario where the business environment is changing over time. It presents an approach to automatically re-select the model by continuously optimising the hyperparameter settings to keep up with the changes. The new approach applies an automatic hyperparameter optimisation technique in collaborative filtering algorithms while the system collects more and more new data. Experiments have been conducted in a real world large retail dataset to challenge traditional assumption that a one-off initial model selection is sufficient. In particular, the proposed approach has been compared with a baseline approach and a widely used approach with two scalable collaborative filtering algorithms. The evaluations of this experiments are based on a 2-year real purchase transaction dataset of a large retail chain business, both its online e-commerce site and its offline retail stores. It is demonstrated that continuous model selection can effectively improve the prediction accuracy of a recommender system. This chapter presents a new direction in improving the prediction performance of a large-scale recommender system in a real-world retail scenario.

5.1 Background

5.1.1 Motivation

Model selection is commonly regarded as an initial step in constructing a predictive model in a recommender system. Since the choice of values of hyperparameters for a learning algorithm would significantly affect the prediction accuracy of the system, applying an algorithm without proper hyperparameter settings may render the system invalid, regardless of how powerful the algorithm is originally designed to be. While the problem of model selection has been studied for decades in various disciplines, the adaptiveness of the initial selected model under a changing environment is not well understood.

This chapter is organised as follows: I will first review the related work in Section 5.1.2. This will be followed by the definitions of preference prediction and model selection in Section 5.3. I will then present and analyse the proposed approach in Section 5.4. Lastly, I will provide experimental evaluations in Section 5.5 and conclude in Section 5.6.

5.1.2 Related Work

In this section, I review the related work in hyperparameter optimisation for collaborative filtering. Almost all collaborative filtering algorithms come with one or more adjustable hyperparameters, such as regularisation and learning rate. The setting of these values plays a key role in the generalisability and accuracy of the predictive model [93]. The problem of finding better hyperparameter values for an algorithm to improve prediction accuracy or to optimise certain goals is called hyperparameter optimisation or model selection. This problem has been studied in multiple disciplines in the past [94, 95, 96, 97, 98]. Response surface methodology [99] is a popular method for tuning parameters in statistics. In Machine Learning, advanced methods to optimise hyperparameters have been proposed, such as the use of Gaussian Process [100], Random Forests [101], Tree-structured Parzen Estimator [102] and Reinforcement learning [103]. On the other hand, two simple techniques are widely used in commercial industries and in software packages such as scikit-learn [104] and libsvm [105]; they are grid search and random search [106]. Grid search is a process that searches exhaus-

tively through a manually specified subset of the hyperparameter space of the targeted algorithm. Random search, on the other hand, selects a value for each hyperparameter independently using a probability distribution. Existing literature in collaborative filtering, or in recommender systems in general, handle hyperparameter settings by trial and error manually. While this approach may be acceptable if the system needs to conduct model selection only once at the initial stage, it is impractical when the system wants to conduct continuous model selection when new data comes it repeatedly. I, therefore, apply random search as a baseline hyperparameter optimisation technique on collaborative filtering algorithms in this chapter.

Literatures such as [107] have also presented the benefits of online model selection. To my best knowledge, no related work has been conducted for Collaborative Filtering and in the retail settings. Online model selection for Collaborative Filtering can be an interesting topic for future work. It is also worth mentioning that online model selection is different from online learning algorithms such as online collaborative filtering [108] and online matrix factorization [109]. Online model selection describes methods that update model hyperparameters continuously as new data is collected, on the other hand, online learning describes methods that update the model, without updating its hyperparameter such as regularization rate, by learning from new data continuously. In future work, the comparison in prediction accuracy between the proposed online model selection method and online learning techniques can be done.

5.2 Research Challenge

Recommender systems are increasingly important for retail businesses. Retailers commonly provide personalised product recommendations to consumers through various mediums, for instance, web advertisement, email newsletters and in-store discount vouchers. Recommender systems attempt to accurately predict consumers' preferences, or their behaviours, so that they can recommend products according to the best expected scenarios.

Some retail businesses collect a large amounts of data every day. Scalable algorithms that can process data and construct predictive models using multiple machines in parallel are used in large-scale recommender systems. In this chapter, I focus on two scalable latent factor models, namely Alternating Least Squares with Weighted

Regularisation (ALS-WR) [52] and Stochastic Gradient Descent (SGD) [110], both of which construct consumer and product feature matrices in Singular Value Decomposition (SVD) form. Both algorithms can be run on distributed Hadoop systems [111]. Like almost all other algorithms, these algorithms require the setting of one or more hyperparameters, such as regularisation and learning rate, to build the latent factor models. These hyperparameters affect the prediction accuracy of the resulting models directly. Hyperparameter optimisation problems and techniques have been studied in different disciplines such as Machine Learning and Statistics. Grid search and random search [106] are two popular methods for global optimisation. All these studies, however, take for granted that hyperparameter optimisation (or setting) is a one-off initial process that needs to be done only once. No study, to the best of my knowledge, has been conducted on the effect of re-selecting new hyperparameters for learning algorithms after new data has been collected over time.

Retail business environment changes rapidly. Product prices, for instance, may be changing frequently and new products appear on the market regularly. Simply re-training the same predictive model with the addition of new data may not be sufficient to accommodate these changes. My hypothesis is that a recommender system model may render less accurate or even invalid results, as the business dynamic changes over time. I propose a continuous model selection approach, in which hyperparameters are re-evaluated, and are re-selected if needed, before the model is re-trained with new data. The prediction accuracy of this approach is compared to traditional non-continuous approaches empirically, based on large and real retail datasets that contain both online e-commerce transactions and offline in-store transactions. Experiments show that the proposed approach outperforms a baseline approach and a state-of-the-art approach significantly in my retail scenarios.

5.3 Preference Prediction

Retailers sometimes aim to achieve different objectives through Recommender Systems. For instance, some recommender systems are built to focus on presenting a diverse set of products to consumers [112, 113], while others aim at maximising profit [114]. In this chapter, I narrowly define that the objective of a recommender system is to suggest the top N products to each customer that he or she will most likely pur-

chase. This kind of system, broadly speaking, involves a two-step process: preference prediction and ranking. During the prediction step, the system predicts a preference score on each available product for a targeted consumer. A higher score means that the consumer likes the product more, and therefore has a higher chance to make a purchase. Then, during the ranking step, the system ranks all available products for this consumer according to the predicted score. The top-ranked N products will be returned as recommendation results.

To predict the preference scores, a recommender system commonly relies on a predictive model constructed by a content-based approach or a collaborative filtering approach. In this chapter, I focus on collaborative filtering algorithms since they are widely used in the commercial industry [2, 115]. Collaborative filtering algorithms construct predictive models based on consumers' previous behaviours to predict their future, or unknown, preferences. Here are the formal definitions:

Definition 1 (Consumer Preference): Consumers may express their preferences explicitly, for instance, they may give ratings to products. Ratings can be converted into preference scores directly using a desired scale. Additionally, consumer preferences can be implied by consumers' behaviours. For instance, it is reasonable to assume that a consumer is somewhat interested in a product if he or she has browsed the product details web page. It can also be assumed that a consumer likes a product even more if he or she has purchased it. These behaviours can also be converted into preference scores using a pre-defined scale. These preference scores derived from existing dataset become the labels for training a predictive model that predicts consumers' future preferences.

Definition 2 (Prediction): In my retail scenario, a prediction of consumer preference can be regarded as an estimation of the probability a consumer will purchase a product, i.e. $p(purchase = 1|u, i)$ where u is a targeted consumer and i is a targeted product. Preference prediction of all customers on all products can be represented by a $M \times N$ matrix $\boldsymbol{r}$ where M is the total number of consumers, N is the total number of products and $r_{mn} = p(purchase = 1|m, n)$. There are two main techniques of Collaborative Filtering for prediction: the neighbourhood approach and latent factor models approach. Latent Factor Models is a popular discipline for Collaborative Filtering. Literature [116] has combined neighbourhood approach into the latent factor

model format for better accuracy. These methods transform both products and consumers to the same latent factor space. The latent space is then used to predict future consumer preferences by characterising both products and consumers in term of factors automatically inferred from previous consumer behaviours.

A popular approach to latent factor models is induced by an SVD-like lower rank decomposition of the prediction matrix. Let's say the number of latent features is set as k. I then define $\boldsymbol{p}$ as a $k \times M$ matrix representing factors of k features of M consumers. I also define $\boldsymbol{q}$ as a $k \times N$ matrix representing factors of k features of N products. $\boldsymbol{p}$ and $\boldsymbol{q}$ are trained by algorithms such as Alternating Least Squares with Weighted Regularisation and Stochastic Gradient Descent using previous behaviours of M consumers on these N products. The basic latent factor model for consumer-product preference prediction can be written as:

$$\boldsymbol{r} = \boldsymbol{p}\boldsymbol{q}^{\boldsymbol{T}}$$

The u^{th} consumer is associated with a consumer-factors vector $\boldsymbol{p_u}$, and the i^{th} product has an product-factors vector $\boldsymbol{q_i}$. If I am concerned about the model of just one consumer, say the u^{th} consumer, I can form a single consumer model like this:

$$\boldsymbol{r_u} = \boldsymbol{p_u}\boldsymbol{q}^{\boldsymbol{T}}$$

where

$$\boldsymbol{r_u} = \begin{bmatrix} r_{u1} \\ r_{u2} \\ \vdots \\ r_{uN} \end{bmatrix}, \boldsymbol{p_u} = \begin{bmatrix} p_{1u} \\ p_{2u} \\ \vdots \\ p_{ku} \end{bmatrix}$$

$$\boldsymbol{q} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1N} \\ q_{21} & \ddots & & q_{2N} \\ \vdots & & \ddots & \vdots \\ q_{k1} & q_{k2} & \cdots & q_{kN} \end{bmatrix}$$

$\boldsymbol{r_u}$ and $\boldsymbol{p_u}$ are the u^{th} column of $\boldsymbol{r}$ and $\boldsymbol{p}$ respectively. Obviously, preference prediction of one consumer to one product is done by the rule: $r_{ui} = \boldsymbol{p_u}\boldsymbol{q_i}^{\boldsymbol{T}}$.

Definition 3 (Model Selection): The term hyperparameter is used to distinguish it from regular parameters of a predictive model. Regular parameters characterise the model and they are estimated by model training using existing data. Hyperparameters,

on the other hand, control how these regular parameters are estimated. Therefore, hyperparameter optimisation is also called model selection. Like most machine learning algorithms, most collaborative filtering algorithms come with some tunable hyperparameters such as learning rate and regularisation. Selecting appropriate settings for these hyperparameters is crucial for the accuracy of prediction. Model selection, or hyperparameter optimisation, can be defined formally as an optimisation problem itself, which can be written as:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, C(\boldsymbol{p}_v, \boldsymbol{q}_v, \boldsymbol{r}_v, M(\boldsymbol{p}_{tr}, \boldsymbol{q}_{tr}, \boldsymbol{r}_{tr}, \theta))$$

and the performance of a set of hyperparameter settings can be evaluated by a cost function:

$$C(\boldsymbol{p}_{test}, \boldsymbol{q}_{test}, \boldsymbol{r}_{test}, M(\boldsymbol{p}_{tr+v}, \boldsymbol{q}_{tr+v}, \boldsymbol{r}_{tr+v}, \theta^*))$$

where C is the cost function, M is the modeling function, θ is a hyperparameter set, $\boldsymbol{p}_{tr}$ and $\boldsymbol{q}_{tr}$ are the consumer and product feature vectors of the training dataset, $\boldsymbol{p}_v$ and $\boldsymbol{q}_v$ are the consumer and product feature vectors of the validation dataset, $\boldsymbol{p}_{test}$ and $\boldsymbol{q}_{test}$ are the consumer and product feature vectors of the test dataset and $\boldsymbol{r}_{tr}$, $\boldsymbol{r}_v$ and $\boldsymbol{r}_{test}$ are the vectors of preference scores of training, validation and test datasets respectively.

5.4 Proposed Continuous Modeling

In this chapter, I study the effect of conducting automatic hyperparameter optimisation continuously on recommender systems in a retail chain business. My hypothesis is that a continuous model selection approach can improve prediction accuracy of a recommender system in a retail environment. I propose a novel approach to search for better hyperparameters for a targeted Collaborative Filtering algorithm before each iteration of model re-training with the addition of new data. Algorithm 1 is an overview of the process. Major components of the proposed approach is described as follows.

5.4.1 Collaborative Filtering

Some retail businesses collect a large amount of data every day. A parallel distributed collaborative filtering algorithm is necessary to handle the data in a scalable way. In my experiments, two popular distributed collaborative filtering algorithms, Stochastic Gradient Descent (SGD) and Alternating Least Squares with Weighted Regularisation

Data: D (dataset of month 1-24), A (algorithm)
Result: RMSE and MAP@k on each month of 13-24
$trainD$ = D[month 1-12];
for $m \in \{13..24\}$ **do**
  θ^* = select hyperparameters of A based on $trainD$;
  P = model of A trained by $trainD$ and θ^*;
  evaluate RMSE of P on $D[m]$;
  evaluate MAP@k of P on $D[m]$;
  $trainD = trainD + D[m]$;
end

Algorithm 1: Continuous Model Selection

(ALS-WR), are implemented. I also apply a technique to handle data of implicit consumer feedback since consumers do not express their preferences explicitly, such as rating products, in many retail scenarios.

Stochastic Gradient Descent

SGD initialises feature vectors that represent the profiles of consumers and products with random values. It then computes the gradient of the cost function and updates the values with steps in the direction of the gradient based on training data [50, 117]. This chapter follows the SGD implementation of [43], namely Bias SGD, which is a slightly improved version of pure SGD as it computes a bias for each consumer and product. The prediction accuracy is improved at the cost of extra computation. The tuning of three hyperparameters, which are listed on Table 5.1, is needed in order to apply a SGD algorithm to construct a predictive model.

Parameter Name	Description
λ	regularisation to prevent overfitting
$lrate$	learning rate
$decay$	learning rate decay

Table 5.1: Hyperparameters of Bias SGD Algorithm

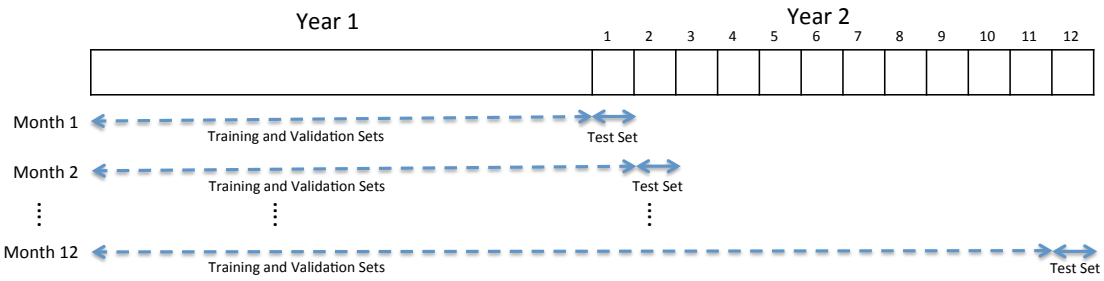

Figure 5.1: Training, Validation and Test Sets Split for 12 Months

Alternating Least Squares with Weighted Regularisation

ALS-WR also initialises feature vectors that represent the profiles of consumers and products with random values. It then updates the feature vectors by applying a least squares solution to solve a quadratic cost function [52]. Although the computational cost of a least squares is more expensive than the one of SGD, usually fewer ALS iterations are required to obtain similar prediction accuracy as SGD. Also, the distributed implementation of ALS is more efficient than the one of SGD. Furthermore, ALS requires less hyperparameter settings. The only hyperparameter required by ALS-WR is shown in Table 5.2.

Parameter Name	Description
λ	regularisation to prevent overfitting

Table 5.2: Hyperparameters of ALS Algorithm

Implicit Feedback

In the real world retail environment, consumer behavioural data is usually collected implicitly. Consumers are not required to express their preferences on products explicitly as it may disrupt their shopping experience. Therefore, the most common form of data being collected is non-negative feedback, such as purchase transactions and website browsing history. Explicit feedback, such as rating, is rarely collected. Under this situation, many collaborative filtering algorithms cannot be applied directly as there is a lack of feedback on which products consumers dislike. Some techniques can be applied to solve this problem. For ALS-WR, consumers' purchase history is transformed into confidence for preference. The cost function takes not only consumer and product pairs that have interactions in the past into consideration, but all other unobserved consumer and product pairs as well. The details of this technique has been discussed in [35]. For SGD, I follow the technique described in [118], which is a straightforward procedure to add negative examples randomly for unobserved consumer and product pairs.

5.4.2 Update with New Data

Existing literature assumes that model selection for collaborative filtering algorithms needs to be done once only at the initialisation stage. Under this assumption, recommender systems re-train models with the same hyperparameter settings when new data comes in. This chapter challenges this status quo. My hypothesis is that the dynamics of real world environments, especially in retail businesses, change rapidly; a model that is appropriate in the past may become less effective as time goes by. Prediction accuracy of a recommender system should be improved continuously if the model is updated, and not only re-trained. To update the model continuously, the system should be able to automatically select new hyperparameter settings for the underlying algorithm when more data is collected. Tuning these hyperparameters manually is not practical as it would require human involvement every time. Automatic hyperparameter optimisation technique is discussed in the next subsection.

5.4.3 Automatic Hyperparameter Optimisation

Automatic hyperparameter tuning has been studied in various disciplines, such as Machine Learning and Statistics. While some techniques are algorithm-specific such as

[119] and [120], some global optimisation approaches can be applied to any algorithms, including collaborative filtering algorithms. Two widely used global optimisation approaches are *grid search* and *random search* [106]. Grid search is a process that searches exhaustively through a manually specified subset of the hyperparameter space of the targeted algorithm. Random search, on the other hand, selects a value for each hyperparameter independently using a probability distribution. Both approaches evaluate the cost function based on the generated hyperparameter sets. Despite its simplicity, random search has proven to be more efficient than grid search in some cases empirically [106]. Since I am mainly concerned about the comparison between the traditional one-off modeling approach and my proposed continuous modeling approach in this chapter, the detailed study of the efficiency of hyperparameter optimisation techniques is not within the scope. It is therefore reasonable to choose either one of these techniques to perform automatic hyperparameter optimisation. The random search technique is chosen due to its simplicity.

5.5 Experiment

5.5.1 Data Set

My dataset, which is provided by a large UK retail chain business, is the 2-year anonymized product purchase records of loyalty card holders on the retailer's e-commerce site and in all physical stores of the retailer in the UK. It contains complete transaction records of 10,217,972 unique loyalty card holders and 2,939 unique products under 10 selected brands. There are 21,668,137 in-store purchase transaction records and 2,583,531 online purchase transaction records. All data is collected in a real non-experimental setting.

I take all records of purchase transactions and give each of them a preference score of 1. For instance, consumer u purchases a product i at time t is represented by a comma-separated line:

$$u, i, t, \mathbf{1}$$

The resulting dataset contains only consumer-product pairs with preference score of 1 only. This transaction dataset is further divided into two: One contains transaction records of the online e-commerce site, and the other one contains transaction records

of offline retail stores.

5.5.2 Evaluation Metrics

One way to evaluate the prediction accuracy of a recommender system is to measure its prediction error of preference scores directly. Recommender systems predict consumer preferences based on predictive models that are built for predicting preference scores of unseen consumer-product pairs. A preference score ranges between 0 and 1, which can also be seen as an estimated probability of a consumer purchasing a product, i.e. $p(purchase = 1|u, i)$ where u is the targeted consumer and i is the targeted product. A higher score means that the consumer is more likely to purchase the product. The predicted scores will be evaluated against test sets that contain the actual purchase transactions i.e. consumer-product pairs with score of 1. Root Mean Squared Error (RMSE) can be used as the prediction error metric: $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i - \hat{r}_i)^2}$, where n is the total number of pairs in a test set, r_i is the true score which is always 1 in this case and $\hat{r}_i$ is the predicted score. The lower the RMSE score, the more accurate the predictive model.

Normally, the end result of a recommender system is to return a top-K recommendation list. Another way to evaluate the prediction accuracy of a recommender system is, therefore, to treat the prediction as a ranking problem. Specifically, I want to evaluate the average precision of the predicted recommendation list for each consumer [9]. Suppose a consumer has purchased n products in the test dataset and the system can recommend up to K products to this consumer. The average precision score at K, i.e. ap@K, is: $ap@K = \sum_{k=1}^{K} P(k)/min(n, K)$ where $P(k) = 0$ if the consumer has not purchased the k-th product of the recommended list in the test dataset and $P(k) = k$ if otherwise. The mean average precision for M consumers at K, i.e. MAP@K, is the average of the average precision of each consumer, which is defined as: $MAP@K = \sum_{m=1}^{M} ap@K/M$. The higher the MAP@K score, the better the recommender system performs.

I am going to evaluate my proposed continuous model selection approach with both RMSE and MAP@K metrics.

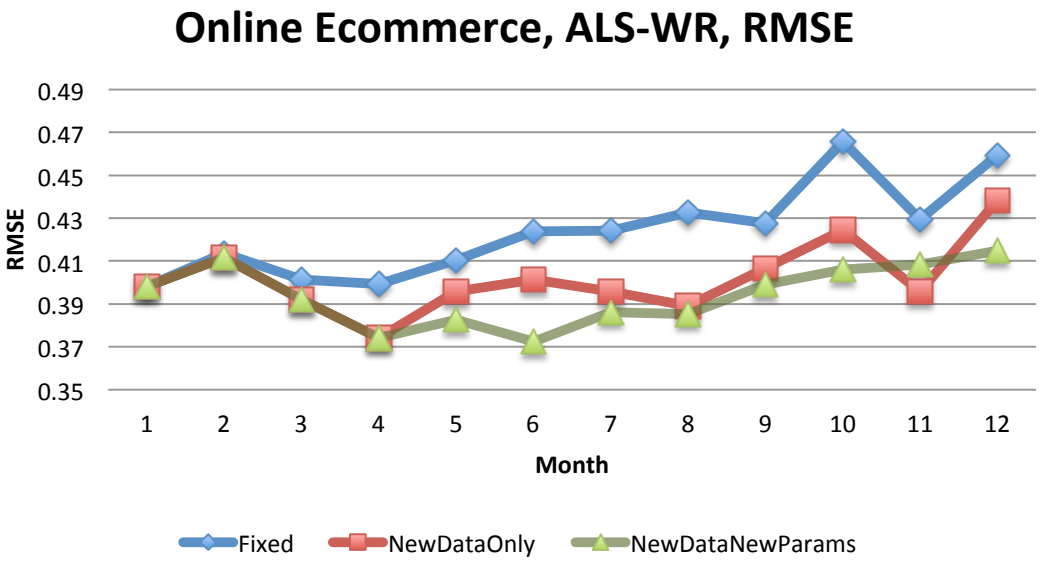

Figure 5.2: RMSE on ALS-WR - 2nd year online e-commerce

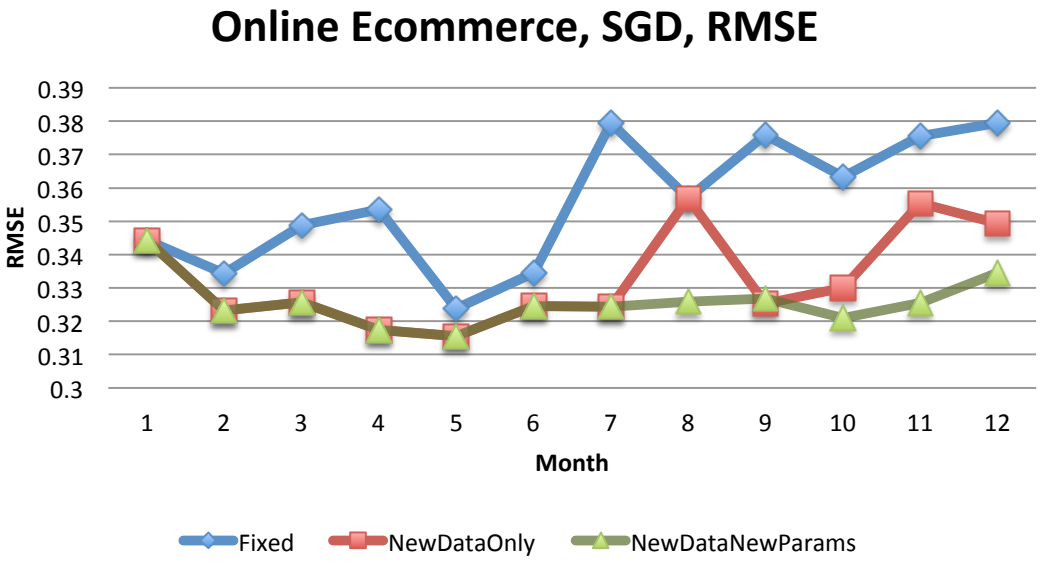

Figure 5.3: RMSE on SGD - 2nd year online e-commerce

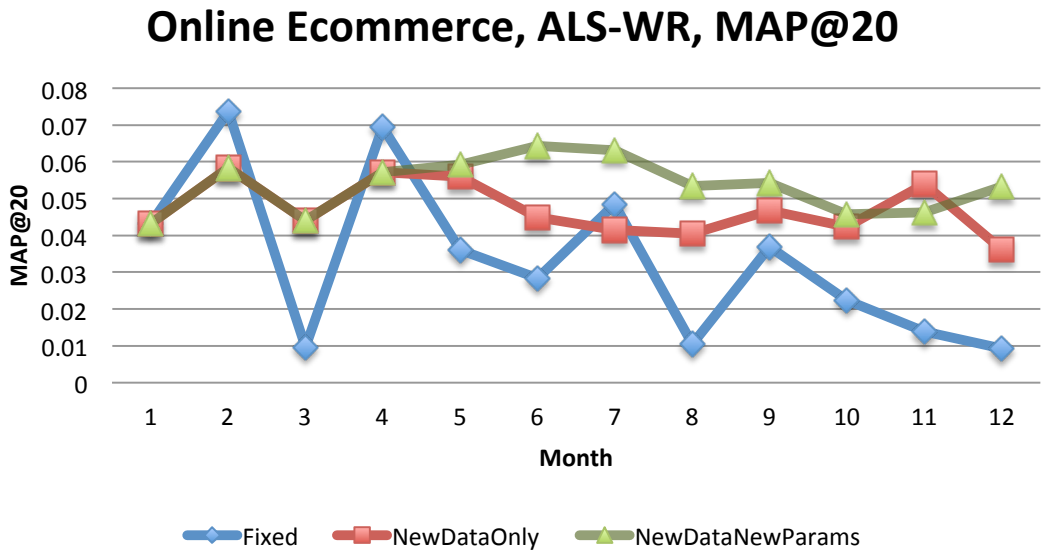

Figure 5.4: MAP@20 on ALS-WR - 2nd year online e-commerce

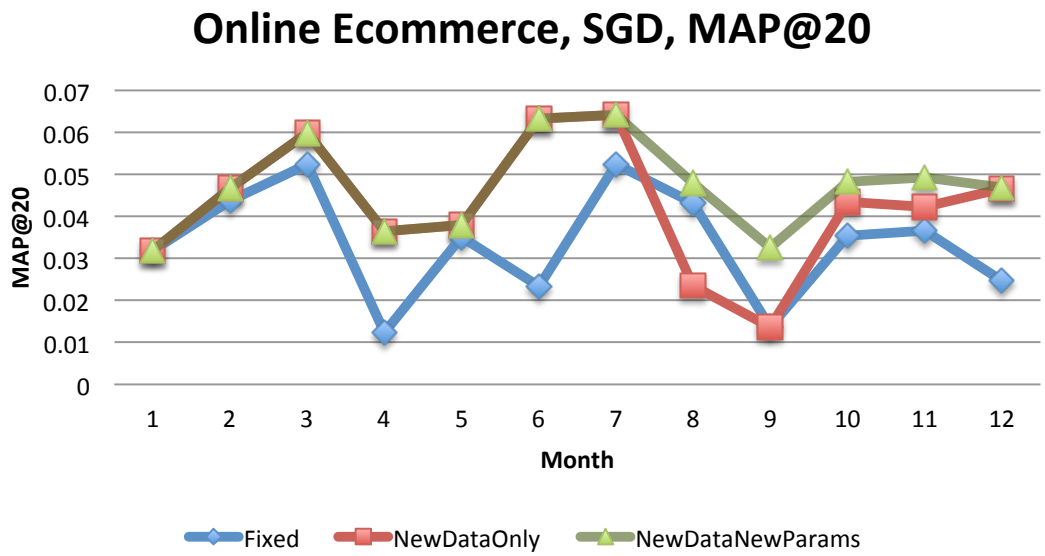

Figure 5.5: MAP@20 on SGD - 2nd year online e-commerce

5.5.3 Experimental Setup

In this experiment, I evaluate the effect of continuous model selection under two separate environments, namely online e-commerce and offline stores. Under each environment, the prediction accuracy over time of three approaches is compared. In particular, this experiment evaluates and compares the consumer preference prediction accuracy of predictive models built, and updated, by three approaches on each month of the second year of my dataset, as illustrated in Figure 5.1. The detailed procedure is described as follows.

I first evaluate the online e-commerce environment along with ALS-WR algorithm. I use $D1$ to represent all online e-commerce transaction data of the first year and $D2_m$ to represent all online e-commerce transaction data of the m month of the second year. For instance, $D2_1$ means all transaction data of the 1^{st} month of the second year.

In this experiment, I define the width of the feature vectors for consumers and products as 20 and the maximum number of iterations as 30. For ALS-WR, there is only one hyperparameter to be tuned, namely the λ variable which regularises overfitting. By using random search technique, a pool of 1000 available hyperparameter sets, known as $\boldsymbol{\theta}$, is generated for ALS-WR. I set the search range of λ to be [0.01, 0.1]. The RMSE and MAP@20 of the following three approaches on each $D2_m$ are evaluated:

Fixed: The first approach starts with selecting one set of hyperparameters from $\boldsymbol{\theta}$ that performs the best based on the first year data. This set of hyperparameters is written as $S_{\boldsymbol{\theta}}(D1)$. A predictive model, $M(D1, S_{\boldsymbol{\theta}}(D1))$, is built using $D1$ and $S_{\boldsymbol{\theta}}(D1)$. I evaluate the RMSE and MAP@20 of this model on each month of the second year, i.e. $D2_m$. This model is not updated with any new data. It serves as the baseline to compare with the other two approaches.

NewDataOnly: The second approach also starts with selecting the same set of hyperparameters, i.e. $S_{\boldsymbol{\theta}}(D1)$, from $\boldsymbol{\theta}$. A predictive model, $M(D1, S_{\boldsymbol{\theta}}(D1))$, is built as well. I evaluate the RMSE and MAP@20 of this model on the *first month* of the second year, i.e. $D2_1$, only. Then, I re-train the predictive model using $S_{\boldsymbol{\theta}}(D1)$ and the first year data *plus* the data of the first month of the second year, which becomes $M(D1 + D2_1, S_{\boldsymbol{\theta}}(D1))$. I evaluate the RMSE and MAP@20 of this updated model on $D2_2$. Similarly, I build $M(D1 + D2_1 + D2_2, S_{\boldsymbol{\theta}}(D1))$ and evaluate its RMSE and MAP@20

on $D2_3$ and so on, until I have evaluated all twelves $D2_m$. The hyperparameter set remains unchanged throughout the whole process.

NewDataNewParams: The third approach, as outlined in Algorithm 1, also starts with selecting the same set of hyperparameters, i.e. $S_{\boldsymbol{\theta}}(D1)$, from $\boldsymbol{\theta}$. Again, a predictive model, $M(D1, S_{\boldsymbol{\theta}}(D1))$, is built. Similar to the second approach, I evaluate the RMSE and MAP@20 of this model on the *first month* of the second year, i.e. $D2_1$, only. Uniquely for this proposed approach, I do the model selection again at this stage. I try to find another set of hyperparameters from $\boldsymbol{\theta}$ that performs the best based on the first year data *plus* the data of the first month of the second year. This new hyperparameter set, $S_{\boldsymbol{\theta}}(D1 + D2_1)$, may be the same as $S_{\boldsymbol{\theta}}(D1)$ if it is still the best performing set. Then, I re-train the predictive model using $S_{\boldsymbol{\theta}}(D1 + D2)$ and the first year data *plus* the data of the first month of the second year, which becomes $M(D1 + D2_1, S_{\boldsymbol{\theta}}(D1 + D2_1))$. I evaluate the RMSE and MAP@20 of this updated model on $D2_2$. Similarly, I re-select the best hyperparameter set from $\boldsymbol{\theta}$ and evaluate the RMSE and MAP@20 of $M(D1 + D2_1 + D2_2, S_{\boldsymbol{\theta}}(D1 + D2_1 + D2_2))$ on $D2_3$ and so on, until I have evaluated all twelve $D2_m$.

For all these approaches, the selection of the best set of hyperparameters is done by evaluating RMSE or MAP@20, according to the final evaluation metrics, with k-fold cross-validation where $k = 10$. Also, techniques to handle implicit feedback data for ALS-WR and SGD discussed previously have been applied.

I wish to state my observation of two issues in particular:

- whether a hyperparameter set that does not perform the best at earlier month would become the best choice in later months
- compare the prediction accuracy of **Fixed**, **NewData-Only** and **NewDataNew-Params**

The results of the experiment are shown in Figure 5.2 and 5.4.

This experiment is repeated with another collaborative filtering algorithm - SGD. For SGD, I define the width of the feature vectors for consumers and products as 20 and the maximum number of iterations as 30. There are three hyperparameters to be tuned for SGD, namely the λ variable which regularises overfitting, $lrate$ which is the learning rate and $decay$ which controls the decay of the learning rate. By using random

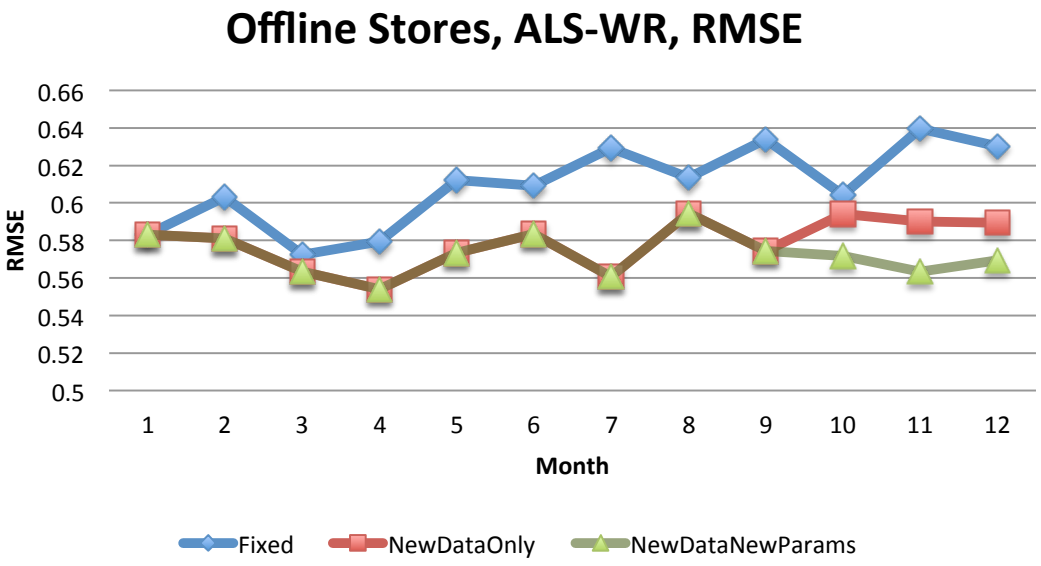

Figure 5.6: RMSE on ALS-WR - 2nd year offline stores

search technique, a pool of 1000 available hyperparameter sets is also generated for SGD. I set the search range of λ to be [0.001, 0.01], the range of $lrate$ to be [0.001, 0.01] and the range of $decay$ to be [0.1, 1]. The results are shown in Figure 5.3 and 5.5.

In the end, the whole experimental setup is re-run again for the purchase transaction dataset of offline stores. The results are shown in Figure 5.6, 5.7, 5.8 and 5.9.

5.5.4 Effectiveness on the Online Site

Figure 5.2 shows the RMSE evaluation of the consumer preference prediction using ALS-WR for the second year of the online e-commerce dataset. The *NewDataOnly* approach clearly outperforms the *Fixed* baseline approach in every month. It means that re-training the predictive model with new data every month helps to improve preference score prediction in this case. For the proposed *NewDataNewParams* approach, new hyperparameter sets are selected at month 5 and month 10, as shown in Table. 5.3. The predictive model is therefore re-trained based on a new λ value from month 5 to month 9 and another new value from month 10 to month 12. During the period between month 5 to month 12, inclusively, *NewDataNewParams* outperforms *NewDataOnly* in every month except month 11.

Figure 5.4 shows the MAP@20 evaluation of Top-20 recommendation, i.e. MAP@20, for the same ALS-WR. As shown, the curve representing the Fixed ap-

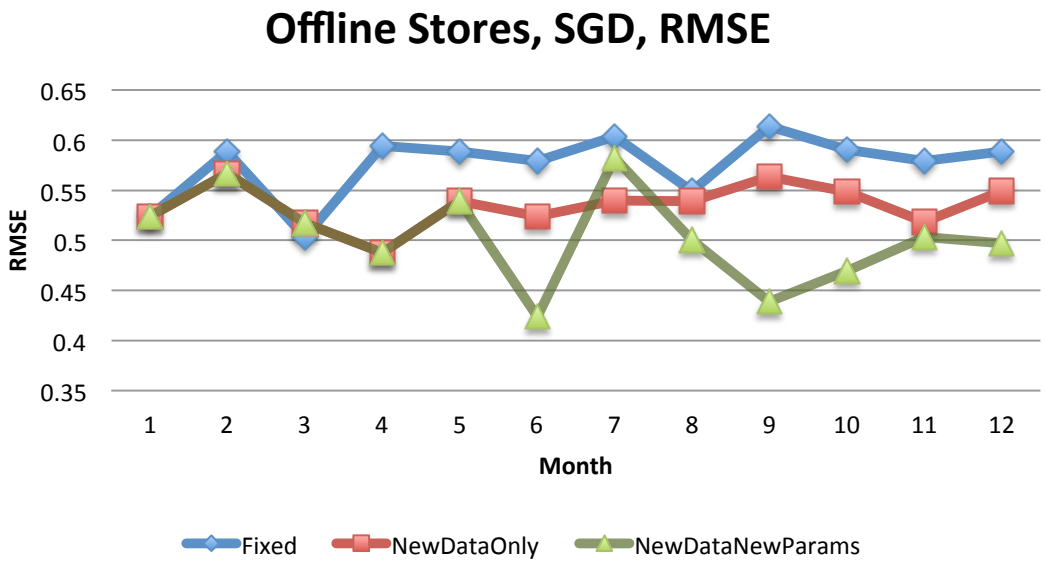

Figure 5.7: RMSE on SGD - 2nd year offline stores

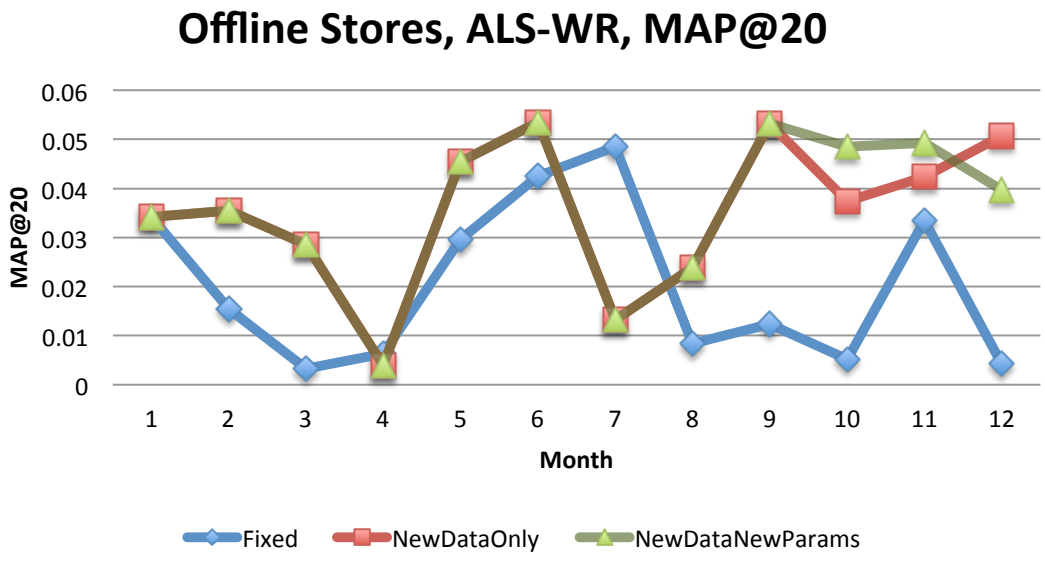

Figure 5.8: MAP@20 on ALS-WR - 2nd year offline stores

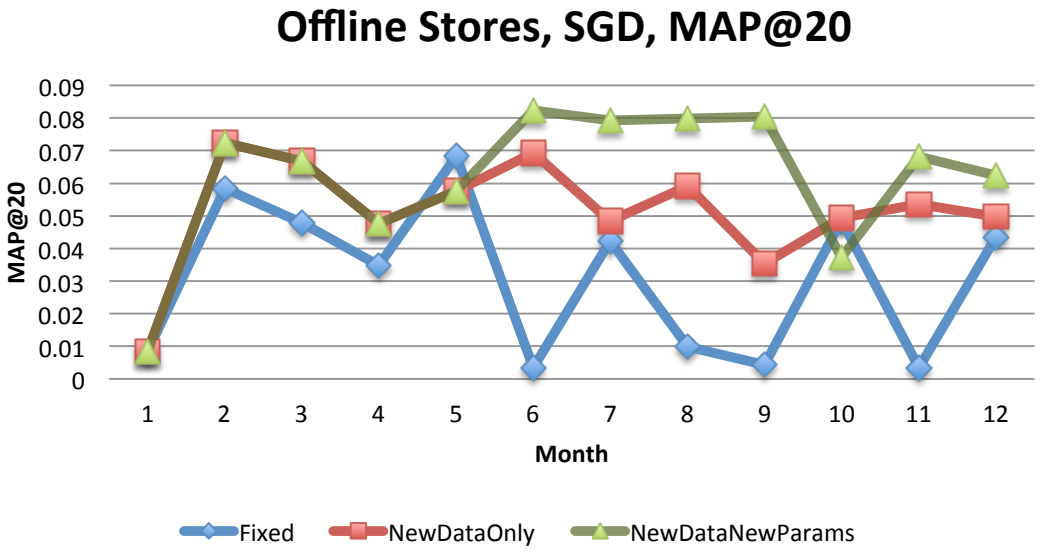

Figure 5.9: MAP@20 on SGD - 2nd year offline stores

proach fluctuates in a relatively large range while the curves of the other two approaches both appear to be relatively smoother. For the Fixed approach, the results of the top 20 recommendation perform badly on month 3, 8 and 12. The *NewDataOnly* approach improves the recommendation results in these months and some other months significantly. It outperforms the *Fixed* baseline approach in 8 out of the 11 months after an exact same start in the first month. For the *NewDataNewParams* approach, it outperforms *NewDataOnly* every month from month 5 to month 9 - the period during which the predictive model is re-trained by the first new hyperparameter set. From month 10 to 12, when another new hyperparameter set is used, *NewDataNewParams* outperforms *NewDataOnly* in 2 out of 3 months.

	Month	λ
Online	1 to 4	0.054
	5 to 9	0.038
	10 to 12	0.033
Offline	1 to 9	0.062
	10 to 12	0.048

Table 5.3: Selected ALS-WR Hyperparameters

Figure 5.3 and 5.5 show the RMSE and MAP@20 evaluation of prediction using SGD respectively. For the proposed *NewDataNewParams* approach, a new hyperparameter set is selected at month 8, as shown in Table. 5.4. Similar to the results for ALS-WR, the *NewDataOnly* approach outperforms the *Fixed* baseline approach in most months. Unlike ALS-WR, a new hyperparameter set is selected once only for the *NewDataNewParams* at month 8. Since then, the *NewDataNewParams* approach slightly outperforms the *NewDataOnly* approach in every month except in month 9 based on RMSE evaluation. In addition, the *NewDataNewParams* approach slightly outperforms the *NewDataOnly* approach in every month based on MAP@20 evaluation. The observations are similar to those under ALS-WR.

	Month	λ	lrate	decay
Online	1 to 7	0.009	0.004	0.98
	8 to 12	0.006	0.004	0.95
Offline	1 to 5	0.006	0.005	0.91
	6 to 12	0.005	0.001	0.78

Table 5.4: Selected SGD Hyperparameters

In this experiment for the online e-commerce site, there are two main observations: Firstly, a hyperparameter set that is not the best choice at the beginning may become the best choice as more data is collected over time. Secondly, conducting model selection continuously as more data is collected improves prediction accuracy in general.

5.5.5 Effectiveness on Offline Stores

Figure 5.6 shows the RMSE evaluation of the consumer preference prediction using ALS-WR for the second year of the offline stores dataset. Same as the result for the online e-commerce dataset, the *NewDataOnly* approach clearly outperforms the *Fixed* baseline approach in every month. For the proposed *NewDataNewParams* approach, a new hyperparameter set is not selected until month 10, and it is the only time a new hyperparameter set is selected, as shown in Table. 5.3. The predictive model is therefore re-trained based on a new λ value from month 10 to month 12. In these 3 months, *NewDataNewParams* always outperforms *NewDataOnly* .

Figure 5.8 shows the MAP@20 evaluation of Top-20 recommendation for ALS-

WR. As shown, the curve representing the Fixed approach fluctuates in a relatively large range while the curves of the other two approaches appear to be relatively smoother. Unlike the online e-commerce environment, the *NewDataOnly* approach and the *NewDataNewParams* approach do not seem to have smoothened the curve when comparing to the *Fixed* approach under the offline stores environment. Both approaches outperform the *Fixed* approach in all months except for month 4 and month 7. For the *NewDataNewParams* approach, a new hyperparameter set is selected for month 10. It outperforms *NewDataOnly* in months 10 and 12 but underperforms in month 11. The advantage of *NewDataNewParams* over *NewDataOnly* is relatively uncertain in this case.

Figure 5.7 and 5.9 show the RMSE and MAP@20 evaluation of prediction using SGD respectively. For RMSE evaluation, the *NewDataOnly* approach outperforms the *Fixed* baseline approach in all months except month 3. A new hyperparameter set is selected once only for the *NewDataNewParams* in month 6, as shown in Table. 5.4, which is much earlier than the case of ALS-WR. Since then, the *NewDataNewParams* approach outperforms the *NewDataOnly* approach in every month except in month 7 based on RMSE. For MAP@20 evaluation, the *NewDataOnly* approach outperforms the *Fixed* baseline approach significantly in all months except month 5. After the predictive model is re-trained with the newly selected hyperparameter set in month 6, the *NewDataNewParams* approach outperforms the *NewDataOnly* approach significantly, except in month 10 where *NewDataNewParams* performs even worse than the baseline.

In this experiment for the offline stores of the retail business, the general outcomes are similar to those of the online e-commerce site. The major difference is that a new hyperparameter set for ALS-WR is not selected until a much later stage in the offline stores environment. Because of this, the advantage of the *NewDataNewParams* approach over the *NewDataNewParams* approach is not conclusive in this single scenario. One possible reason is that the business dynamics or consumer preference does not change much in a year in the offline store environment, therefore a new model selection is not necessary until a later time. Another possible reason, however, is that the model selection technique I apply is not efficient enough to find an optimal hyperparameter set that can improve the performance earlier. Identifying the real reason may become part of my future work.

5.6 Assessment

In this chapter, I have investigated whether re-selecting hyperparameters for model selection repeatedly after the introduction of new data would have any impact on the prediction accuracy of recommender systems. Previous literature assumes that the process of model selection, or hyperparameter optimisation, is needed only once at the initial stage. Therefore, the widely accepted approach nowadays is to re-train the predictive model periodically using the *same* hyperparameter settings after new data is collected.

The main contribution of this chapter is to present a novel study on the effect of optimising hyperparameters continuously in Collaborative Filtering. Three approaches have been evaluated. The first one is $Fixed$, which is a baseline approach that does not re-train the predictive model at all. The second one is $NewDataOnly$, which is the existing state-of-the-art approach used by researchers and practitioners in the industry. In this approach, the predictive model is re-trained periodically with new data. However, the hyperparameter settings are fixed once they are defined at the initial stage. The third approach is my proposed $NewDataNewParams$, which tries to optimise the hyperparameter settings every time before the predictive model is re-trained with new data. I have shown that the proposed continuous model selection approach, i.e. $NewDataNewParams$, improves prediction accuracy of a recommendation system most of the time empirically. In particular, experiments show that improvement has been achieved in both scalable collaborative algorithms (ALS-WR and SGD) that I have implemented. Experiments also show that the proposed approach improves the prediction accuracy of recommender systems for both online e-commerce site and offline retail stores of the retail chain business that I have investigated.

To conclude, in my retail scenarios, not only does my proposed approach outperform the baseline approach $Fixed$ significantly, it also outperforms the existing state-of-the-art approach $NewDataOnly$: For experiments on the online e-commerce dataset, $NewDataNewParams$ has an average of 15.8% and 22.6% improvement over $NewDataOnly$ in terms of MAP@20 for ALS-WR and SGD respectively, with a maximum improvement of 51.9% and 139.5% respectively on the best months. $NewDataNewParams$ also has an average of 1.9% and 2% improvement over $NewDataOnly$ in terms of RMSE for ALS-WR and SGD respectively, with a max-

imum improvement of 7.2% and 8.7% respectively on the best months. For experiments on the offline stores dataset, $NewDataNewParams$ has an average of 2.1% and 22.65% improvement over $NewDataOnly$ in terms of MAP@20 for ALS-WR and SGD respectively, with a maximum improvement of 30% and 126.7% respectively on the best months. $NewDataNewParams$ also has an average of 1% and 5.6% improvement over $NewDataOnly$ in terms of RMSE for ALS-WR and SGD respectively, with a maximum improvement of 4.5% and 22.1% respectively on the best months. I have also discovered that a hyperparameter set that is not selected at the initial stage may become the best choice at a later time.

This chapter presents a new direction to improve the prediction performance of large-scale recommender systems in real-world retail scenarios. It is worth mentioning that a new hyperparameter set can only be selected in much later iteration in some situations in my experiments. Further work needs to be done to determine whether it is caused by the inefficiency of the selected automatic hyperparameter tuning technique, or that it is due to some limitations of the continuous model selection approach in certain cases. Future work should also consider the use of other automatic hyperparameter optimization techniques that are more efficient and less expensive computationally.

Chapter 6

A Distributed Framework with Automated Modeling

One of the biggest challenges for software developers to build real-world predictive applications with machine learning is the steep learning curve of data processing frameworks, learning algorithms and scalable system infrastructure. I present PredictionIO, an open source machine learning architecture that comes with a step-by-step graphical user interface for developers to (i) evaluate, compare and deploy scalable learning algorithms, (ii) tune hyperparameters of algorithms manually or automatically and (iii) evaluate model training status. The system also comes with an Application Programming Interface (API) to communicate with software applications for data collection and prediction retrieval. The whole infrastructure of PredictionIO is horizontally scalable with a distributed computing component based on Hadoop. The demonstration shows a live example and workflows of building real-world predictive applications with the graphical user interface of PredictionIO, from data collection, algorithm tuning and selection, model training and re-training to real-time prediction querying. The contribution of this piece of research work is validated by the industry. PredictionIO has engaged over 5000 developers on Github and is being used by hundreds of applications on production.

6.1 Research Challenge

Machine learning increasingly plays an important role in real-world software applications. More and more data is available everyday from multiple sources, such as mobile and web applications. Some examples include digital assistants (Siri, Google Now); spam detection (Gmail); and product recommendation (Amazon). However, whilst companies such as Google, Apple and Amazon are investing heavily in machine learning development, the application of machine learning remains challenging for most software developers. Applying machine learning into software applications for implementing predictive features is not straightforward. It involves multiple, often quite complex, steps.

Several challenges lead to the steep learning curve associated with machine learning in practical software development. Firstly, there are a large number of data processing frameworks to evaluate, including: Hadoop [111], GraphLab [121] and Spark [122]. Secondly, many algorithms exist for each type of prediction task. For example, collaborative filtering for product recommendation alone has over a few dozen different algorithms. Selecting the right algorithm, also, setting and tuning their parameters provides a challenge. Finally, having a scalable system that supports real-world software applications is very important. Development teams must put a tremendous effort into building scalable machine learning infrastructure to support predictive features. Such a system much support data storage, analysis, distributed processing (such as MapReduce), prediction result storage, caching and retrieval. Building a system to handle such a heavy-load often takes a lot of time, effort and resources.

PredictionIO is an integrated solution for software developers. It is a distributed and horizontally scalable system for applying machine learning to real-world applications, as described in the next section. Furthermore, it has been open sourced and is built upon existing open-source technologies such as the Apache Hadoop and Mahout projects.

The contribution of PredictionIO is twofold. Firstly, PredictionIO presents a novel architecture integrating multiple machine learning processes into one scalable system. Secondly, PredictionIO provides a graphical user interface (GUI) for developers to evaluate, select, tune and deploy the appropriate learning algorithm for their software ap-

plications.

The remainder of this chapter is structured as follows: Section 6.2 outlines the system concept while Section 6.3 describes the system architecture. I present the demonstration scenarios in Section 6.4 and conclude in Section 6.5.

6.2 A Machine Learning Server

A machine learning server works like a database server. The difference is that a database server performs search on data while a machine learning server performs prediction. PredictionIO serves as a single integrated system to provide a series of functionalities for building predictive applications: The system communicates with external software through an Application Programming Interface (API). It collects data and stores them in distributed databases. The system comes with several readily usable scalable learning algorithms already built-in. Users of the system can tune hyperparameters either manually or automatically. The performance of these algorithms can be compared through offline evaluation using existing data based on pre-selected metrics. A GUI is provided to adjust and control these specific tasks. The processing tasks are scheduled and managed by a workflow scheduler. The system also takes prediction queries and return results in real-time.

6.3 System Architecture

6.3.1 Basic System Architecture

Figure 1 shows the conceptual architecture of the system. The system consists of an REST API Server layer which adopts the Representational State Transfer (REST) style standard [123]. This API layer serves as the communication bridge between software applications and the system. The main purposes of the API are to collect input data, receive queries and output prediction results. Software Development Kits (SDKs) of various programming languages are provided to ease the development effort of connecting software applications to the API. These SDKs support asynchronous communication model [124] to make high performance connections. Through the API, input data such as user information, item information and user behaviors are stored to distributed application databases. Whenever MapReduce processing is needed for certain functions, such as algorithm evaluation, hyperparameter tuning and predictive

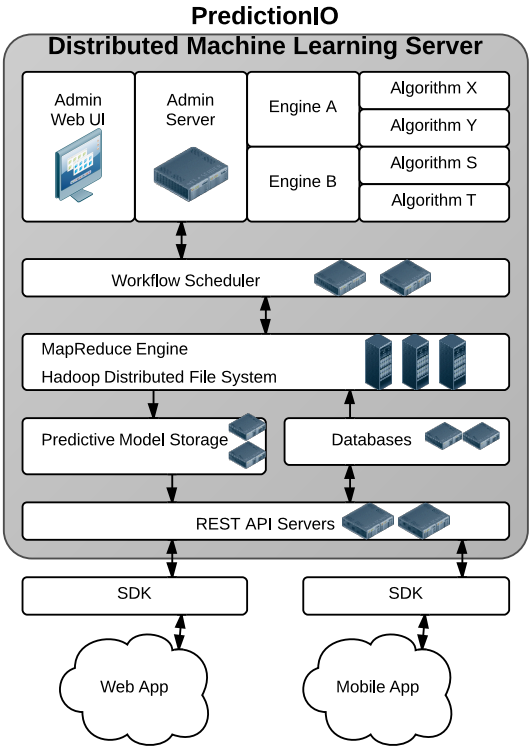

Figure 6.1: System Architecture Diagram

model training, the corresponding data will be transferred from application databases to the Hadoop Distributed File System (HDFS). The time and frequency of such data transfer action is controlled by the Workflow Scheduler. Predictive models are generated, or updated, by the selected learning algorithm and stored in the Prediction Model Storage databases, which is a separate datastore designed for real-time querying. Software applications sent prediction query to the system through SDKs, which will then reach the API layer. API layer will perform real-time querying, filtering and sorting of data on the prediction model databases. The results will be returned to the software through the SDKs. The Admin Server manages engines and algorithms. Each engine represents one specific prediction task, such as item recommendation or classification. Each software application can make use of multiple engines. Under each engine, a number of algorithms are available to be evaluated and deployed. One algorithm has to be deployed for an engine to become functional. The Admin Server also controls the settings of the system. These settings determine how the system operates, for instance, what is the goal of an engine to be optimised, which algorithms should be evaluated and how, what are the constraints of automatic hyperparameter tuning and which algorithm

is deployed for training the final predictive model. Users of the system interact with an secure Admin Web UI to customize these settings.

6.3.2 Highlighted System Features

Offline Evaluation and Algorithm Comparison

The performance of a learning algorithm is evaluated, and thus compared, by offline evaluation using a user-selected metric based on existing collected data. Each engine comes with some built-in metrics. For instance, the item recommendation engine in the system provides Mean Average Precision (MAP@K) and Root Mean Squared Error (RMSE) metrics [9]. For MAP@K, the goal is to evaluate the average precision of the predicted recommendation list for each consumer. Suppose a consumer has purchased n products in the test dataset and the system can recommend up to K products to this consumer. The average precision score at K (ap@K) is:

$$ap@K = \sum_{k=1}^{K} P(k)/min(n, K) \tag{6.1}$$

where $P(k) = 0$ if the consumer has not purchased the k-th product of the recommended list in the test dataset and $P(k) = k$ if otherwise. The mean average precision for M consumers at K (MAP@K), is the average of the average precision of each consumer, which is defined as:

$$MAP@K = \sum_{m=1}^{M} ap@K/M \tag{6.2}$$

The higher the MAP@K score, the better the recommender system performs. For RMSE, it aims to measure the prediction error of preference scores directly. Recommender systems predict consumer preferences based on predictive models that are built for predicting preference scores of unseen consumer-product pairs. A higher preference score suggests that the consumer is more likely to purchase the product. The predicted scores will be evaluated against test sets that contain the actual preference given by consumers previously. RMSE can be used as the prediction error metric:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_i - \hat{r}_i)^2} \tag{6.3}$$

, where n is the total number of pairs in a test set, r_i is the true score which is always 1 in this case and $\hat{r}_i$ is the predicted score. The lower the RMSE score, the more accurate

the predictive model is. Both RMSE and MAP@K are computed in the distributed Hadoop environment.

Automatic Hyperparameter Tuning

Most machine learning algorithms come with some tunable hyperparameters such as learning rate and regularization. Selecting appropriate settings for these hyperparameters is crucial for the accuracy of prediction. Hyperparameter tuning can be defined formally as an optimisation problem itself, which can be written as:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, C(\boldsymbol{x}_v, \boldsymbol{y}_v, M(\boldsymbol{x}_{tr}, \boldsymbol{y}_{tr}, \theta))$$

and the performance of a set of hyperparameter settings can be evaluated by a cost function:

$$C(\boldsymbol{x}_{test}, \boldsymbol{y}_{test}, M(\boldsymbol{x}_{tr+v}, \boldsymbol{y}_{tr+v}, \theta^*))$$

where C is the cost function, M is the modeling function, θ is a hyperparameter set, $\boldsymbol{x}_{tr}$ is the feature vectors of the training dataset, $\boldsymbol{x}_v$ is the feature vectors of the validation dataset, $\boldsymbol{x}_{test}$ is the feature vectors of the test dataset and $\boldsymbol{y}_{tr}$, $\boldsymbol{y}_v$ and $\boldsymbol{y}_{test}$ are the vectors of labels of training, validation and test datasets respectively. Whilst users can set hyperparameters manually through the GUI, the system provides automatic tuning methods such as Random Search [106], which are computed in the distributed Hadoop environment.

Real-time Querying

The prediction results computed by MapReduce processes in batch mode are stored in the Predictive Model Storage, which is a non-relational database with non-unique indexing on the user ID and prediction score fields and geospatial indexing on the location information of items. These indexes enable the REST API Server to perform real-time searching and ranking based on inputing queries. With the implementation of geospatial indexes, real-time location-aware predictive features can be built with the system.

6.4 Demonstration Scenarios

The proposed system eases the challenges of building predictive software applications with machine learning. I demonstrate the system in two parts: 1) The GUI of the system; 2) Source code and live usage in software development.

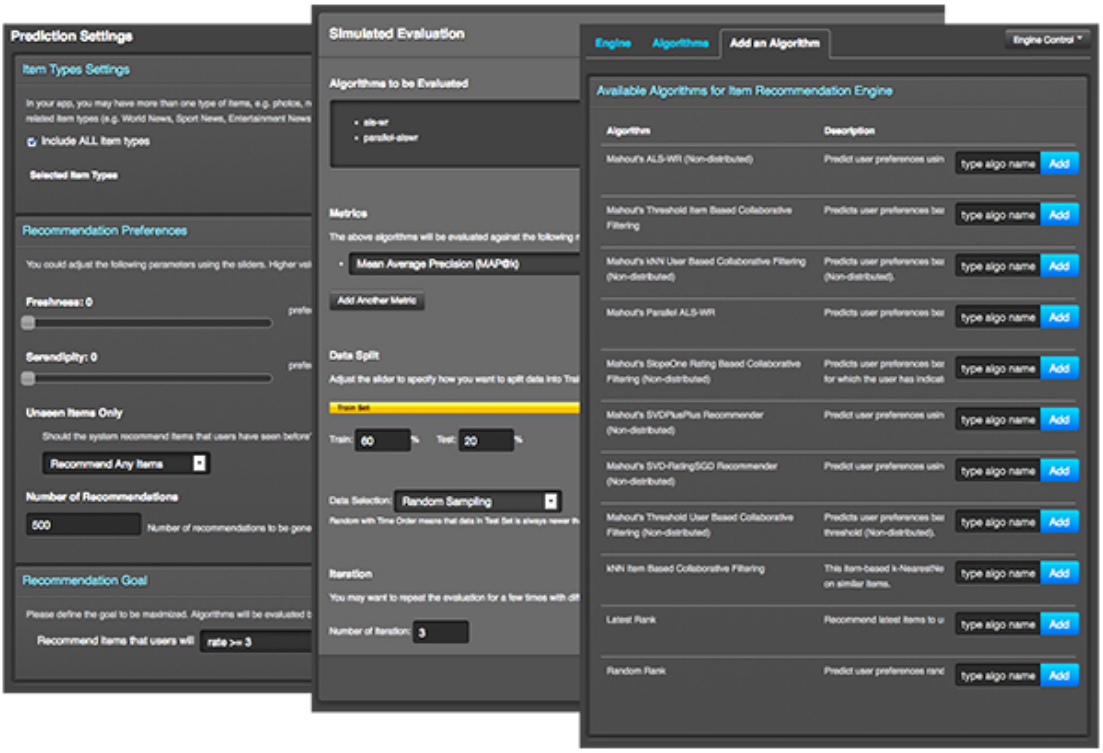

Figure 6.2: Graphical User Interface

The first part is a step-by-step navigation of the graphical user interface (GUI) to (i) create an engine, (ii) select multiple available learning algorithms within the engine, (iii) tune the hyperparameters of the these algorithms, some with manual tuning, some with automatic tuning and (iv) view an evaluation report in order to compare the performance of the selected algorithms, and finally (v) deploy the best performing algorithm. Figure 2 illustrates some of the graphical user interfaces. For this part of the demo, the small MovieLens dataset [90] which consists 1682 movies, 943 users and 100,000 ratings is imported into the system in advance. A small dataset is chosen purely to avoid lengthy processing time for demonstration purpose.

The second part is a showcase of using PredictionIO to build a location-aware restaurant recommendation application. I provide sample source code which illustrates core functionality including: importing new users; importing new restaurants; tracking user behaviors (such as ratings); and location-based prediction retrieval. The data input and output format will also be shown.

6.5 Assessment

This chapter has outlined the concept, system architecture and demonstration scenarios for PredictionIO, a distributed machine learning server for practical software development.

The contribution of PredictionIO is twofold. Firstly, PredictionIO presents a novel architecture integrating multiple machine learning processes into one scalable system. Secondly, PredictionIO provides a GUI for developers to evaluate, select, tune and deploy the appropriate learning algorithm for their own software applications. The contribution of this research work is validated by the industry participation of its open source project on Github [125]. PredictionIO has engaged over 5000 developers on Github as of this assessment is made. It is also being implemented by hundreds of applications on production environments.

In the future, I hope to explore the possibility of providing additional functionality, such as feature selection, and online evaluation, both important steps in the machine learning process. Furthermore, support for (including non-parallel) extended or custom algorithms. I look forward to feedback and contributions from academia and wider industry.

6.6 Screenshots

Screenshots of the PredictionIO graphical user interface are shown below.

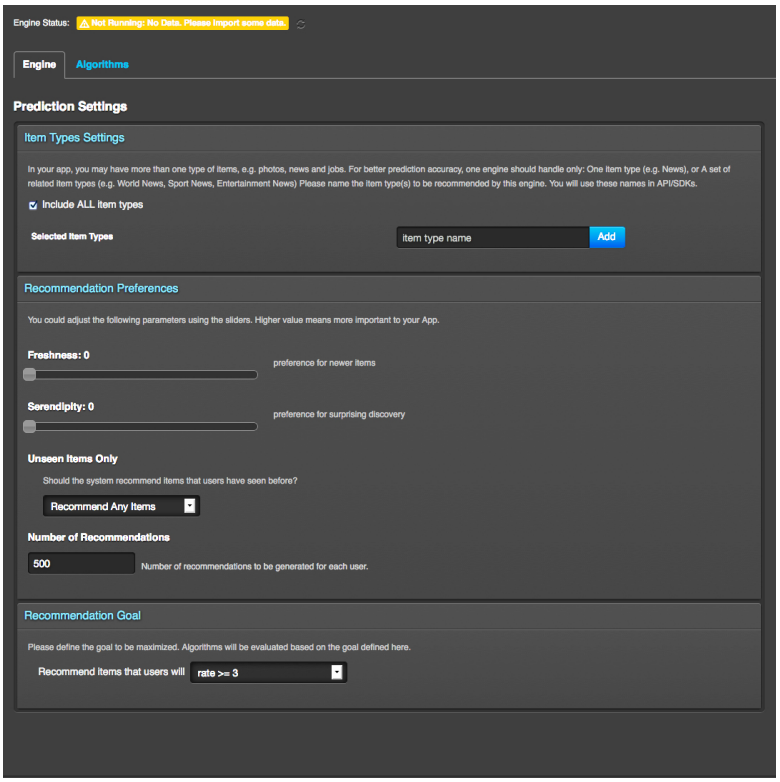

Figure 6.3: Prediction Engine Configuration

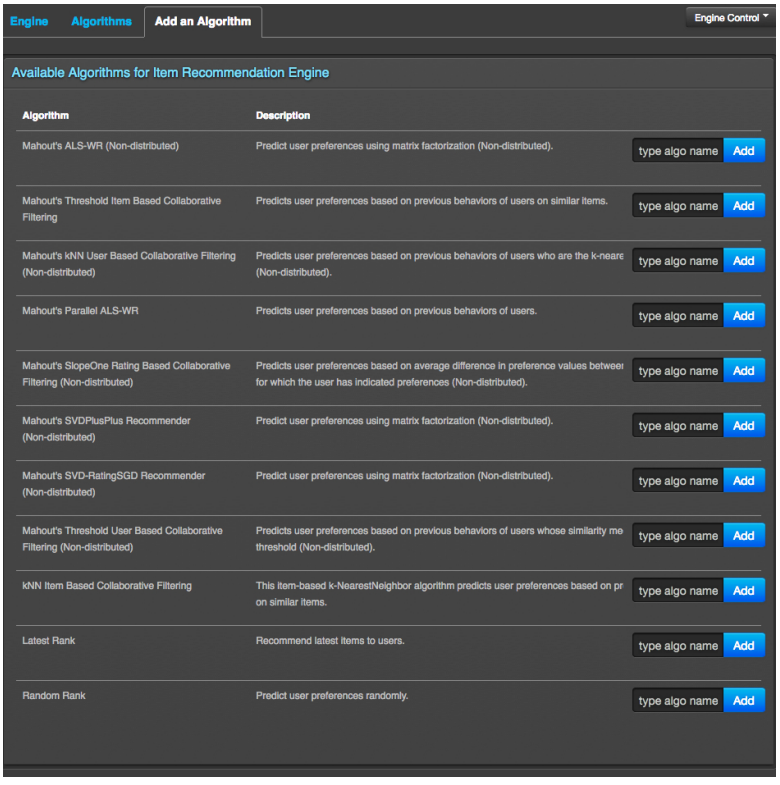

Figure 6.4: Algorithm Selection

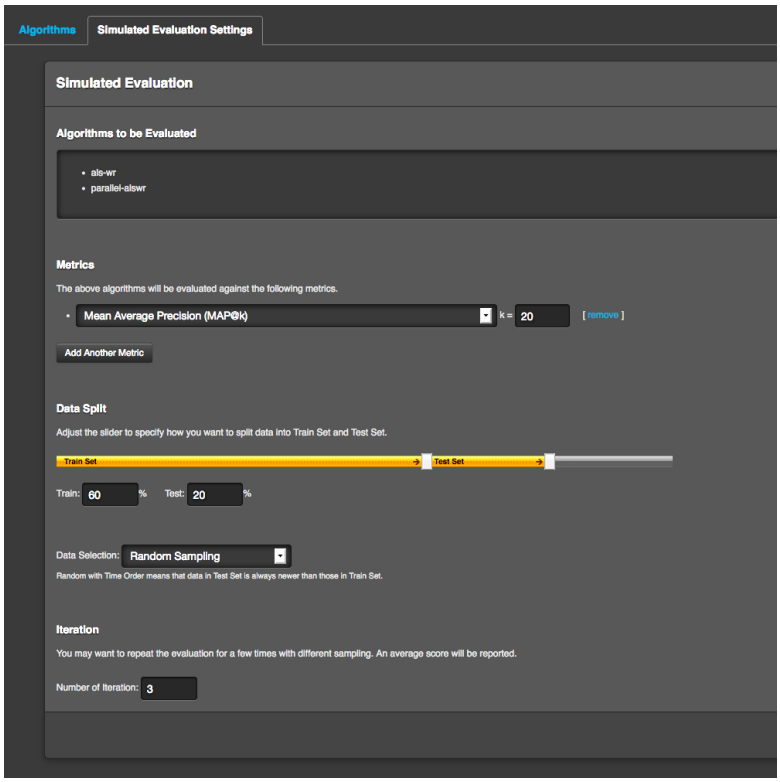

Figure 6.5: Offline Evaluation

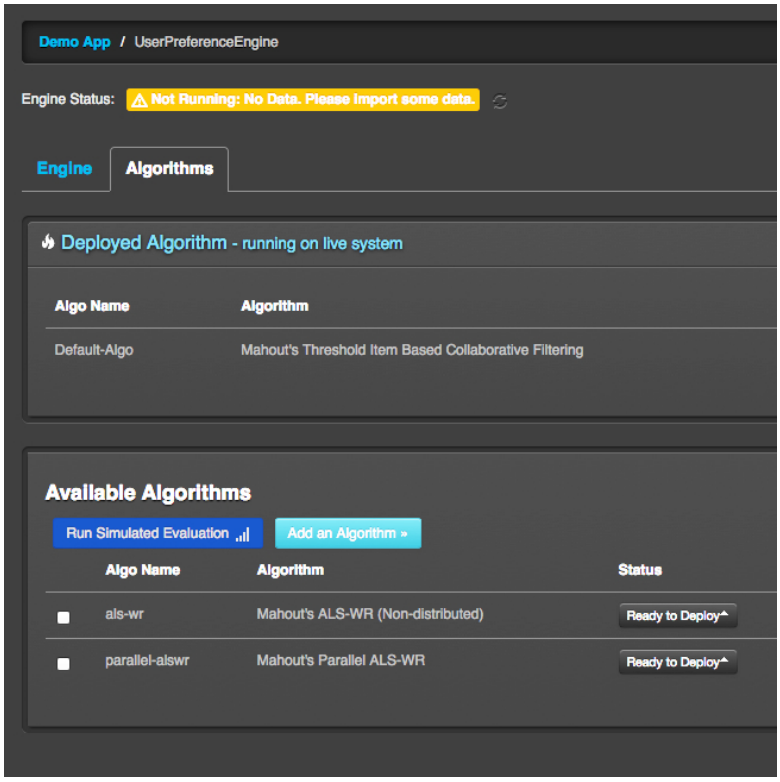

Figure 6.6: Algorithm Deployment

Chapter 7

Conclusion and Future Work

7.1 Summary of Contributions

The contribution of this thesis to science is to address scientific challenges of conducting predictive analytics for consumers' purchase preferences in real world operational settings. The proposed computational statistics and machine learning methodologies can 1) analyze correlation of factors in noisy non-experimental settings; 2) analyze how decisions of resources allocation in recommendation affect business gains; 3) improve prediction accuracy in changing business environments.

Chapter 2 introduces the general context of consumer preference prediction in the retail industry. I present the motivation for retail businesses to predict customer preference and distinguish the difference between prediction and nudge. The difference between conducting correlation and causation analysis in experimental environment and in non-experimental retail environment is also illustrated. I further introduce the type of data retail businesses can use to predict customer preference. An introduction and a detailed review of some latest research of recommender systems, a common application of consumer preference prediction in the retail industry, is also presented.

Chapter 3 derives a contextual factor from consumers' recent online browsing behaviours on the retailers' website for the prediction of their offline in-store purchase. Some retailers aim to improve the accuracy of customer preference prediction in Recommender Systems. In retail and many other business environments, extra contextual factors are sometimes available. The use of some contextual factors can improve the accuracy of prediction in some situations. The relevance of some factors, however, is controversial in the industry. For instance, consumers' recent online exposure to prod-

ucts can decrease the chance of in-store purchase as consumers may choose to purchase products online. On the other hand, online exposure can be seen as an evidence of consumers' preference on products, which implies a higher chance of in-store purchase. The understanding of true relevance is important for product recommendation in-store in this case. The research challenge is how to evaluate the relevance of potential factors for prediction. I present a statistical approach to conduct a preliminary analysis on the relevance between this factor and the offline purchase decisions on brands using odds ratio and stratified analysis techniques. Unlike machine learning approaches, the proposed methodology can analyze the relevancy of factors even before a recommender system is built. My work further shows that the relevance of a contextual factor depends on extraneous attributes, such as consumers' ages and gender. This contribution serves as a preliminary step to analyze relevant contextual factors for building context-aware recommender systems.

The contribution of Chapter 4 is the presentation of a two-stage batch solution to approximately optimise a practical objective function of group buying systems with operational constraints for new items under the cold-start situation. In some retail scenario, there is a series of new items and only a limited number of items can be recommended to a single user at a time. For example, in a group buying case, retailers are only allowed to send every potential buyer a fixed number of recommendations per day. Thus, within this limited number of recommendation opportunities (resources), the question is how to balance the exploration (estimating the prediction model) and exploitation (utilizing the model) in order to maximise the overall effectiveness of the system, e.g., the overall positive response rate. Furthermore, there are some business operational constraints. For instance, a deal of an item would become valid only if the minimum purchase requirement is fulfilled. Experiments show that my solution significantly increases the response rate comparing to the non-personalized recommendation. I discover that using the popular D-Optimal design approach for exploration does not provide significant advantage over using the random selection approach in some cases. It is explained by my experiments which show that the D-Optimal design prefers to use the quota of resources of active buyers during the exploration stage, which affects the performance of the exploitation stage negatively.

Chapter 5 considers a practical scenario where the business environment is chang-

ing over time. Retail business environment changes rapidly. Product prices, for instance, may be changing frequently and new products appear on the market regularly. Simply re-training the same predictive model with the addition of new data may not be sufficient to accommodate these changes. My hypothesis is that a recommender system model may render less accurate or even invalid results, as the business dynamic changes over time. I presents an approach to automatically re-select the model by continuously optimising the hyperparameter settings to keep up with the changes. The new approach applies an automatic hyperparameter optimisation technique in collaborative filtering algorithms while the system collects more and more new data. Experiments have been conducted in a real world large retail dataset to challenge traditional assumption that a one-off initial model selection is sufficient. In particular, the proposed approach has been compared with a baseline approach and a widely used approach with two scalable collaborative filtering algorithms. The evaluations of this experiments are based on a 2-year real purchase transaction dataset of a large retail chain business, both its online e-commerce site and its offline retail stores. It is demonstrated that continuous model selection can effectively improve the prediction accuracy of a recommender system. This contribution presents a new direction in improving the prediction performance of a large-scale recommender system in a real-world retail scenario.

Chapter 6 addresses one of the biggest challenges for software developers to build real-world predictive applications with machine learning. It is the steep learning curve of data processing frameworks, learning algorithms and scalable system infrastructure. I present PredictionIO, an open source machine learning architecture that comes with a step-by-step graphical user interface for developers to (i) evaluate, compare and deploy scalable learning algorithms, (ii) tune hyperparameters of algorithms manually or automatically and (iii) evaluate model training status. The system also comes with an Application Programming Interface (API) to communicate with software applications for data collection and prediction retrieval. The demonstration shows a live example and workflows of building real-world predictive applications with the graphical user interface of PredictionIO, from data collection, algorithm tuning and selection, model training and re-training to real-time prediction querying. The contribution of this piece of research work is validated by the industry. PredictionIO has engaged over 5000 developers on Github and is being used by hundreds of applications on production.

In conclusion, this thesis contributes to the existing literature in a number of ways. First, this thesis presents an overall of state-of-the-art researches for solving real world retail prediction problems. Second, this research examines the influence of confounding factors in consumer preference prediction, a topic which has received little attention in empirical literature. Third, the analysis of real retail datasets, which contain both online and in-store purchase transactions of more than ten millions of consumers, is novel. Forth, this research identifies the correlating factors between consumer online browsing and in-store purchase behaviours, which has not been studied in the past as far as I know. Fifth research examines the influence of the balance between exploration and exploitation of new product profiles on maximising business gains. Sixth, this research proves that random selection surprisingly outperforms D-optimal experimental design in some retail cases. Seventh, this research shows that simply re-training predictive model with new data is insufficient to adapt to changing business environments in some cases. Finally, this thesis presents the implementation of an adaptive model selection approach by using automatic hyperparameter tuning techniques.

7.2 Future Work

This research pursued its objectives within the pre-defined scope successfully. It also inspires future research directions based on existing work. A number of subsequence topics may worth further investigations as a continuation of this research. First, future research work can go beyond purely predicting customer preference. New methodologies to influence, or nudge, customers' choices can be investigated. Second, future work may make use of the contextual information identified by the stratified odds ratio analysis conducted in my work to improve the prediction accuracy. Third, a context-aware recommender system that uses correlated online behaviors to predict and recommend products to customers in physical stores can be investigated in the future. Forth, my experiments of exploration and exploitation analysis can be conducted in real retail datasets, when they become available in the research community. Fifth, I may evaluate the effectiveness of more techniques for speeding up the learning process during the exploration stage for the cold-start product problem in retail. Finally, future work may involve the investigation of the reasons a new hyperparameter set for ALS-WR is not selected until a much later stage in the offline stores environment in my Continuous

Model Selection experiment.

Appendix A

Conceptual Data Overview in Retail

The dataset of a retailer that operates both an e-commerce website and physical stores can usually be conceptualized as below.

Store Data.

This is a list of stores with associated address information. This can be linked to the transactions data using Store ID.

Customer Data.

This is a list of all customers . This can be linked to the transactions data using Customer ID. It may contain anything about the customer, for example:

- Customer ID - A unique identifier for a customer which you can use to match to transactions to see what each person bought
- Loyalty Card ID - If the retailer runs a loyalty program, it is the unique card identifier for the customer
- Age
- Cluster Name - Retailer may assign a customer to a pre-defined cluster that represents some sort of preference or attributes using data mining or Machine Learning technique
- Location - Address or area the customer lives
- Gender
- Age or range of age

Item Data.

This is a list of all of the products. This can be linked to the transactions data using Item ID.

Transactions Data.

This contains all of the actual purchases made by customers. A breakdown of file structure may look like below:

- Customer ID Unique identifier of a customer
- Loyalty Card ID The number of the Loyalty Card that was presented during the transaction.
- Time Transaction Date The date the transaction took place.
- Till Transaction Time The time at which the transaction took place.
- Item ID Unique identifier for the item bought.
- Store ID The store at which the transaction took place if it's in a physical store
- Sales with Tax The value of the sale (inclusive of tax).
- Sales without Tax The value of the sale (exclusive of tax).
- Sales Units How many of that particular item was bought in that transaction.

Online Browsing Data

This contains all of the browsing record of a customer who has signed up and identified himself or herself on the e-commerce site. The structure may look like below:

- Customer ID Unique identifier of a customer
- Item ID Unique identifier for the item page browsed.
- Time Date The timestamp of the browsing behavior

Appendix B

Stratified odds ratios analysis of 6 product brands

This is a listing of all figures of odds ratios analysis of 6 product brands of a retailer, stratified by age, gender and cluster. At the end, it comes with an example of odds ratios analysis with the confidence interval range.

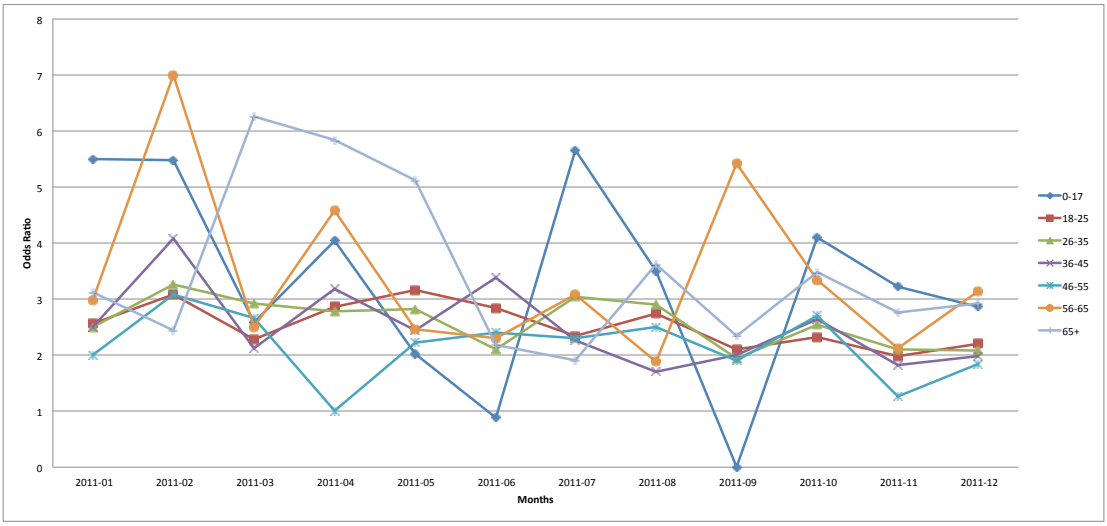

Figure B.1: Odds Ratio by Age (Brand A)

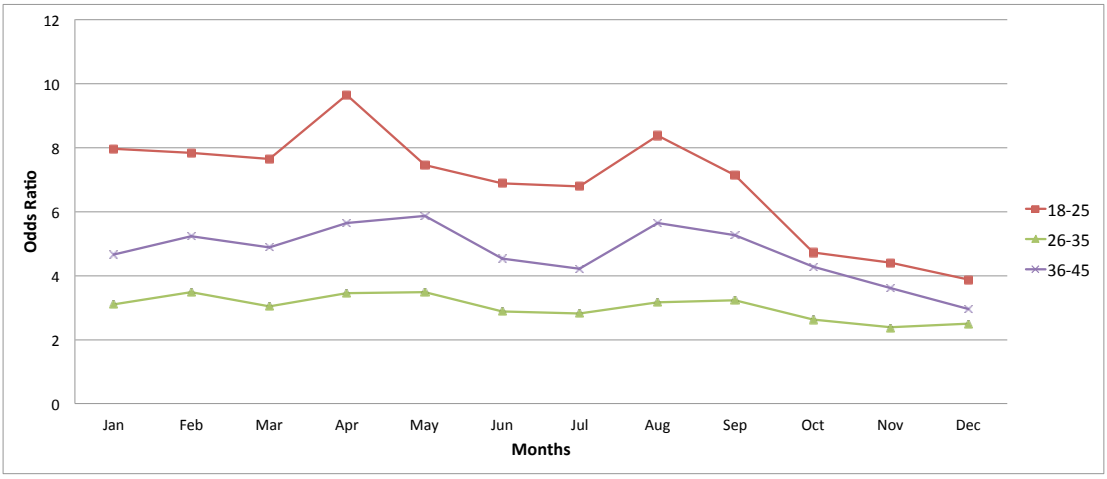

Figure B.2: Odds Ratio by Age (Brand B)

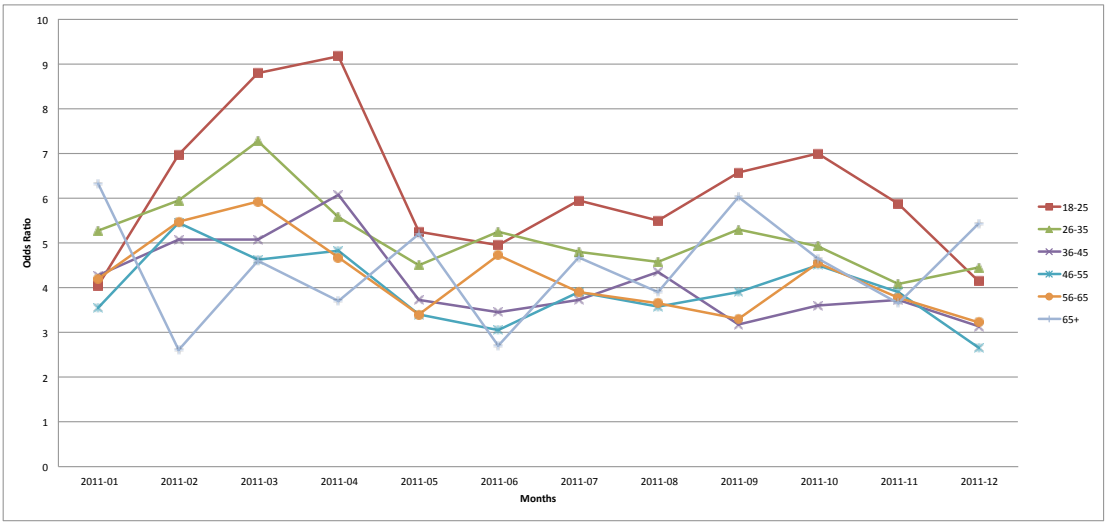

Figure B.3: Odds Ratio by Age (Brand C)

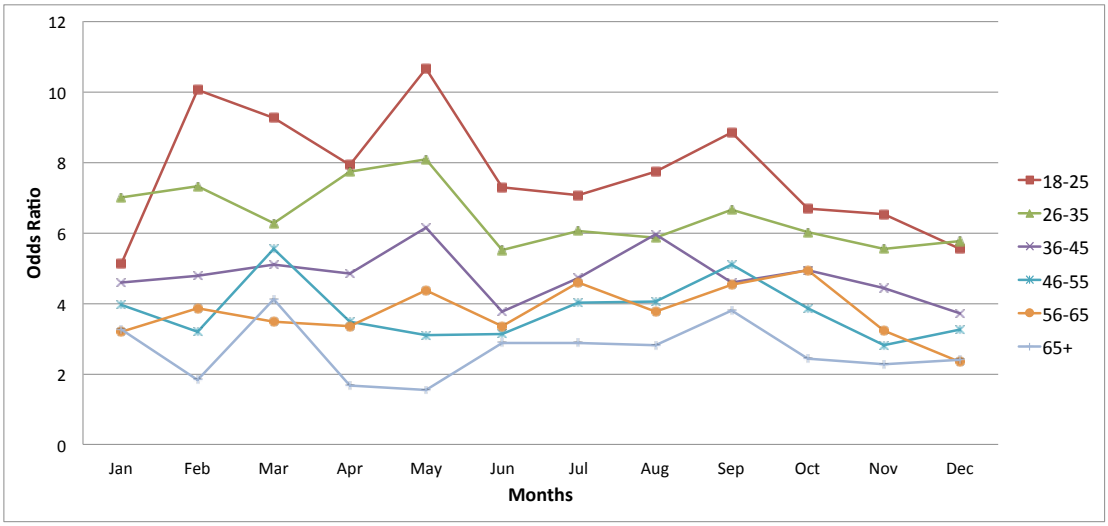

Figure B.4: Odds Ratio by Age (Brand D)

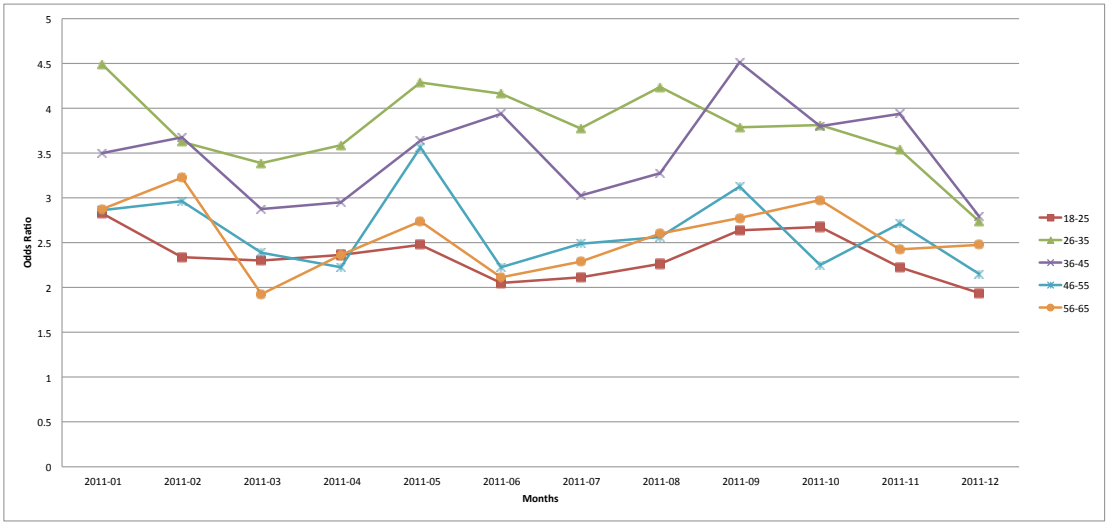

Figure B.5: Odds Ratio by Age (Brand E)

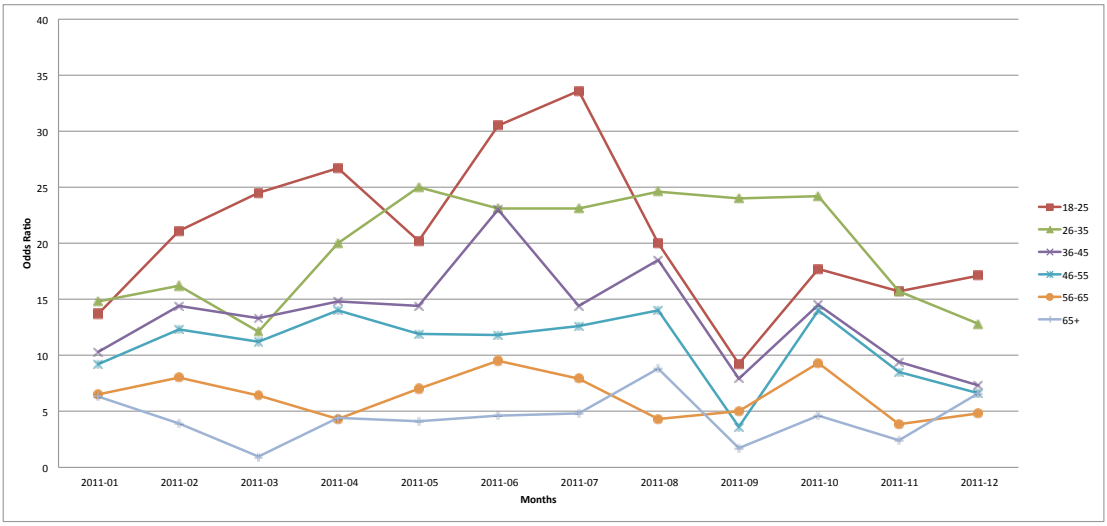

Figure B.6: Odds Ratio by Age (Brand F)

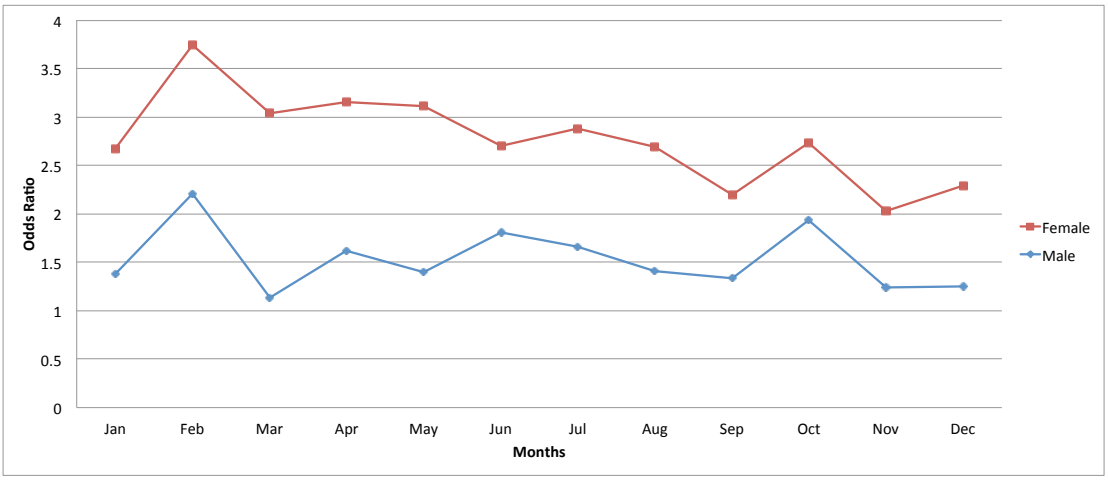

Figure B.7: Odds Ratio by Gender (Brand A)

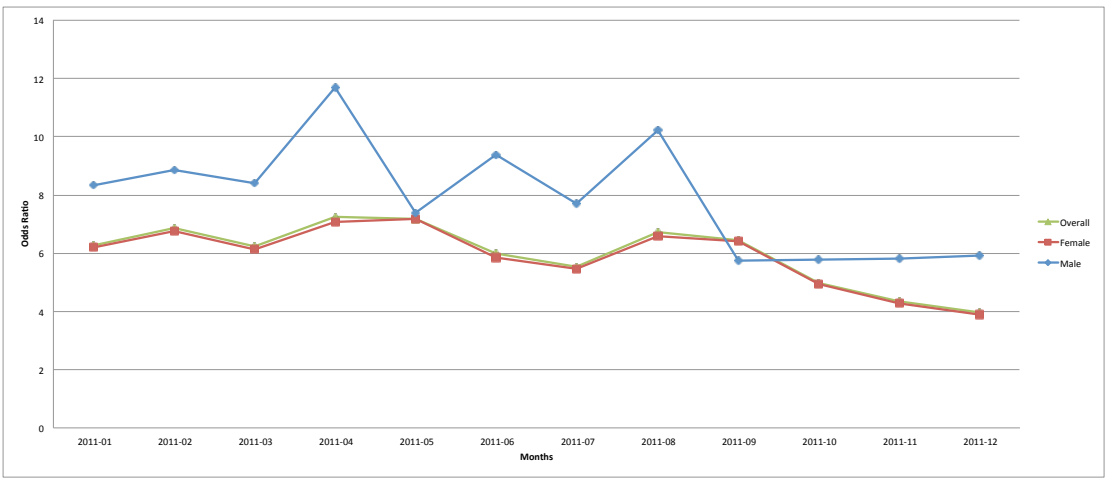

Figure B.8: Odds Ratio by Gender (Brand B)

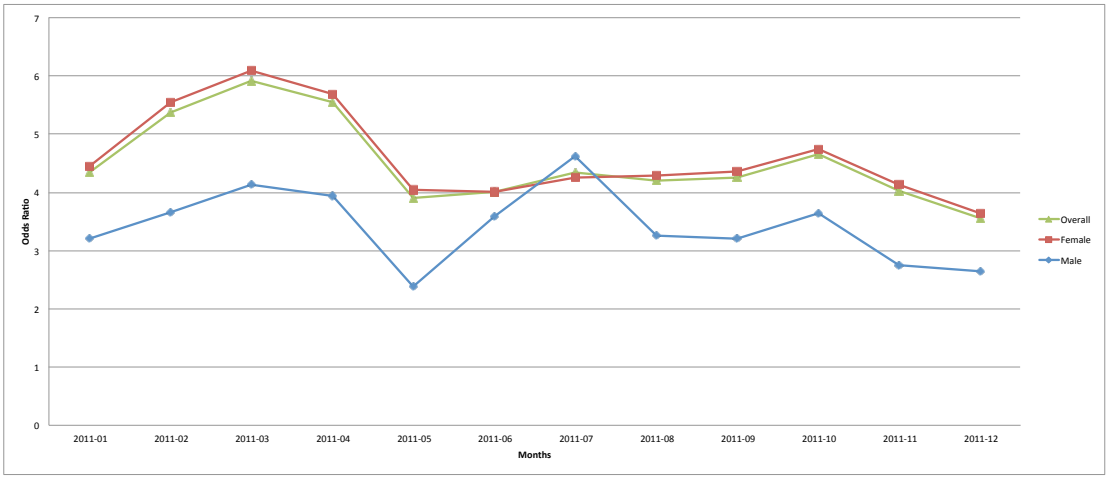

Figure B.9: Odds Ratio by Gender (Brand C)

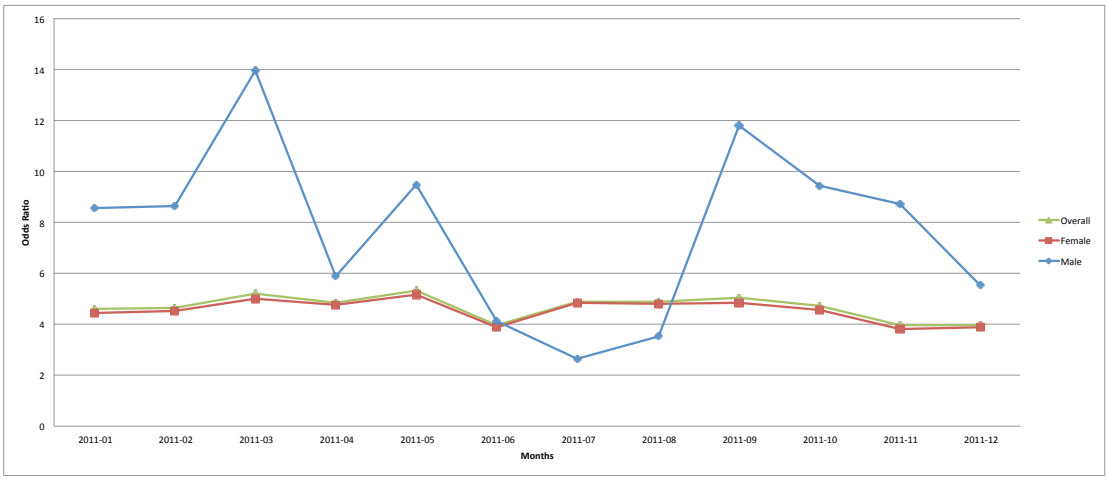

Figure B.10: Odds Ratio by Gender (Brand D)

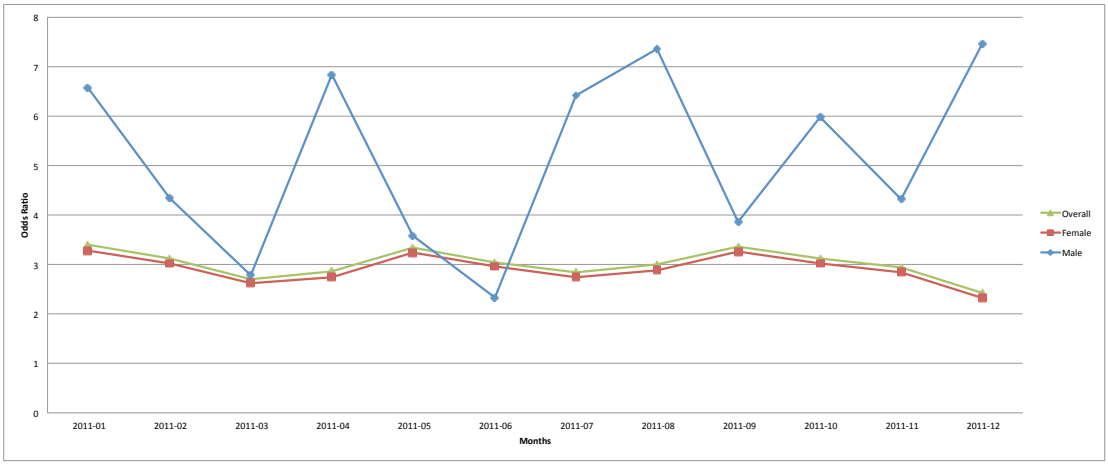

Figure B.11: Odds Ratio by Gender (Brand E)

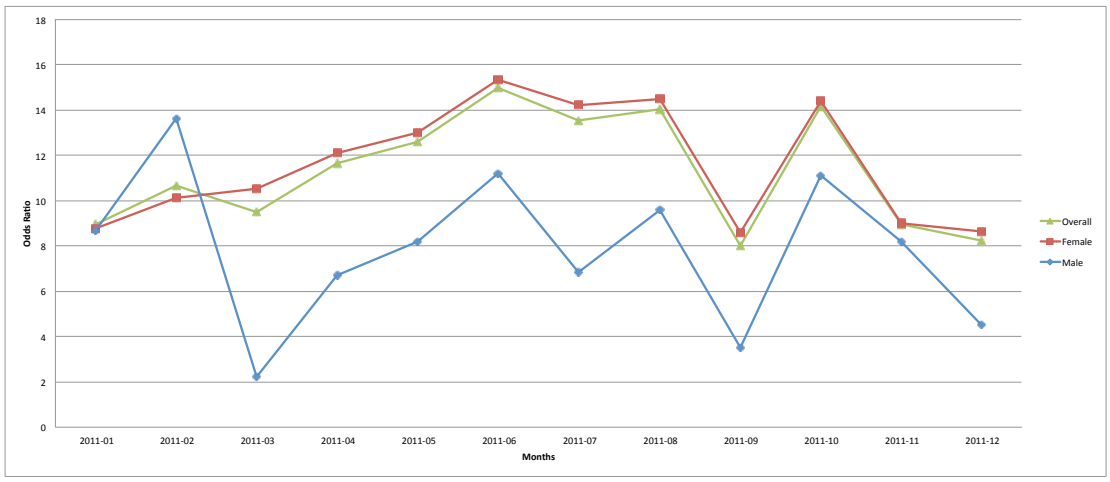

Figure B.12: Odds Ratio by Gender (Brand F)

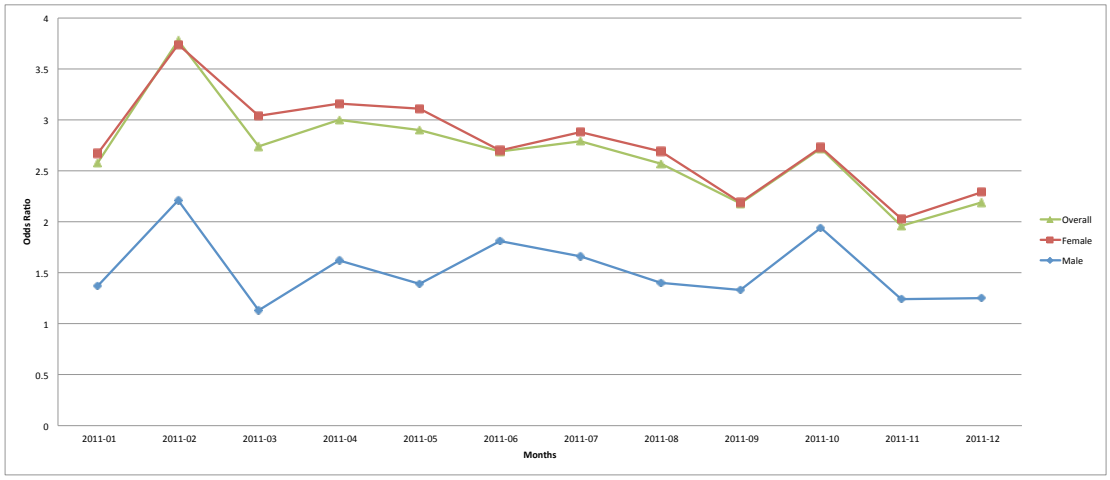

Figure B.13: Odds Ratio by Cluster (Brand A)

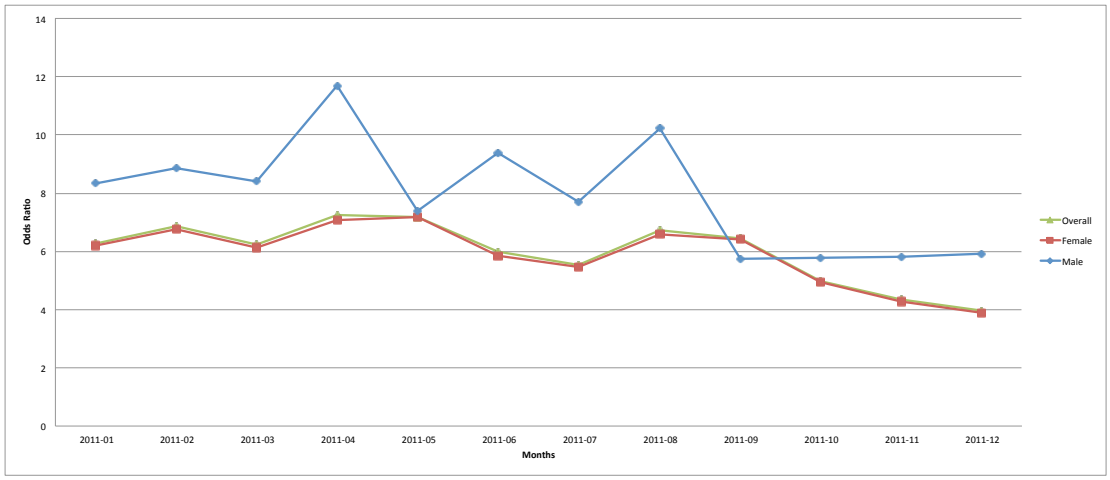

Figure B.14: Odds Ratio by Cluster (Brand B)

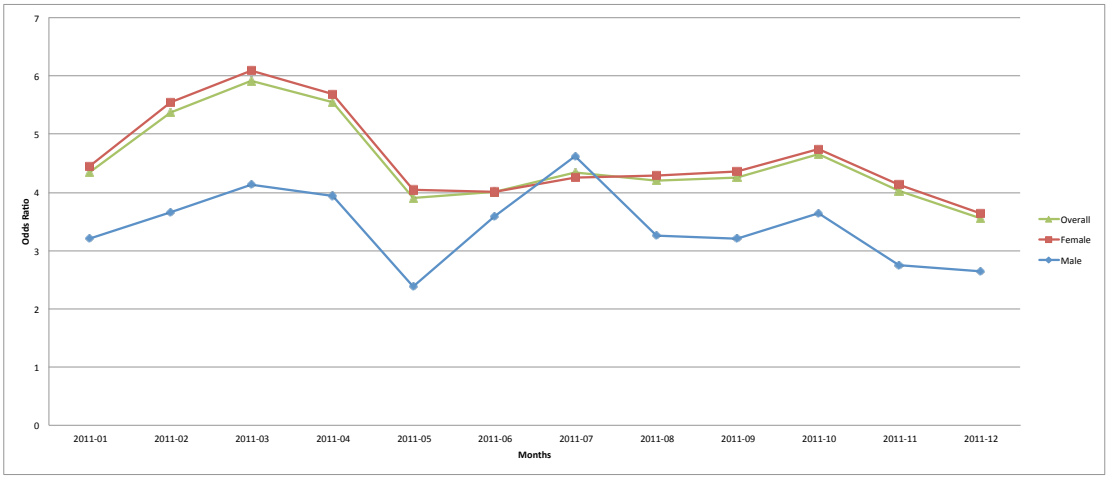

Figure B.15: Odds Ratio by Cluster (Brand C)

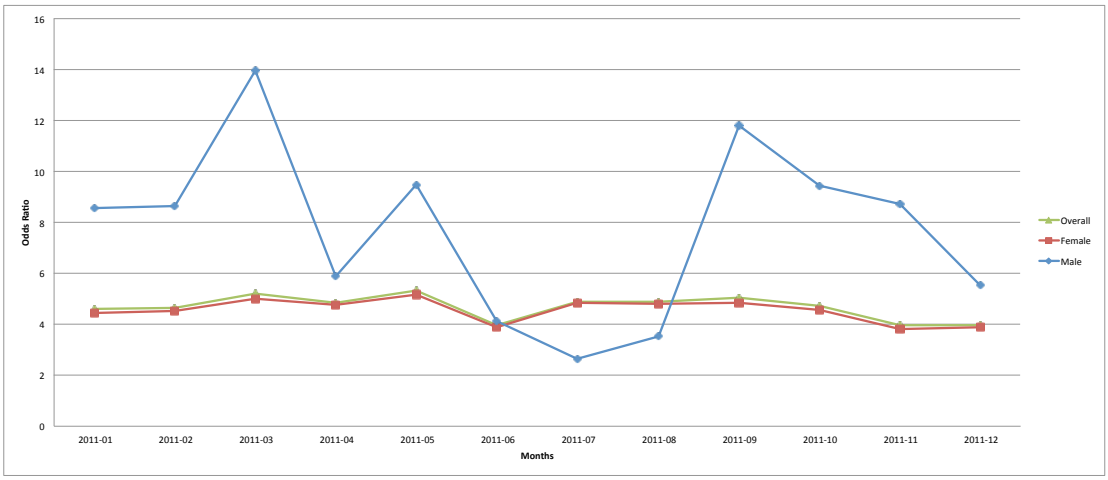

Figure B.16: Odds Ratio by Cluster (Brand D)

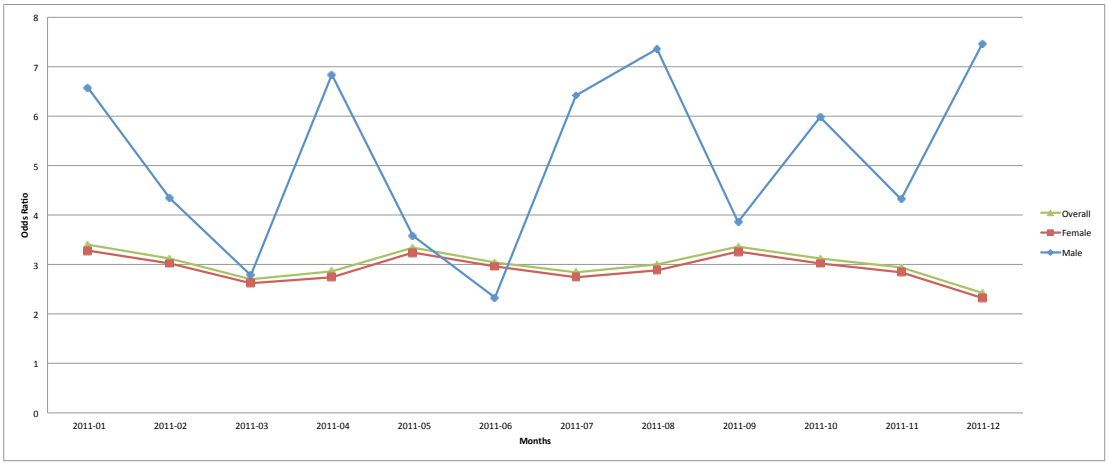

Figure B.17: Odds Ratio by Cluster (Brand E)

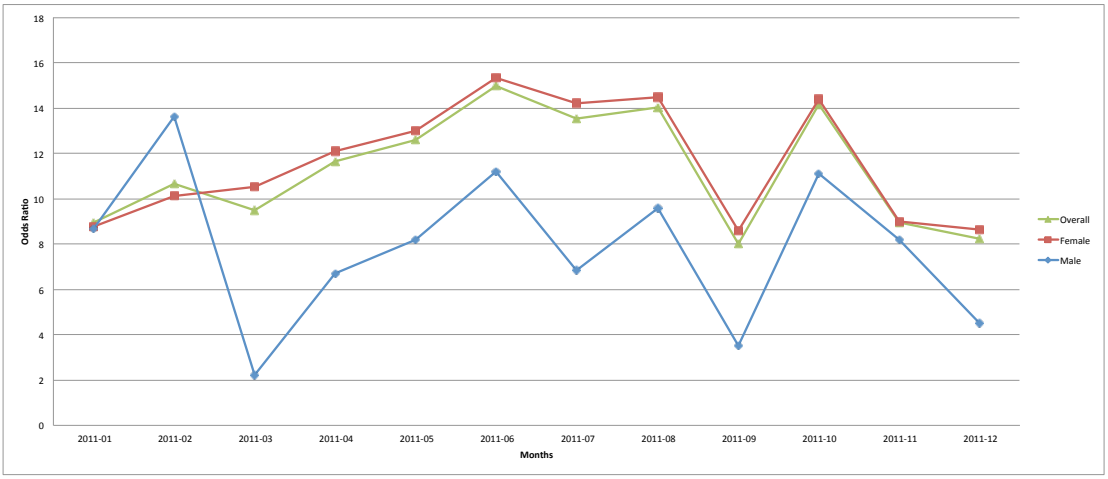

Figure B.18: Odds Ratio by Cluster (Brand F)

	Male			Female		
	Odds Ratio	95% CI High	95% CI Low	Odds Ratio	95% CI High	95% CI Low
Jan	1.269962138	1.595661248	1.01074325	2.883892964	3.268348201	2.544661131
Feb	2.175974456	2.703604834	1.75131542	4.141109783	4.724810492	3.629519166
Mar	1.075979162	1.515938088	0.7637061	3.384170634	4.004312515	2.860069197
Apr	1.528991405	2.201885401	1.06173315	3.470937665	4.09095965	2.944885627
May	1.368557619	1.939915464	0.96548019	3.442833095	3.990713085	2.970170861
Jun	1.714892496	2.38118606	1.23503842	2.926783434	3.437177646	2.492178803
Jul	1.594930417	2.197206217	1.15774433	3.115046797	3.622015703	2.679037681
Aug	1.356471828	1.868973063	0.98450633	2.981639761	3.488839682	2.548175461
Sep	1.297948106	1.894141832	0.88941032	2.374923196	2.883559421	1.956006228
Oct	2.050142238	2.841664745	1.47909186	3.132246294	3.600844171	2.724629664
Nov	1.224950411	1.712335424	0.87629064	2.235342788	2.506111847	1.993828563
Dec	1.223916687	1.614457907	0.92784832	2.455755849	2.668559751	2.259921963
Annualised	1.470591012	1.603989101	1.34828717	2.94480703	3.061282436	2.832763269

Figure B.19: Odds Ratio Numbers with Confidence Interval by Gender (Brand A)

Appendix C

Definition of Machine Learning and Relevant Terminology

Machine Learning is a term used not only within the research domain but also in industries such as retail, government, healthcare, marketing and education. Machine Learning, along with other terms such as Predictive Analytics, Data Analytics, Classification, Pattern Recognition and Data Science, are often casually, if not wrongly, used without much interpretation. In addition, there are more traditionally used terms such as Statistical Analysis, Data Warehousing, Data Mining, Knowledge Discovery, Artificial Intelligence and Business Intelligence. The line between these terms is often fine even for people working in the field. Please note that some related terms such as Data Visualization, Natural Language Processing and Signal Processing in Electrical Engineering are more self-explanatory. Therefore, I am not going to elaborate on them here.

Machine learning, in short, refers to computers learning to predict from data. Machine Learning has empowered many smart applications. For example, Apple Siri learns from data to predict the meaning of human voice and the desired answers or actions to be performed. Facebook photo album learns from data to predict (or recognize) faces to be tagged in photos. LinkedIn learns from data to predict who you want to connect with. Google driverless car learns from data to predict the appropriate driving actions. So the goal of Machine Learning is for a computer to predict something. While predicting a future event is one of the obvious scenarios, it also encompasses the prediction of events or things that are unknown to the computer, i.e. something you have not inputted or programmed into it. Arthur Samuel describes it way back in 1959:

"(Machine Learning is a) field of study that gives computers the ability to learn without being explicitly programmed".

Learning implies improvement through gaining experience or knowledge. Performance measurement is an essential component in Machine Learning. Mitchell (1997) emphasizes the learning requirement: "(Machine Learning) computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience". Without a meaningful evaluation metric, whether it is quantitative or qualitative, the goal of Machine Learning can hardly be achieved. Data are the known properties, experience or knowledge that the computer learns from. Data used for training prediction model in Machine Learning is called training data. Data used for evaluating performance during the training stage is called validation data. Data used for evaluating the final performance is called test data.

In summary, Machine Learning is about attempting to teach computers to predict future, or otherwise unknown events by applying computer science or statistics techniques to analyze existing data. It can be seen as a transformation from existing data to improve insights about the unknown. Machine Learning is often associated with other frequently used terms such as predictive analytics, big data, data mining, business intelligence and data science. Let us take a look at them briefly.

Predictive Analytics

An equivalent, but less formal, term for Machine Learning is Predictive Analytics. It is widely used in business. As Eric Siegel states in his new book Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die: "(Predictive Analytics is) technology that learns from experience [i.e. data] to predict the future behaviour of individuals in order to drive better decisions".

Big Data

Big data is a fuel of Machine Learning development. O'Reilly summarizes the characteristics of Big Data with Volume, Velocity and Variety [126]. Gartner similarly states that "data is high-volume, -velocity and -variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making." [127]. Organizations collect a large variety of data in huge volume with high

velocity from different sources such as smartphones and social media. The availability of big data poses new challenges and opportunities at the same time. It stretches the limits of scalability, computational power, processing speed and flexibility of existing Machine Learning technology.

Supervised Learning

Supervised Learning is one of the popular task types in Machine Learning. The task for the computer is to take an input and then predict an output based on what it has learnt from the training data. Training data consists of pairs of input and correct output, which are called labeled data. When the output is a discrete variable, e.g. predict whether tomorrow will be 1) sunny; 2) cloudy; or 3) raining, it is called a Classification problem. When the output is a continuous variable, e.g. predict the temperature of tomorrow, it is called a Regression problem. Classification is also called Discriminant Analysis in statistics and Pattern Recognition in engineering.

Unsupervised Learning

Unsupervised Learning is another popular task type in Machine Learning. The task for the computer is to discover structures or patterns in the training data, and then to predict which one the input belongs to. The training data consists of only input but not any example output, which is called unlabeled data. The searching for structure with Unsupervised Learning in Machine Learning is called Density Estimation in statistics.

Statistical Analysis

As mentioned, Machine Learning is about teaching computers to predict the unknown by learning from known data, and this differs from the goal of Statistics. Statistics, or Statistical Analysis, is about teaching humans what has happened or what is happening by looking at data, in order to make better decisions. Machine Learning is a learning process to build up reusable knowledge for a computer to perform tasks in the future. Statistical Analysis is an exploratory process to discover previously unknown trends, patterns or other knowledge about the data provided, and the end result should be interpretable by humans. Statisticians also have their own set of language. For example, as Ethem Alpaydm (2004) mentions, “statistics, going from particular observations to general descriptions is called inference and learning is called estimation.”. Despite the

difference in terms of their goals, Statistical Analysis and Machine Learning overlap a lot. They both analyze data and extract insights from it.

Data Mining

Roughly speaking, Data Mining, a term used in Computer Science, shares exactly the same goals as Statistical Analytics. The historical difference is described in Friedman (1998), "Despite the obvious connections between data mining and statistical data analysis, most of the methodologies used in Data Mining have so far originated in fields other than Statistics.". However, the two fields have evolved so much since, that they now share many common methodologies. Nowadays, these two terms are basically interchangeable. When Data Mining is discussed, it is worth mentioning a closely related term: Knowledge Discovery in Databases (KDD). The distinction between Data Mining and Knowledge Discovery in Databases was discussed in [128]: "KDD refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process. Data mining is the application of specific algorithms for extracting patterns from data.". The additional steps of KDD mentioned in this paper are data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining. These are not part of Data Mining.

Business Intelligence / Business Analytics

Business Intelligence, or equivalently Business Analytics, is a very broad term referring to commercial organizations using data to learn about the business, market or customers and to make factually-supported decisions. It may also involve making prediction for the future or the unknown for the benefit of the business. It may be described as the KDD for Business.

Data Science

Data science is an umbrella term for everything mentioned above that makes use of data, with an emphasis of the use of sophisticated algorithms or scientific methods. The term has a strong academic background. Peter Naur, a professor of computer science at University of Copenhagen, first coined it in the 1960s. In the mid-1990s, the International Federation of Classification Societies describes it as "research in problems of

classification, data analysis, and systems for ordering knowledge". Data Science was proposed as a new academic discipline in 2001 by William Cleveland, who states that it involves "advances in computing with data" on statistics. The term Data Science is widely used in business in recent years. Before its widespread use, the business community tended to use two other umbrella terms with similar meanings: Data Analytics and Data Warehousing. They are about analyzing data to extract useful information or knowledge. Data warehousing is usually about tasks that produce one-off reports in an offline mode, while data analytics can be conducted in a real-time online environment. Unlike Data Science, however, they do not imply the use of sophisticated algorithms. The profession of a Data Scientist is also becoming more popular. An article in the Harvard Business Review [129] describes it as "the sexiest job of the 21st century". Data science usually involves techniques based on research in fields such as computer science, statistics, math, physics, finance, economics and psychology.

Bibliography

[1] T. Davenport and J. Harris, *Competing on analytics: the new science of winning*. Harvard Business Press, 2007.

[2] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *Internet Computing, IEEE*, vol. 7, pp. 76 – 80, jan/feb 2003.

[3] R. Bell, J. Bennett, Y. Koren, and C. Volinsky, "The million dollar programming prize," *Spectrum, IEEE*, vol. 46, pp. 28 –33, may 2009.

[4] R. Kohavi, N. J. Rothleder, and E. Simoudis, "Emerging trends in business analytics," *Commun. ACM*, vol. 45, pp. 45–48, Aug. 2002.

[5] Z. Huang, D. Zeng, and H. Chen, "Analyzing consumer-product graphs: Empirical findings and applications in recommender systems," *Management Science*, vol. 53, no. 7, pp. 1146–1164, 2007.

[6] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[7] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[8] K. Wei, J. Huang, and S. Fu, "A survey of e-commerce recommender systems," in *Service Systems and Service Management, 2007 International Conference on*, pp. 1 –5, june 2007.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, pp. 5–53, January 2004.

[10] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: the montage example," in *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, p. 50, IEEE Press, 2008.

[11] C. Bizer, P. Boncz, M. L. Brodie, and O. Erling, "The meaningful use of big data: four perspectives–four challenges," *ACM SIGMOD Record*, vol. 40, no. 4, pp. 56–60, 2012.

[12] J. Altmann, "Observational study of behavior: sampling methods," *Behaviour*, pp. 227–267, 1974.

[13] G. Adomavicius and A. Tuzhilin, "Using data mining methods to build customer profiles," *Computer*, vol. 34, pp. 74–82, Feb 2001.

[14] C. Cumby, A. Fano, R. Ghani, and M. Krema, "Predicting customer shopping lists from point-of-sale purchase data," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, (New York, NY, USA), pp. 402–409, ACM, 2004.

[15] M. B. Dias, D. Locher, M. Li, W. El-Deredy, and P. J. Lisboa, "The value of personalised recommender systems to e-business: A case study," in *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, (New York, NY, USA), pp. 291–294, ACM, 2008.

[16] T. Belluf, L. Xavier, and R. Giglio, "Case study on the business value impact of personalized recommendations on a large online retailer," in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, (New York, NY, USA), pp. 277–280, ACM, 2012.

[17] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, (New York, NY, USA), pp. 39–46, ACM, 2010.

[18] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Croc: A new evaluation criterion for recommender systems," 2005.

[19] M. R. McLaughlin and J. L. Herlocker, "A collaborative filtering algorithm and evaluation metric that accurately model the user experience," in *SIGIR '04*, 2004.

[20] A. Azaria, A. Hassidim, S. Kraus, A. Eshkol, O. Weintraub, and I. Netanely, "Movie recommender system for profit maximization," in *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, (New York, NY, USA), pp. 121–128, ACM, 2013.

[21] V. Kalnikaite, Y. Rogers, J. Bird, N. Villar, K. Bachour, S. Payne, P. M. Todd, J. Schöning, A. Krüger, and S. Kreitmayer, "How to nudge in situ: Designing lambent devices to deliver salient information in supermarkets," in *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, (New York, NY, USA), pp. 11–20, ACM, 2011.

[22] M. Moss, "Nudged to the produce aisle by a look in the mirror."

[23] S. Thurston, "Donate a dollar at the register? checkout charity is big business for nonprofits."

[24] A. D. Kramer, J. E. Guillory, and J. T. Hancock, "Experimental evidence of massive-scale emotional contagion through social networks," *Proceedings of the National Academy of Sciences*, p. 201320040, 2014.

[25] M. A. Hall and L. A. Smith, "Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper," in *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, pp. 235–239, AAAI Press, 1999.

[26] E. Neufeld and S. Kristtorn, "Does non-correlation imply non-causation?," *Int. J. Approx. Reasoning*, vol. 46, pp. 257–273, Oct. 2007.

[27] J. Aldrich, "Correlations genuine and spurious in pearson and yule," *Statistical Science*, vol. 10, no. 4, pp. pp. 364–376, 1995.

[28] P. W. Holland, "Statistics and Causal Inference," *Journal of the American Statistical Association*, vol. 81, pp. 945–960, Dec. 1986.

[29] J. Pearl, *Causality: Models, Reasoning, and Inference*. New York, NY, USA: Cambridge University Press, 2000.

[30] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *SIGMOD Rec.*, vol. 22, pp. 207–216, June 1993.

[31] K.-I. Ahn, "Effective product assignment based on association rule mining in retail," *Expert Syst. Appl.*, vol. 39, pp. 12551–12556, Nov. 2012.

[32] G. Kronberger and M. Affenzeller, "Market basket analysis of retail data: Supervised learning approach," in *Proceedings of the 13th International Conference on Computer Aided Systems Theory - Volume Part I*, EUROCAST'11, (Berlin, Heidelberg), pp. 464–471, Springer-Verlag, 2012.

[33] Y.-L. Chen, K. Tang, R.-J. Shen, and Y.-H. Hu, "Market basket analysis in a multiple store environment," *Decis. Support Syst.*, vol. 40, pp. 339–354, Aug. 2005.

[34] R. A. Lewis, J. M. Rao, and D. H. Reiley, "Here, there, and everywhere: Correlated online behaviors can lead to overestimates of the effects of advertising," in *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, (New York, NY, USA), pp. 157–166, ACM, 2011.

[35] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*, pp. 263 –272, dec. 2008.

[36] D. Oard and J. Kim, "Implicit feedback for recommender systems," in *Proceedings of the AAAI Workshop on Recommender Systems*, pp. 81–83, 1998.

[37] M. Claypool, P. Le, M. Wased, and D. Brown, "Implicit interest indicators," in *IUI '01*, 2001.

[38] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," *Recommender Systems Handbook*, pp. 217–253, 2011.

[39] S. Anand and B. Mobasher, "Contextual recommendation," *From Web to Social Web: Discovering and Deploying User and Content Profiles*, pp. 142–160, 2007.

[40] B. Marlin, "Collaborative filtering: a machine learning perspective," Master's thesis, Department of Computer Science, University of Toronto, 2004.

[41] M. Pazzani and D. Billsus, "Content-based recommendation systems," *The adaptive web*, pp. 325–341, 2007.

[42] X. Su and T. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, vol. 2009, p. 4, 2009.

[43] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 426–434, ACM, 2008.

[44] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, "Application of dimensionality reduction in recommender system – a case study," in *Proc. of ACM WebKDD Workshop*, (New York, NY), ACM Press, 2000.

[45] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," in *AAAI/IAAI*, pp. 187–192, 2002.

[46] M. Bazire and P. Brezillon, "Understanding context before using it," *Modeling and using context*, pp. 113–192, 2005.

[47] P. Dourish, "What we talk about when we talk about context," *Personal Ubiquitous Comput.*, vol. 8, pp. 19–30, Feb. 2004.

[48] X. Jin, Y. Zhou, and B. Mobasher, "Task-oriented web user modeling for recommendation," *User Modeling 2005*, pp. 149–149, 2005.

[49] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Trans. Inf. Syst.*, vol. 23, pp. 103–145, Jan. 2005.

[50] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[51] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische Mathematik*, vol. 14, no. 5, pp. 403–420, 1970.

[52] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the netflix prize," pp. 337–348, Springer, 2008.

[53] C. Palmisano, A. Tuzhilin, and M. Gorgoglione, "Using context to improve predictive modeling of customers in personalization applications," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 20, pp. 1535 –1549, nov. 2008.

[54] J. Cornfield *et al.*, "A method of estimating comparative rates from clinical data; applications to cancer of the lung, breast, and cervix.," *Journal of the National Cancer Institute*, vol. 11, no. 6, p. 1269, 1951.

[55] F. Mosteller, "Association and estimation in contingency tables," *Journal of the American Statistical Association*, vol. 63, no. 321, pp. 1–28, 1968.

[56] A. Edwards, "The measure of association in a 2×2 table," *Journal of the Royal Statistical Society. Series A (General)*, pp. 109–114, 1963.

[57] N. Mantel and W. Haenszel, "Statistical aspects of the analysis of data from retrospective studies of disease," *The Challenge of Epidemiology: Issues and Selected Readings*, vol. 1, no. 1, pp. 533–553, 2004.

[58] D. Kleinbaum, L. Kupper, and H. Morgenstern, *Epidemiologic research: principles and quantitative methods*. Wiley, 1982.

[59] A. BellogÌn, I. Cantador, P. Castells, and A. Ortigosa, "Discovering relevant preferences in a personalised recommender system using machine learning techniques," in *Proceedings of the ECML-PKDD 2008 Workshop on Preference Learning*, 2008.

[60] B. Vargas-Govea, G. González-Serna, and R. Ponce-Medellín, "Effects of relevant contextual features in the performance of a restaurant recommender system," *In RecSys11: Workshop on context-aware recommender systems (CARS-2011)*, 2011.

[61] S. Lombardi, S. Anand, and M. Gorgoglione, "Context and customer behavior in recommendation," *In RecSys09: Workshop on context-aware recommender systems (CARS-2009)*, 2009.

[62] U. Panniello and M. Gorgoglione, "Does the recommendation task affect a cars performance?," *In RecSys10: Workshop on context-aware recommender systems (CARS-2010)*, 2010.

[63] J. Morris and M. Gardner, "Statistics in medicine: Calculating confidence intervals for relative risks (odds ratios) and standardised ratios and rates," *British medical journal (Clinical research ed.)*, vol. 296, no. 6632, p. 1313, 1988.

[64] W. Hauck, "The large sample variance of the mantel-haenszel estimator of a common odds ratio," *Biometrics*, pp. 817–819, 1979.

[65] J. Robins, N. Breslow, and S. Greenland, "Estimators of the mantel-haenszel variance consistent in both sparse data and large-strata limiting models," *Biometrics*, pp. 311–323, 1986.

[66] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. of SIGIR*, 2002.

[67] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 1998.

[68] A. Schein, A. Popescul, L. Ungar, and D. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 253–260, ACM, 2002.

[69] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Generative models for cold-start recommendations," in *Proceedings of the 2001 SIGIR Workshop on Recommender Systems*, 2001.

[70] L. He, N. N. Liu, and Q. Yang, "Active dual collaborative filtering with both item and attribute feedback," in *AAAI*, 2011.

[71] E. Spyromitros-Xioufis, E. Stachtiari, G. Tsoumakas, and I. Vlahavas, "A hybrid approach for cold-start recommendations of videolectures," in *Proc. of ECML-PKDD 2011 Discovery Challenge Workshop*, pp. 29–39, 2011.

[72] N. Rubens, R. Tomioka, and M. Sugiyama, "Output divergence criterion for active learning in collaborative settings," *IPSJ Online Transactions*, vol. 2, pp. 240–249, 2009.

[73] R. Jin and L. Si, "A bayesian approach toward active learning for collaborative filtering," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, (Arlington, Virginia, United States), pp. 278–285, AUAI Press, 2004.

[74] C. Piao, J. Zhao, and L. Zheng, "Research on entropy-based collaborative filtering algorithm and personalized recommendation in e-commerce," *Service Oriented Computing and Applications*, vol. 3, no. 2, pp. 147–157, 2009.

[75] K. Zhou, S.-H. Yang, and H. Zha, "Functional matrix factorizations for cold-start recommendation," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, (New York, NY, USA), pp. 315–324, ACM, 2011.

[76] N. Golbandi, Y. Koren, and R. Lempel, "Adaptive bootstrapping of recommender systems using decision trees," in *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, (New York, NY, USA), pp. 595–604, ACM, 2011.

[77] A. Brun, S. Castagnos, and A. Boyer, "Social recommendations: mentor and leader detection to alleviate the cold-start problem in collaborative filtering," in *Social Network Mining, Analysis and Research Trends: Techniques and Applications* (T.-P. H. I-Hsien Ting and L. S. Wang, eds.), IGI Global, Nov. 2011.

[78] A. Jameson and B. Smyth, "The adaptive web," ch. Recommendation to groups, pp. 596–627, Berlin, Heidelberg: Springer-Verlag, 2007.

[79] N. J. Belkin and W. B. Croft, "Information filtering and information retrieval: two sides of the same coin?," *Commun. ACM*, vol. 35, no. 12, pp. 29–38, 1992.

[80] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[81] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[82] K. S. Anand and R. Aron, "Group buying on the web: A comparison of price-discovery mechanisms," *Manage. Sci.*, vol. 49, pp. 1546–1562, November 2003.

[83] C. Boutilier, R. Zemel, and B. Marlin, "Active collaborative filtering," in *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, pp. 98–106, Citeseer, 2003.

[84] A. Atkinson, A. Donev, and R. Tobias, *Optimum experimental designs, with SAS*, vol. 34. Oxford University Press, USA, 2007.

[85] D. Chakrabarti, R. Kumar, F. Radlinski, and E. Upfal, "Mortal multi-armed bandits," in *Advances in Neural Information Processing Systems*, pp. 273–280, 2009.

[86] J. C. Gittins, "Bandit processes and dynamic allocation indices," *Journal of the Royal Statistical Society Series B Methodological*, vol. 41, no. 2, p. 148177, 1979.

[87] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons Inc, 1998.

[88] J. Lu and Q. Zhan, *Optimum Experimental Designs, with SAS*. Blackwell Publishing, 2008.

[89] R. Penrose, "A generalized inverse for matrices," in *Proc. Cambridge Philos. Soc*, vol. 51, p. C655, Cambridge Univ Press, 1955.

[90] DataSet. MovieLens: http://www.grouplens.org/.

[91] A. Mason and I. Dunning, "Opensolver."

[92] A. S. Harpale and Y. Yang, "Personalized active learning for collaborative filtering," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, (New York, NY, USA), pp. 91–98, ACM, 2008.

[93] P. M. Rasmussen, L. K. Hansen, K. H. Madsen, N. W. Churchill, and S. C. Strother, "Model sparsity and brain pattern interpretation of classification models in neuroimaging," *Pattern Recognition*, vol. 45, no. 6, pp. 2085 – 2100, 2012. Brain Decoding.

[94] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[95] S. Kirkpatrick, D. G. Jr., and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[96] M. J. Powell, *A direct search optimization method that models the objective and constraint functions by linear interpolation*. Springer, 1994.

[97] T. Weise, "Global optimization algorithms–theory and application," *Self-Published,*, 2009.

[98] F. Hutter, *Automated configuration of algorithms for solving hard computational problems*. PhD thesis, Citeseer, 2009.

[99] C. Weihs, K. Luebke, and I. Czogiel, "Response surface methodology for optimizing hyper parameters," tech. rep., Technical Report/Universität Dortmund, SFB 475 Komplexitätsreduktion in Multivariaten Datenstrukturen, 2006.

[100] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *arXiv preprint arXiv:1206.2944*, 2012.

[101] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization*, pp. 507–523, Springer, 2011.

[102] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: Automated selection and hyper-parameter optimization of classification algorithms," *CoRR*, vol. abs/1208.3719, 2012.

[103] A. Nareyek, "Choosing search heuristics by non-stationary reinforcement learning," *Applied Optimization*, vol. 86, pp. 523–544, 2003.

[104] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[105] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[106] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Mar. 2012.

[107] M.-A. Sato, "Online model selection based on the variational bayes," *Neural Computation*, vol. 13, no. 7, pp. 1649–1681, 2001.

[108] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*, (New York, NY, USA), pp. 271–280, ACM Press, 2007.

[109] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[110] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis, "Large-scale matrix factorization with distributed stochastic gradient descent," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, (New York, NY, USA), pp. 69–77, ACM, 2011.

[111] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pp. 1–10, IEEE, 2010.

[112] S. Vargas and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, (New York, NY, USA), pp. 109–116, ACM, 2011.

[113] M. Zhang, "Enhancing diversity in top-n recommendation," in *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, (New York, NY, USA), pp. 397–400, ACM, 2009.

[114] A. Das, C. Mathieu, and D. Ricketts, "Maximizing profit using recommender systems," *arXiv preprint arXiv:0908.3633*, 2009.

[115] K. Ali and W. Van Stam, "Tivo: making show recommendations using a distributed collaborative filtering architecture," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 394–401, ACM, 2004.

[116] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2008.

[117] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *J. Mach. Learn. Res.*, vol. 10, pp. 623–656, June 2009.

[118] R. Pan, Y. Zhou, B. Cao, N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*, pp. 502–511, 2008.

[119] H. Dongyuan and C. Xiaoyun, "Tuning svm hyperparameters in the primal," in *Computational Intelligence and Natural Computing Proceedings (CINC), 2010 Second International Conference on*, vol. 1, pp. 201–204, 2010.

[120] J. Acevedo, S. Maldonado, P. Siegmann, S. Lafuente, and P. Gil, "Tuning l1-svm hyperparameters with modified radius margin bounds and simulated annealing," in *Proceedings of the 9th international work conference on Artificial neural networks*, IWANN'07, (Berlin, Heidelberg), pp. 284–291, Springer-Verlag, 2007.

[121] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, "Graphlab: A new framework for parallel machine learning," *arXiv preprint arXiv:1006.4990*, 2010.

[122] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, pp. 10–10, 2010.

[123] R. T. Fielding, *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, 2000.

[124] L. D. Paulson, "Building rich web applications with ajax," *Computer*, vol. 38, no. 10, pp. 14–17, 2005.

[125] N. McDonald and S. Goggins, "Performance and participation in open source software on github," in *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, (New York, NY, USA), pp. 139–144, ACM, 2013.

[126] R. Magoulas and B. Lorica, "Introduction to big data, release 2.0, issue 11," 2009.

[127] M. A. Beyer and D. Laney, "The importance of'big data': a definition," *Stamford, CT: Gartner*, 2012.

[128] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, p. 37, 1996.

[129] T. H. Davenport and D. Patil, "Sexiest job of the 21st century," October 2012.