Price Incentives for Online Retailers using Social Media Data

by

Ludovica Rizzo

B.Eng., Ecole Polytechnique, Paris (2013)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2015

Author .....
Sloan School of Management
May 15, 2015

Certified by .....
Georgia Perakis
William F. Pounds Professor of Management
Thesis Supervisor

Accepted by .....
Dimitris Bertsimas
Boeing Professor of Operations Research
Co-Director, Operations Research Center

Price Incentives for Online Retailers using Social Media Data

by

Ludovica Rizzo

Submitted to the Sloan School of Management
on May 15, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

In the era of Big Data, online retailers have access to a large amount of data about their customers. This data can include demographic information, shopping carts, transactions and browsing history. In the last decade, online retailers have been leveraging this data to build a personalized shopping experience for their customers with targeted promotions, discounts and personalized item recommendations. More recently, some online retailers started having access to social media data: more accurate demographic and interests information, friends, social interactions, posts and comments on social networks, etc. Social media data allows to understand, not only what customers buy, but also what they like, what they recommend to their friends, and more importantly what is the impact of these recommendations. This work is done in collaboration with an online marketplace in Canada with an embedded social network on its website. We study the impact of incorporating social media data on demand forecasting and we design an optimized and transparent social loyalty program to reward socially active customers and maximize the retailer's revenue.

The first chapter of this thesis builds a demand estimation framework in a setting of heterogeneous customers. We want to cluster the customers into categories according to their social characteristics and *jointly* estimate their future consumption using a distinct logistic demand function for each category. We show that the problem of joint clustering and logistic regression can be formulated as a mixed-integer concave optimization problem that can be solved efficiently even for a large number of customers. We apply our algorithm using the actual online marketplace data and study the impact of clustering and incorporating social features on the performance of the demand forecasting model.

In the second chapter of this thesis, we focus on price sensitivity estimation in the context of missing data. We want to incorporate a price component in the demand model built in the previous chapter using recorded transactions. We face the problem of *missing data*: for the customers who make a purchase we have access to the price they paid, but for customers who visited the website and decided not to make a purchase, we do not observe the price they were offered. The EM (Expectation Maximization) algorithm is a classical approach for estimation with missing data. We propose a non-parametric alternative to the EM algorithm, called NPM (Non-Parametric Maximization). We then show analytically the consistency of our algorithm in two particular settings. With extensive simulations, we show that NPM is

a robust and flexible algorithm that converges significantly faster than EM.

In the last chapter, we introduce and study a model to incorporate social influence among customers into the demand functions estimated in the previous chapters. We then use this demand model to formulate the retailer' revenue maximization problem. We provide a solution approach using dynamic programming that can deal with general demand functions. We then focus on two special structures of social influence: the nested and VIP models and compare their performance in terms of optimal prices and profit. Finally, we develop qualitative insights on the behavior of optimal price strategies under linear demand and illustrate computationally that these insights still hold for several popular non-linear demand functions.

Thesis Supervisor: Georgia Perakis
Title: William F. Pounds Professor of Management

Acknowledgments

The two years I spent at MIT have been extremely enriching both intellectually and personally. I would like to thank all the people that have been in my life during this journey.

First of all, I would like to thank The Accenture and MIT Alliance on Business Analytics for funding this project and SHOP.CA for providing the data. In particular, I would like to thank Marjan Baghaie, Andrew Fano, Thania Villatoro, Leslie Sheppard, Trevor Newell and Gary Black for helpful discussions.

I would like to thank my advisor, Georgia Perakis. I am extremely grateful for the opportunity you offered me to work on this applied research project. I loved learning how academic research interfaces with industry and how data can be extremely powerful to support modern decision making. Your guidance, support and enthusiasm have made my stay at MIT an incredibly enriching experience. I would also like to thank Maxime Cohen for being a great research teammate. Without your help, this project wouldn't have been the same.

I am grateful to the ORC staff Laura Rose and Andrew Carvalho. Your incredible flexibility and patience makes the ORC a perfectly organized place to work in.

My time at the Operations Research Center wouldn't have been nearly as memorable without the "ORCers". I would like to thank all my ORC friends for making me feel at home in Boston. A special thank to JLVD for the great time we had organizing Informs social events. Thanks to Andrew, Charlie, Anna, Mapi, Stef, Zach and all the first, second and third years for the amazing weekends we spent in SidPac and exploring Boston. Finally, thanks to Cécile for being the best roommate I could have hoped for.

Last but not least, thanks to Elhadi for always being there to support me. Finally, I would like to express my gratitude to my family and especially to my parents who have always supported me in all these years spent abroad.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem and Motivation

In the last years, the amount of data available to online retailers has increased exponentially. The Big Data revolution is drastically changing e-commerce. Online retailers have access to a tremendous amount of information about their customers: browsing history, shopping carts and demographic information. According to [1], online retailers have to use Big Data for "personalization, dynamic pricing and predictive analytics". Customers shop from the same retailer in different ways. The available data should be used to offer personalized items or promotions and to reward loyal customers. Big Data can also be used to build accurate predictive models. By learning customers' tastes and preferences, the retailer can forecast the future demand for specific items. This is of great help for managing inventory for example. Finally, Big Data is a key tool for effective pricing strategies. According to a study by McKinsey, a retailer using Big Data to its current potential could increase their operating margin by up to 60% ([2]) and companies that used big data and analytics outperformed their peers by 5% in productivity and 6% in profitability. Recently, social media data has become accessible to some online retailers. Social media generate an incredibly large amount of data that can be useful for a better understanding of customers' tastes, social interactions and behavior. Incorporating social media in a Big Data analytics approach is the new challenge for pricing for online retailers.

This research is part of the Accenture and MIT Alliance on Business Analytics in col-

laboration with SHOP.CA: an online retailer who uses social media data to build a social loyalty program where customers receive different prices according to their social activity. Using their data, we build an optimized loyalty program.

1.2 SHOP.CA

SHOP.CA is a Canadian online marketplace with a social network in its website. After creating an account, customers can connect to other members with "Friend requests". They can use the social network to send private messages and item recommendations. They can share with their friends comments and reviews about their purchases. They can connect their account with Facebook, LinkedIn, Twitter and with personal blogs. This embedded social network is built to improve the customer shopping experience. The basic idea is the following: when a customer purchases an item online, most of the times he uses its reviews and ratings to evaluate its quality. But the review written by a total stranger may not have the same effect as one written by a "good" and trusted friend. With an embedded social network, before making a purchase, customers will be able to see which of their friends bought the item and what they thought about it. This creates a personalized shopping experience.

From the retailer perspective, building a social network platform increases the available information about customers. This additional information about customers' behavior is extremely valuable to identify different segments in the population and to better predict their behavior.

Badges and Rewards SHOP.CA's business model is based on social interactions between its customers. To incentivize customers to be more social (add friends, send messages and reviews, share the items they purchased), they build a "social loyalty program" where customers are rewarded for their social activity with cash back (discount to use for future purchases).

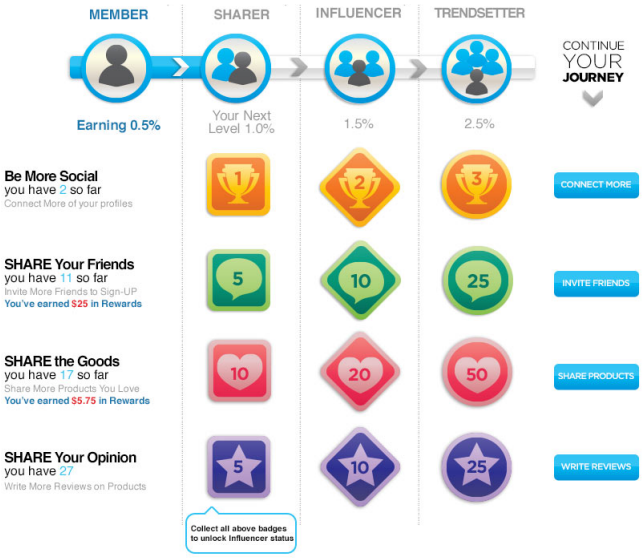


Figure 1-1: Business Model

Figure 1-1 summarizes the current social loyalty program. Customers are clustered into 4 different categories (called "badges") according to their social activity. Social activity is measured in terms of number of friends, connections to external social platforms (as Facebook and Twitter), reviews written and item recommendations (called "shares"). A customer who just created an account is a "Member". If he connects his account to a social network, adds 5 friends, shares 10 items AND writes 5 reviews then he becomes a "Sharer". Adding additional social networks, friend shares and reviews, he can successively become an "Influencer" and a "Trendsetter". Each of these badges is associated with a nominal level of rewards on every purchase. Using SHOP.CA vocabulary we use the term rewards to mean discount. For example, 3% rewards means that the customers receives a 3% discount on his purchase. A Member receives 0.5% reward on each purchase, a Sharer receives 1%, an Influencer 1.5% and a Trendsetter 2.5%. This nominal level of discount is applied on every transaction. The customer receives it on a personal cash balance on the website. This cash-back can be used without restrictions on any future purchase. Figure 1-1 is a screen shot from a user's account and reports the different level of badges and associated rewards. This specific user is connected with 2 social networks, has 11 friends, 17 shares and 27 reviews, he is thus a "Sharer". In addition to this nominal level of discount that depends on the badge category, social interactions are rewarded with a fixed amount of cash-back. For example, if a customer

refers a friend to join SHOP.CA and this friend subsequently creates an account and makes a first purchase, the referrer receives $ 10 cash-back.

SHOP.CA business model is based on transparency. Rather than sending complicated targeted discounts campaigns where the customer is not aware of why he is offered a specific discount and what level of discount is offered to the other customers, SHOP.CA aims on a transparent rewards program. Every customer knows exactly what level of reward he is assigned to and what actions are required to get to the next badge category. Furthermore, the criteria defining the badges are based exclusively on social activities. The badge definition does not depend on the amount spent on the website or the number of recent purchases. SHOP.CA wants to incentivize his customers to be more social: to write more reviews, more recommendations, have more friends and has built his loyalty program with this objective in mind.

1.3 Contributions

SHOP.CA's current business model is based on four badges levels defined according to reviews, shares, friends and connections to external social network. For example, to get to the level "Sharer", 1 social network, 5 friends, 10 shares and 5 reviews are required. Furthermore, the badges categories are associated with 0.5%, 1%, 1.5% and 2.5% rewards. The goal of this work is to optimize the current social loyalty program and to answer the questions:

1. How should the badges categories be defined (which social features should be used and how should the badges levels be defined) in order to have a good customer segmentation and demand estimation?

2. What levels of rewards should be given to each badge in order to maximize the total revenue?

In this work, we answer these questions in three steps. In Chapter 2, we focus on the first of the two questions. We present an optimization framework that defines the badges categories and estimates distinct demand models parameters for each badge. For each cluster, we use the transaction history and social interactions of the customers to predict their future

consumption with a logistic choice model. We show that this problem can be formulated as a mixed-integer convex optimization that can be solved efficiently by commercial solvers. We apply our results on SHOP.CA data and show that clustering customers and incorporating social features in the demand model significantly improves the prediction accuracy.

The model built in Chapter 2 uses customers' transaction history and social interactions to predict their future consumption. A key aspect in a revenue management perspective that is not incorporated in this model is the customers' price sensitivity. Working only with transaction data, we face a problem of *missing data*. We observe which customers make purchases and which do not, but the prices are recorded only when a transaction occurs. *We do not observe the prices offered to customers that decide not to purchase.* In Chapter 3, we propose two possible approaches to incorporate price sensitivity to our predictive model. We first present the EM algorithm: a classical parametric approach to parameter estimation in a setting of missing data. We then propose a novel estimation algorithm that can be applied in a more general non-parametric setting. We compare the two approaches on simulated data.

In Chapter 4, we address the second question formulated above. We start by proposing a model to capture the social influence between badges. Given a badge structure and associated demand model, we then solve the problem of maximizing the total revenue with a Dynamic Programming approach. We then focus on two specific structures of social influence: the nested and VIP models. In a symmetric setting, we develop insights on the behavior of optimal pricing policies.

1.4 Data

For our models, we used social and transaction data from SHOP.CA collected from January 2013 to February 2014. We considered four types of data: transaction data, item data, customers' demographic information, social activity data. In this section, we present the raw data we received, in later chapters we will explain how this data has been transformed to answer specific questions. Descriptive statistics and histograms can be found in Appendix A.1.

1.4.1 Transaction Data

For every transaction on the website we had the following information:

USER_ID	ITEM_ID	Date	Nominal Price	Rewards
unique identifier of the buyer	unique identifier of the item purchased		price listed on the website (before discount)	discount received ($)

Table 1.1: Transaction data

1.4.2 Item Data

For every item in the catalog we have

ITEM_ID	category	Reviews statistics
	Electronics, Books, Cooking, Outdoor ...	average rating, number of reviews, number of recommendations

Table 1.2: Item data

1.4.3 Customer Demographic Information

For every customer registered on the website we have access to the following demographic information. Note that some of the features may be missing if the customer didn't reveal the information

USER_ID	Gender	Age	Zipcode

Table 1.3: Demographic data

1.4.4 Social Activity Data

The social activity data is the most interesting component of our data. Every user's action or activity is recorded in the following format:

USER_ID	Date and time	Action and Action Attributes

Table 1.4: Social Activity data

The social actions that have the most important role in our work are:

- Log In : customer logs in to his account
- Review written
- Number of friends
- Recommendation/"share": personal recommendation sent about a specific item
- Referral: invitation sent to a non-member to join SHOP.CA website (a referral is considered as "successful" if the non-member creates an account after receiving the invitation)

In addition, we have access to SHOP.CA internal social network. We know its structure (who is friend with who), who sent the friend request and when and we observe the volume of personal messages and item recommendations between every pair of friends. Furthermore, we know which customers linked their profile with external social networks (Facebook, LinkedIn, Twitter ...) and have access to SHOP.CA related posts on these external platforms.

A detailed presentation of all the recorded actions and their attributes can be found in Table A.1 in the appendix.

Chapter 2

Joint Clustering and Logistic Regression

2.1 Introduction and Literature Review

In this Chapter, we consider an online retailer who faces a set of heterogeneous customers. The retailer aims to see if customers with "similar" characteristics naturally cluster into segments. This can allow the retailer to build a distinct demand model for each of these segments.

Customer segmentation: With the rise of airline revenue management in the last few decades, customer segmentation has become a central topic of interest. In the context of airlines, it is hard to model well every potential customer using the same demand function. On one side, leisure travelers book a long time in advance, have flexible dates and are very price sensitive. On the other, business travelers often book at the last minute, have a tight schedule and are not price sensitive. For an efficient seat allocation among these two types of travelers, the airline has to estimate different demand functions for the two segments. The same problem arises in e-commerce when retailers have multiple sources of information about their customers (not only transaction data but also browsing history and social interactions). Adding customer segmentation can be extremely useful in order to build an accurate demand model.

Logistic demand function: Traditional retailers collect information in an aggregate way. They collect the number of purchases in a given store, in a given day, they may have some information on the store inventory and on the prices but they do not have information on the individual customer preferences. In this setting, only the cumulative demand (at a store, or at a regional level for a given day for example) can be forecasted as a function of the price (and of some characteristics of the store and the day).

Modern online retailers often have access to a large amount of data about their customers (transaction and browsing history, demographic information, and more recently social network data). It is then possible to forecast the demand at a more personalized level. In this setting, choice models are a powerful and flexible tool that allows to forecast which individuals will make a purchase in the near future and what they will buy. In this work, we will use the most common choice model for the demand estimation: the logit model. The multinomial logit model can be used to predict purchase behavior when customers are offered a set of distinct items. (Binary) logistic regression deals with the simple (and more aggregated) setting where customers are offered only two alternatives: to buy or not to buy.

Joint clustering and demand estimation-Literature Review: The problem of joint clustering and regression has been widely studied in the computational statistics literature. Classification and Regression Trees (CART) greedily build a tree by recursively splitting the data sets in rectangular areas ([3]). Regression trees then build a distinct regression model for every leaf. Multivariate Adaptive Regression Splines (MARS) also split the points into region but use splines for regression to guarantee continuity of the prediction between one region and another ([4]). Another possible approach used in the revenue management literature is a mixture of models. Mixtures do not allow to classify customers into categories but rather represent the global demand as a weighted average of different demand functions. This increases the granularity of the problem and gives a more accurate aggregated demand prediction. These methods rely on continuous optimization techniques and can often be approximately solved with greedy heuristics. They have been studied by the data mining community in the last 20 years and have been used for a large number of applications thanks to commercial software packages that have been developed.

A different approach for classification and linear regression is proposed by Bertsimas and Shioda in [5] using integer (rather than continuous) optimization. They derive a linear mixed integer program able to separate the points into polyhedral regions and build a distinct linear regression for each of these regions. With the advances in integer optimization in the last decade, integer formulations of clustering problems have become competitive with classical greedy approaches and can be solved even for large data sets.

Finally, in the context of pricing with choice models, Keller, Levi and Perakis in [6] show that, taking advantage of monotonicity properties of choice models, it is possible to reformulate an intractable constrained non-convex pricing problem into a convex program that can be solved fast.

Contributions: To the best of our knowledge, the problem of joint clustering and logistic regression has not been studied before. In this work, we show that it can be reformulated as a strictly concave mixed-integer program and illustrate computationally that it can be solved efficiently for large data sets. Least absolute value linear regression is inherently a linear optimization problem and [5] is able to reformulate the problem of joint clustering and $\mathbb{L}_1$ regression as a linear mixed-integer problem. In this paper, we exploit the monotonicity properties of choice models highlighted in [6] to show that joint multinomial logistic regression and clustering can be rewritten as a mixed-integer concave optimization problem. We then apply our method to predict customers' future consumption using SHOP.CA data. We show that our model is able to capture the heterogeneity in the customers' population and that, by adding a clustering step on top of the logistic demand estimation, the prediction accuracy increases significantly. Furthermore, we show that social media data is extremely valuable for online retailers. Considering social media features as the number of friends, the number of messages sent, connections to social networks etc. allows a deeper understanding of customers' behavior and preferences which leads to a more accurate demand estimation. Finally, performing sensitivity analysis on the output of our model can help retailers to understand which features (social or from transaction data) drive customers' purchase behavior and what is the marginal effect of each one of the features (the impact of an additional friend or review on the customer's likelihood to buy for example). This can be useful to decide how

to allocate the Marketing efforts and to allocate rewards.

Outline: In Sections 2.2 and 2.3, we motivate and define our model and introduce the notations we use in the rest of the Chapter. In Section 2.4, we present the problem of joint clustering and (binary) logistic regression and show how it can be reformulated into a concave mixed-integer program. In Section 2.5, we extend the previous result to the case where the demand follows a multinomial logit model. Finally, in Sections 2.6 and 2.7 we analyze the performance of our approach on SHOP.CA data.

2.2 Problem definition and Motivation

In this Section, we present the main motivations for our model.

2.2.1 A transparent and hierarchical business model

Recall the screen shot presented in Figure 1-1. In SHOP.CA's website, customers are clustered into 4 different categories (called "badges") according to their social activity on the website. Each of these badges is associated with a nominal level of rewards on every purchase. SHOP.CA's business model is based on transparency. Every customer knows exactly what level of reward he is assigned to and what actions are required to get to the next badge category. Finally, by definition of a loyalty program, there has to be a hierarchical structure. If customer A is more active (more friends, more reviews, ...) than customer B, he has to receive at least the same amount of reward as customer B.

SHOP.CA's business model is based on transparency and hierarchy. We need to cluster the customers into categories in a transparent and hierarchical way that is easily understandable by the customer.

2.2.2 Customer heterogeneity

We want to build a model that predicts customers' future purchase behavior and is able to take into account customer heterogeneity. We have data on different types of customers' activity: purchases, reviews, recommendations, friends, activity on social networks etc. In

a traditional revenue management setting, marketers have only access to purchase history. They can define as "good customers" the customers who made several purchases (on a regular basis or in the recent past). Once we consider social activity data, the definition of "good customers" becomes more complex. A "good customer" can be an individual who purchases from the website on a regular basis but is not active socially, or also a customer who shares many items through social medias with "friends" and regularly sends referrals emails. In this setting where purchase history and social activity are combined, it is hard to describe the entire population in a single model.

In our data set, customers are extremely heterogeneous in terms of price sensitivity (quantity of discounted items purchased), social activity and purchase behavior. This is a characteristic of online retailing where different population segments purchase from the same channel.

We want to create a framework that is able to identify the different segments in the population and build a demand model for every segment of customers.

2.2.3 Demand estimation: Logistic choice model

There are several possible approaches for demand modeling. Linear models can be used to predict the amount ($) spent and choice models can be used to predict which specific item is purchased. Our modeling choice is driven from the data we have available. In e-commerce, data is often sparse. Online retailers often have hundred of thousands of registered customers where only a small fraction makes a purchase in a given time period, and even less make multiple purchases. In order to have a robust model, we chose to *aggregate* the available information and used a logistic choice model to predict whether a customer makes a purchase in a Future period (for all categories of items together). A choice model allows to estimate a probability to buy for each customers as a function of his observable features (past purchases, social interactions, number of friends etc.).

To conclude, our goal is two-fold. In order to capture customers' heterogeneity, we want to, jointly,

- cluster customers into categories with hierarchical and transparent rules

- fit a logistic demand model for every category (cluster)

2.3 Model

In this Section, we introduce the notation and assumptions we will use throughout this work.

2.3.1 Notations

Let us consider a set of customers $\{i \in [1, \ldots, N]\}$ characterized by their social and transaction history. We consider a point in time t_0 and we denote by "Past Period" (resp. "Future Period") all the actions taken before (resp. after) t_0. For every customer i, we build a vector of Past features $\mathbf{X_i}$ (a subset of the features presented in table A.2) and a binary variable y_i that indicates whether customer i makes a purchase in the Future Period. We want to cluster the customers into K categories and jointly estimate demand with a logistic model for each cluster.

We will use the following notations:

- N number of customers
- $i \in [1, \ldots, N]$ index of a customer
- $\mathbf{X_i} \in \mathbb{R}^m$ vector of features of customer i (the first element of $\mathbf{X_i}$ is 1 for each customer to incorporate an intercept term)
- K number of clusters
- $\mathcal{C}_k$ cluster k
- $y_i = \begin{cases} 1 & \text{if customer } i \text{ makes at least one purchase in the Future Period} \\ 0 & \text{otherwise} \end{cases}$
- $a_{i,k} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to cluster } \mathcal{C}_k \\ 0 & \text{otherwise} \end{cases}$

Assumption 1. *We assume that the purchase behavior follows a different logistic model for each cluster:*

$$\mathbb{P}(y_i = 1|\mathbf{X_i}, i \in \mathcal{C}_k) = \frac{e^{\beta_k.\mathbf{X_i}}}{1 + e^{\beta_k.\mathbf{X_i}}}$$

and

$$\mathbb{P}(y_i = 0|\mathbf{X_i}, i \in \mathcal{C}_k) = \frac{1}{1 + e^{\beta_k.\mathbf{X_i}}}$$

We want to define the clusters (by allocating customers to clusters) and jointly estimate the logistic coefficients β_k for each cluster.

2.3.2 Logistic Model

We describe, in this paragraph, the logistic model estimation in the classical setting where $K = 1$, i.e. where clustering is not required. The logit model is by far the most commonly used discrete choice model. It is widely used because there exists a closed form formula for the choice probabilities which is easy to interpret. We will present here a short derivation of the model, for a detailed description see Chapter 3 of [7].

In a classical choice model setting, assume that a customer i faces a set of possible alternatives $j \in J$. For example, if $J = \{0, 1\}$, $j = 0$ corresponds to "not to buy" and $j = 1$ corresponds to "buy". Assume that customer i obtains from option j an utility U_{ij} that can be decomposed in:

$$U_{ij} = V_{ij} + \epsilon_{ij}$$

The classical choice model assumption is that V_{ij} is observed, while ϵ_{ij} is not. Furthermore customer i decides to choose option j if it is the option that generates the highest utility.

Thus customer i chooses option j_0 if

$$U_{ij_0} = \max_j U_{ij}$$

and this happen with probability

$$p_{i,j_0} = \mathbb{P}(U_{ij_0} = \max_j U_{ij})$$

The logit model is obtained assuming that

1. V_{ij} is a linear function of a customer-alternative vector X_{ij}

2. the residuals ϵ_{ij} are independent uniformly distributed according to the type I extreme value distribution (also called Gumbel distribution)

The cumulative distribution function of the Gumbel distribution is given by

$$F(\epsilon) = e^{e^{-\epsilon}}$$

Thus customer i chooses option j_0 with probability

$$p_{i,j_0} = \mathbb{P}(U_{ij_0} = \max_j U_{ij}) = \frac{e^{V_{ij_0}}}{\sum_j e^{V_{ij}}}$$

where the last equality comes from integrating the cumulative distribution function F.

In the case where there are only two options, we can denote $y_i \in \{0, 1\}$ the choice of customer i, and using the linearity assumption for V_{ij} we get:

$$\mathbb{P}(y_i = 1|\mathbf{X_i}) = \frac{e^{\beta.\mathbf{X_i}}}{1 + e^{\beta.\mathbf{X_i}}} \text{ and } \mathbb{P}(y_i = 0|\mathbf{X_i}) = \frac{1}{1 + e^{\beta.\mathbf{X_i}}}$$

where $\mathbf{X_i}$ is the vector of features of customer i.

The parameter of this model is β and it is traditionally estimated through maximum likelihood. If we observe N customers with attributes $(\mathbf{X_i}, y_i)$ then the likelihood is

$$L(\beta) = \prod_{i=1}^{N} \mathbb{P}(y_i = 1|\mathbf{X_i})^{y_i} \mathbb{P}(y_i = 0|\mathbf{X_i})^{1-y_i} = \prod_{i=1}^{N} \left(\frac{e^{\beta.\mathbf{X_i}}}{1 + e^{\beta.\mathbf{X_i}}}\right)^{y_i} \left(\frac{1}{1 + e^{\beta.\mathbf{X_i}}}\right)^{1-y_i}$$

and the log-likelihood becomes:

$$\mathcal{L}(\beta) = \log(L(\beta)) = \sum_{i=1}^{N} y_i \beta.\mathbf{X_i} - \ln(1 + e^{\beta.\mathbf{X_i}})$$

The log-likelihood $\mathcal{L}$ is a concave function of β and the maximum likelihood estimator

defined by

$$\hat{\beta} = \underset{\beta}{\text{argmax}}\ \mathcal{L}(\beta)$$

is consistent ($\lim_{N\to\infty} \hat{\beta} = \beta$) and efficient (i.e. an estimator with minimum mean squared error among the class of consistent estimators).

2.3.3 Problem Definition

In our case, we want to adapt the classical maximum likelihood approach for the logit model (also called logistic regression) to incorporate different clusters. Assume that you have K clusters and that for each cluster a different logit function describes the decision process. We use a maximum likelihood approach to jointly cluster and estimate the logit coefficients. Intuitively, we want to maximize the overall likelihood, knowing that every customer has to be assigned to exactly one cluster.

We want to find K clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$ and K vectors $\beta_1, \ldots, \beta_K$ in order to maximize:

$$\max_{\beta_1,\ldots,\beta_K} \sum_{k=1}^{K} \sum_{i\in\mathcal{C}_k} y_i \beta_k . \mathbf{X_i} - \ln(1 + e^{\beta_k . \mathbf{X_i}}) \tag{2.1}$$

2.4 Joint clustering and logistic regression

In this Section, we first formulate the problem of joint clustering and logistic regression as a non-linear mixed-integer program. We show that this first formulation is intractable because of a non concave objective function with binary variables. We then state our main contribution: we provide an equivalent formulation of the problem that can be solved efficiently in terms of computational time by commercial solvers even with a large number of customers.

2.4.1 Problem Formulation

The objective is to allocate the customers into K clusters and estimate a logistic demand function for each cluster in order to maximize the overall likelihood. This allows to capture the different segments in the population and can significantly improve the accuracy of the prediction.

As stated in equation (2.1), this can be achieved by finding K clusters $\mathcal{C}_1, \ldots, \mathcal{C}_K$ and K vectors $\beta_1, \ldots, \beta_K$ that maximize:

$$\max_{\beta_k, \mathcal{C}_k} \sum_{k=1}^{K} \sum_{i \in \mathcal{C}_k} y_i \beta_k . \mathbf{X_i} - \ln(1 + e^{\beta_k . \mathbf{X_i}})$$

This problem can be written as an integer program with binary variables $a_{i,k}$ defined by:

$$a_{i,k} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to cluster } \mathcal{C}_k \\ 0 & \text{otherwise} \end{cases}$$

Every customer has to be assigned to exactly one cluster. This can be enforced with the following constraints on $a_{i,k}$

$$\begin{cases} \forall (i,k) & a_{i,k} \in \{0,1\} \\ \forall i & \sum_{k=1}^{K} a_{i,k} = 1 \end{cases}$$

If the sum of binary variables equals 1, then exactly one of these variables is equal to 1 and the others are equal to 0. This translates into the fact that every customer has to be assigned to exactly one cluster.

The joint clustering and logistic regression problem can thus be written as:

$$\begin{aligned} \max_{\beta_k, a_{i,k}} \quad & \sum_{k,i} a_{i,k} \left[y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i}) \right] \\ \text{subject to} \quad & \sum_{k} a_{i,k} = 1, \ i = 1, \ldots, N. \\ & a_{i,k} \in \{0,1\}, \ i = 1, \ldots, N, \ k = 1, \ldots, K. \\ & \beta_k \in \mathbb{R}^m, \ k = 1, \ldots, K. \end{aligned} \tag{2.2}$$

In the objective function of (2.2), for every cluster k we sum over the customers where $a_{i,k} = 1$ thus those that are assigned to cluster k. It is easy to see that problem (2.2) is equivalent to (2.1).

This formulation has a non-concave objective function containing integer variables, thus standard gradient descent techniques cannot be applied in this setting. Solving (2.2) is thus

a challenging optimization problem. On simulations, we observed that commercial solvers can take several hours to solve such an instance even with a small number of customers.

2.4.2 Reformulation

Proposition 1.1. *Problem (2.2) is equivalent to*

$$\max_{\beta_k, a_{i,k}, \Delta_i} \quad \sum_i \Delta_i y_i - \ln(1 + e^{\Delta_i})$$

s. t.

$$\begin{aligned}
&(1-2y_i)\Delta_i \geq (1-2y_i)\beta_k.\mathbf{X}_i - M(1-a_{i,k})\ i = 1,\dots,N,\ k = 1,\dots,K. \\
&\sum_k a_{i,k} = 1,\ i = 1,\dots,N. \qquad (2.3) \\
&a_{i,k} \in \{0,1\},\ i = 1,\dots,N,\ k = 1,\dots,K. \\
&\beta_k \in \mathbb{R}^m,\ k = 1,\dots,K. \\
&\Delta_i \in \mathbb{R},\ i = 1,\dots,N.
\end{aligned}$$

where M is a "big" constant.

In problem (2.3), $y_i, \mathbf{X_i}$ and M are input and the decision variables are the logistic regression coefficients (β_k), the binary variables that associate customers to clusters ($a_{i,k}$) and Δ_i. The definition of the variable Δ_i is presented in the proof of Proposition 1.1. Intuitively, Δ_i comes from a change of variable and $\Delta_i = \beta_k.\mathbf{X}_i$ for k such that $a_{i,k} = 1$. Note that the constraints are linear in terms of the decision variables. Thus problem (2.3) is a mixed-integer optimization problem with linear constraints. In our application $M \geq \max\{\max X_i, 1000\}$ is sufficient (The constant 1000 is chosen in order to have $e^{-M} \simeq 0$).

Proof. The proof of Proposition 1.1 can be decomposed into 4 major steps.

1. The first step aims to remove the binary variables $a_{i,k}$ from the objective function of problem (2.2). We follow the approach used in [5], and do the change of variables

"$\delta_i = y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i})$, if $a_{i,k} = 1$". This can be rewritten as

$$\begin{aligned}
&\max_{\beta_k, a_{i,k}, \delta_i} \quad \sum_i \delta_i \\
&\text{subject to} \\
&\qquad \delta_i \leq y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i}), \ \text{ if } a_{i,k} = 1 \\
&\qquad \sum_k a_{i,k} = 1, \ i = 1, \ldots, N. \\
&\qquad a_{i,k} \in \{0, 1\}, \ i = 1, \ldots, N, \ k = 1, \ldots, K. \\
&\qquad \beta_k \in \mathbb{R}^m, \ k = 1, \ldots, K. \\
&\qquad \delta_i \in \mathbb{R}, \ i = 1, \ldots, N.
\end{aligned} \tag{2.4}$$

Note that the equality constraint presented in the change of variables is relaxed to an inequality. This is due to the fact that we are maximizing over δ_i thus the constraint

$$\delta_i \leq y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i}), \ \text{ if } a_{i,k} = 1$$

will be tight at optimality and thus equivalent to

$$\delta_i = y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i}), \ \text{ if } a_{i,k} = 1$$

Adding the variables δ_i, we have transformed problem (2.2) into an equivalent problem without binary variables in the objective.

2. In the second step, we take advantage of some monotonicity properties of the logistic function. Following the approach used in [6], let us introduce two functions:

$$f_1(x) = x - \ln(1 + e^x)$$

f_1 is strictly increasing and strictly concave. Let f_1^{-1} be its inverse.

$$f_0(x) = -\ln(1 + e^x)$$

f_0 is strictly decreasing and strictly concave. Let f_0^{-1} be its inverse.

Recall that y_i is data and takes *binary values* thus, the first constraint of (2.4) can be rewritten using the functions f_1 and f_0.

- If $y_i = 1$

$$\delta_i \leq y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i})$$

becomes

$$\delta_i \leq f_1(\beta_k . \mathbf{X}_i) \Leftrightarrow f_1^{-1}(\delta_i) \leq \beta_k . \mathbf{X}_i \tag{2.5}$$

- If $y_i = 0$

$$\delta_i \leq y_i \beta_k . \mathbf{X}_i - \ln(1 + e^{\beta_k . \mathbf{X}_i})$$

becomes

$$\delta_i \leq f_0(\beta_k . \mathbf{X}_i) \Leftrightarrow f_0^{-1}(\delta_i) \geq \beta_k . \mathbf{X}_i \tag{2.6}$$

Note that in equation (2.6) the sense of the inequality is reversed because f_0 is a decreasing function.

3. We then introduce one last set of variables defined by:

$$\Delta_i = \begin{cases} f_0^{-1}(\delta_i) & \text{if } y_i = 0 \\ f_1^{-1}(\delta_i) & \text{if } y_i = 1 \end{cases}$$

This definition allows us to rewrite the equations as

$$\delta_i = f_1(\Delta_i) y_i + f_0(\Delta_i)(1 - y_i)$$

And, replacing δ_i by the previous expression in (2.4), we have

$$\max_{\beta_k, a_{i,k}, \Delta_i} \quad \sum_i f_1(\Delta_i) y_i + f_0(\Delta_i)(1 - y_i)$$

subject to

$$\begin{aligned}
&\Delta_i \leq \beta_k.\mathbf{X}_i, \quad \text{if } y_1 = 1 \text{ and } a_{i,k} = 1 \\
&\Delta_i \geq \beta_k.\mathbf{X}_i, \quad \text{if } y_1 = 0 \text{ and } a_{i,k} = 1 \\
&\sum_k a_{i,k} = 1, \ i = 1, \ldots, N. \\
&a_{i,k} \in \{0, 1\}, \ i = 1, \ldots, N, \ k = 1, \ldots, K. \\
&\beta_k \in \mathbb{R}^m, \ k = 1, \ldots, K. \\
&\Delta_i \in \mathbb{R}, \ i = 1, \ldots, N.
\end{aligned} \tag{2.7}$$

4. We then rewrite the "if..then" constraints of problem (2.7) as linear constraints using a "big M" constant. Let M be a large constant (larger than $\max_{i,j} X_{i,j}$). Then, in (2.7), we can replace

$$\begin{cases} \Delta_i \leq \beta_k.\mathbf{X}_i, & \text{if } y_1 = 1 \text{ and } a_{i,k} = 1 \\ \Delta_i \geq \beta_k.\mathbf{X}_i, & \text{if } y_1 = 0 \text{ and } a_{i,k} = 1 \end{cases}$$

by

$$(1 - 2y_i)\Delta_i \geq (1 - 2y_i)\beta_k.\mathbf{X}_i - M(1 - a_{i,k}) \ i = 1, \ldots, N, \ k = 1, \ldots, K.$$

In fact,

- If $a_{i,k} = 0$ the right-hand side of the previous equation becomes very negative (if M is big $-M$ approximates $-\infty$) and there is no constraint on (Δ_i, β_k).
- If $a_{i,k} = 1$ and $y_i = 0$ then the previous equation becomes $\Delta_i \geq \beta_k.\mathbf{X}_i$.
- If $a_{i,k} = 1$ and $y_i = 1$ then the previous equation becomes $-\Delta_i \geq -\beta_k.\mathbf{X}_i \Leftrightarrow \Delta_i \leq \beta_k.\mathbf{X}_i$.

This is exactly what is enforced by the two first constraints in (2.7).

Thus formulation (2.7) can be rewritten as

$$\max_{\beta_k, a_{i,k}, \Delta_i} \quad \sum_i f_1(\Delta_i) y_i + f_0(\Delta_i)(1 - y_i)$$

subject to

$$\begin{aligned}
&(1-2y_i)\Delta_i \geq (1-2y_i)\beta_k.\mathbf{X}_i - M(1-a_{i,k})\ i = 1,\ldots,N,\ k = 1,\ldots,K.\\
&\sum_k a_{i,k} = 1,\ i = 1,\ldots,N.\\
&a_{i,k} \in \{0,1\},\ i = 1,\ldots,N,\ k = 1,\ldots,K.\\
&\beta_k \in \mathbb{R}^m,\ k = 1,\ldots,K.\\
&\Delta_i \in \mathbb{R},\ i = 1,\ldots,N.
\end{aligned} \tag{2.8}$$

Finally, we replace f_0 and f_1 by their expression and obtain formulation (2.3).

$$\max_{\beta_k, a_{i,k}, \Delta_i} \quad \sum_i \Delta_i y_i - \ln(1 + e^{\Delta_i})$$

s. t.

$$\begin{aligned}
&(1-2y_i)\Delta_i \geq (1-2y_i)\beta_k.\mathbf{X}_i - M(1-a_{i,k})\ i = 1,\ldots,N,\ k = 1,\ldots,K.\\
&\sum_k a_{i,k} = 1,\ i = 1,\ldots,N.\\
&a_{i,k} \in \{0,1\},\ i = 1,\ldots,N,\ k = 1,\ldots,K.\\
&\beta_k \in \mathbb{R}^m,\ k = 1,\ldots,K.\\
&\Delta_i \in \mathbb{R},\ i = 1,\ldots,N.
\end{aligned}$$

□

Proposition 1.2. *Formulation (2.3) gives rise to a concave maximization problem with linear constraints.*

Proof. The proof follows as:

- $f : \Delta \to \Delta y_i - \ln(1 + e^{\Delta})$ is a strictly concave function.
- The constraints in Problem (2.3) are linear in terms of the variables (Δ_i, β_k).

□

Note that the objective function of (2.3) is the sum of strictly concave functions of a single variable. Thus its gradient and Hessian are easy to compute. Furthermore, because of the special shape of the objective function, we can approximate $\Delta \to -\ln(1+e^{\Delta})$ by a piece-wise linear function and get an accurate approximation of the solution by solving a linear mixed integer program.

2.4.3 Adding clustering constraints

In the previous Section, we have shown that the joint clustering and logistic regression problem can be reformulated as a concave mixed integer problem. This can be solved efficiently in terms of computational time with a "large" amount of customers as illustrated in the application in Section 2.7. Nevertheless, the formulation proposed in (2.3) leads to overfitting, clustering constraints need to be added.

Remark 1.1. *Formulation (2.3) leads to overfitting.*

Proof. Formulation (2.3) does not include any constraint on how customers should be assigned to clusters. It just enforces that every customer is assigned to exactly one cluster. This lack of "clustering rules" leads to overfitting. Intuitively, the formulation allows to classify the customers according to their purchase behavior (variable y_i), this leads to a perfect classification on the training set but a total lack of prediction power for new unobserved data points.

In order to prove this, let us divide the data set into $\mathcal{D}_b$ the subset of buyers ($y_i = 1$) and $\mathcal{D}_{nb}$ the subset of non-buyers ($y_i = 0$). Recall that a model with perfect fitting accurately predicts the purchase behavior of every single customer. A perfect fitting model predicts that $\mathbb{P}(y_i = 1) = 1$ for customers in $\mathcal{D}_b$ and $\mathbb{P}(y_i = 1) = 0$ for customers in $\mathcal{D}_{nb}$. Recall that the likelihood of a logistic model is given by $\prod_i \mathbb{P}(y_i = 1|X_i)^{y_i}\mathbb{P}(y_i = 0|X_i)^{1-y_i}$. A likelihood is always smaller than 1 and a model with perfect fitting achieves a likelihood value of 1, which translates into a log-likelihood value of 0.

Let us consider that $K \geq 2$ (otherwise the problem is reduced to a simple logistic regression). Let us also assume without loss of generality that $\mathbf{X}_i \geq 0$ for all i. Let us build

a feasible solution for Problem (2.3) that allocates the customers in $\mathcal{D}_b$ to cluster 1 and the customers in $\mathcal{D}_{nb}$ to cluster 2. Let us also set $\beta_1 = B\mathbb{1}$ and $\beta_2 = -B\mathbb{1}$, where $\mathbb{1}$ is a vector of ones and B is a constant that goes to infinity. Let us also set $\Delta_i = \beta_1 X_i$ if $i \in \mathcal{D}_b$ and $\Delta_i = \beta_2 X_i$ otherwise.

We can verify that this solution is feasible for Problem (2.3) as every customer is assigned to exactly one cluster and the first constraint of (2.3) is verified.

We then have,

$$\mathbb{P}(y_i = 1 | X_i, i \in \mathcal{C}_1) = \frac{e^{B\mathbb{1}.X_i}}{1 + e^{B\mathbb{1}.X_i}}$$

$$\mathbb{P}(y_i = 0 | X_i, i \in \mathcal{C}_2) = \frac{1}{1 + e^{-B\mathbb{1}.X_i}}$$

By definition, $y_i = 1$ for every i in $\mathcal{C}_1$ and $y_i = 0$ for every i in $\mathcal{C}_2$ thus the likelihood of this problem is

$$\prod_{i \in \mathcal{C}_1} \frac{e^{B\mathbb{1}.X_i}}{1 + e^{B\mathbb{1}.X_i}} \prod_{i \in \mathcal{C}_2} \frac{1}{1 + e^{-B\mathbb{1}.X_i}}$$

which goes to 1 when B goes to infinity.

Thus, by defining cluster 1 as the cluster of buyers and cluster 2 as the cluster of non-buyers and setting $\beta_1 = B\mathbb{1}$ and $\beta_2 = -B\mathbb{1}$, we build a feasible solution to (2.3). When B goes to infinity this maximizes the likelihood (value of 1). Thus solving formulation (2.3) creates perfect fitting. □

Formulation (2.3) can still be extremely valuable if we add clustering rules to avoid this overfitting issue, namely to prevent from allocating customers according to the customer future purchase behavior (the dependent variable y). There are several possible ways of adding such constraints. Our approach is motivated by two main reasons.

First of all, recall the business model of the online retailer. Customers have to be clustered into categories *according to their social activity* with transparent and hierarchical rules. If a customer is strictly more active socially than another he cannot be assigned to a lower category. Furthermore, customers are allocated to categories following "threshold rules". For example, in Figure 1-1, "sharers" are defined as customers with *at least* n_1 social networks, n_2 friends, n_3 items shared and n_4 reviews. Adding this type of constraints and adding n_1, n_2, n_3, n_4 as decision variables avoids overfitting and allows to implement and optimize

the retailer business model. Another motivation for these clustering rules is that, as we will prove later, they can be written as linear constraints on the decision variables. This is a great advantage, as adding linear constraints to formulation (2.3) does not change the complexity of the problem.

Definition 1.1. *Assume that a set of customers* $\{i \in [1, \ldots, N]\}$ *each defined by a vector of non-negative features* $x_i \in \mathbb{R}^M$ *need to be clustered into* K *groups.*

A "Threshold rule" is a clustering rule defined by K *vectors* $\Gamma^1, \ldots, \Gamma^K \in \mathbb{R}^M$ *such that* $\Gamma^1_j \leq \ldots \leq \Gamma^K_k$ *for all* $j = 1 \ldots M$.

Customer i *is allocated to cluster* k *if and only if*

$$x_{i,j} \geq \Gamma^{k-1}_j \forall j \text{ AND } \exists j \text{ such that } x_{i,j} < \Gamma^k_j \tag{2.9}$$

where $\Gamma^0 = 0$

Example 1.1. *For the sake of simplicity, we illustrate the definition of a threshold rule in the simple case where* $K = 2$. *Note that the same approach can be generalized to any number of clusters* K.

Assume that we want to build two clusters denoted by "Low" and "High". Cluster "High" will capture the social customers and "Low" the rest. Assume that we want to differentiate the two clusters according to their number of friends (denoted by F*) and their number of reviews (denoted by* R*). Then, using the retailer threshold business model we need to define two threshold values* Γ_f *and* Γ_r *and allocate customers to clusters according to the following rule:*

$$\begin{aligned} &\text{if } F_i \geq \Gamma_f \text{ AND } R_i \geq \Gamma_r \text{ then assign } i \text{ to cluster High} \\ &\qquad\qquad \text{else assign } i \text{ to cluster Low} \end{aligned} \tag{2.10}$$

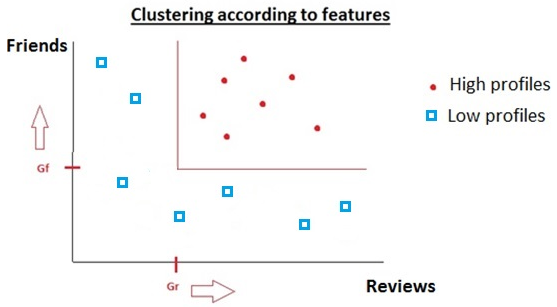


Figure 2-1: Threshold Clustering

This clustering rule is illustrated in Figure 2-1 where cluster High is represented in red and cluster Low in blue. This follows the retailer business model in our setting and constraints (2.10) can be rewritten as linear constraints with binary variables.

Proposition 1.3. *The constraints defining threshold rules (2.9) can be written as linear constraints with binary variables.*

Proof. We derive the result in the case where $K = 2$. The same approach can be used when $K > 2$. Consider a general setting where we want to cluster according to M features represented by the vector $x_i \in \mathbb{R}^M$ for every customer. Let us represent the threshold values in a vector Γ. Then the constraints in (2.10) can be rewritten $\begin{cases} a_{i,H} = 1 & \text{if } \forall j \; x_{i,j} \geq \Gamma_j \\ a_{i,L} = 1 & \text{if } \exists j \text{ such that } x_{i,j} < \Gamma_j \end{cases}$.

We can easily add the previous rule in our formulation using linear constraints.

Let us define a constant Γ_{max} such that $\Gamma_j \leq \Gamma_{max} \; \forall j$. We define this constant using the maximum value of $x_{i,j}$ in the dataset for example. Let us also define ϵ a "small number". (If x_i has integer values $\epsilon = 0.5$ could be an example.)

- The statement $\exists j \; x_{i,j} < \Gamma_j \Rightarrow a_{i,L} = 1$ can be written in the following way:

$$a_{i,L} \geq \frac{1}{\Gamma_{max}}(\Gamma_j - \epsilon - x_{i,j})$$

  In fact if $x_{i,j} < \Gamma_j$ then $\frac{1}{\Gamma_{max}}(\Gamma_j - \epsilon - x_{i,j}) > 0$ and this will enforce $a_{i,L} = 1$.

  If $x_{i,j} \leq \Gamma_j$ this implies $a_{i,L} \geq 0$ which does not enforce any additional constraint on $a_{i,L}$.

- For the constraint on $a_{i,H}$ we have to define the binary variable $b_{i,j} = \begin{cases} 1 & \text{if } x_{i,j} \geq \Gamma_j \\ 0 & \text{otherwse} \end{cases}$.

  In order to do that we can define $b_{i,j}$ as a binary variable and write:

$$b_{i,j} \geq \frac{1}{\Gamma_{max}}(x_{i,j} - \Gamma_j) \; \forall j$$

$$1 - b_{i,j} \geq \frac{1}{\Gamma_{max}}(\Gamma_j - x_{i,j}) \; \forall j$$

  and then the constraint "$a_{i,H} = 1$ if $\forall j x_{i,j} \geq \Gamma_j$" becomes:

$$a_{i,H} \geq \sum_j b_{i,j} - m + 1$$

  where m is the number of features.

  This last constraint enforces $a_{i,H} = 1$ if and only if $\sum_j b_{i,j} = m$, which is equivalent to $x_{i,j} \geq \Gamma_j$ for all j.

Finally, the constraints in (2.10) can be written with linear constraints:

$$\begin{cases} a_{i,L} \geq \frac{1}{\Gamma_{max}}(\Gamma_j - \epsilon - x_{i,j}) & \forall i \forall j \\ b_{i,j} \geq \frac{1}{\Gamma_{max}}(x_{i,j} - \Gamma_j) & \forall i \forall j \\ 1 - b_{i,j} \geq \frac{1}{\Gamma_{max}}(\Gamma_j - x_{i,j}) & \forall j \\ a_{i,H} \geq \sum_j b_{i,j} - m + 1 & \forall i \\ b_{i,j} \in \{0, 1\} & \forall i \forall j \end{cases} \tag{2.11}$$

□

The same approach can be used with any number of cluster K. In general it is possible to implement "If.. then" constraints with linear constraints. Sometimes introduction of auxiliary binary variables is required. In our application, this does not change significantly the complexity of our formulation as we already have binary variables and linear constraints.

Generally the number of features in the clustering (m) is small (4 in Figure 1-1), and we need to add $m \times N$ additional binary variables.

2.4.4 Summary

We have built a mixed integer concave optimization problem for joint clustering and logistic regression. It takes as an input the desired number of clusters K and a set of customers, each defined by two vectors of features: X_i used in the logistic regression and x_i used in the clustering ($x = (Friends, Reviews)$ in Example 1.1). It finds the threshold coefficients $\Gamma_1, \dots \Gamma_K$ and the logistic regression coefficients $\beta_1, \dots, \beta_K$ that maximize the overall log-likelihood. This optimally clusters customers into categories according to threshold rules AND *jointly* estimate a distinct logistic demand function for each cluster. This allows to efficiently capture customers' heterogeneity and build a more accurate demand model differentiated across customers' segments. To conclude, note that we have chosen here to use threshold rules for the clustering to follow the retailer business model. This is not a requirement in our formulation. Any type of linear separators can be used without significantly increasing the complexity of the problem. In Example 1.1, a hyperplane separator can be chosen (as illustrated in Figure 2-2). In this case, the decision variable Γ is replaced by the intercept and slope of the separating hyperplane.

Figure 2-2: Clustering with an hyperplane

2.5 Joint clustering and multinomial logit

In the previous Section, we have formulated the problem of joint clustering and logistic regression as a mixed integer concave optimization problem. The same approach can be used when the customer faces more than two possible choices. We will show here that the same approach can be extended to multinomial logit choice models.

2.5.1 Multinomial logit choice model

The multinomial logit model is a generalization of the logit model where customers have the choice between $J \geq 2$ options. A classical transportation application is when customers can choose between several transportation means: bus, train, bicycle, car.... A probability is associated with each of the choices according to customer/choice attributes. In a online retailing setting, customers have the choice between several items on sale. The set of available options can be {"not to buy","buy item 1","buy item 2","buy item 3",...}. The multinomial logit model can be derived from the discussion in Section 2.3.2. Let us assume that customers have the choice between $J+1$ options where option 0 is the no purchase option and $j \in \{1, \ldots, J\}$ are the different purchase options. Let us assume, as in the previous Section, that customer i is characterized by a vector of features X_i. Then a multinomial logit model is defined by J vectors $\beta^1, \ldots, \beta^J$. Customer i chooses option $j \geq 1$ with probability:

$$\frac{e^{\beta^j . \mathbf{x}_i}}{1 + \sum_{u=1}^{J} e^{\beta^u . \mathbf{x}_i}}$$

and chooses the no purchase option $j = 0$ with probability:

$$\frac{1}{1 + \sum_{u=1}^{J} e^{\beta^u . \mathbf{x}_i}}$$

Note that if $J = 1$ we recover exactly the logit model with binary choice.

2.5.2 Notations and problem definition

We want to generalize the approach presented in Section 2.4 to the case where customers face more than two choices. We start by introducing the notations we will use in this Section

and then formulate the joint clustering and multinomial logit problem.

Notations: To be consistent with the previous part we will adopt the following notations:

- i index of a customer, $i \in \{1, \ldots, N\}$
- k index of a cluster, $k \in \{1, \ldots, K\}$
- j index of a choice, $j \in \{0, \ldots, J\}$
- β_k^j parameters of choice j in cluster k
  β_k^0 is set to 0 for every cluster k
- $X_i \in \mathbb{R}^m$ vector of features of customer i
- decision variable that allocates customers to clusters

$$a_{i,k} = \begin{cases} 1 & \text{if customer } i \text{ is allocated to cluster } k \\ 0 & \text{otherwise} \end{cases}$$

- choice decision variable (data)

$$y_{i,j} = \begin{cases} 1 & \text{if customer } i \text{ chooses option } j \\ 0 & \text{otherwise} \end{cases}$$

  Every customer makes exactly one choice thus $\sum_j y_{i,j} = 1$.

The overall likelihood becomes:

$$\prod_{i,k,j} \left(\frac{e^{\beta_k^j . \mathbf{X}_i}}{1 + \sum_{u=1}^{J} e^{\beta_k^u . \mathbf{X}_i}} \right)^{a_{i,k} y_{i,j}} = \prod_{i,k,j} \left(\frac{e^{y_{i,j} \beta_k^j . \mathbf{X}_i}}{1 + \sum_{u=1}^{J} e^{\beta_k^u . \mathbf{X}_i}} \right)^{a_{i,k}}$$

because for every customer $\sum_j y_{i,j} = 1$.

Problem Formulation: Similarly to Section 2.4, we want to allocate customers to clusters (decision variables $a_{i,k}$) and find the logistic coefficients β_k^j (for each cluster and option) in

order to maximize the overall log-likelihood. The log-likelihood maximization problem can be written:

$$\begin{aligned}
\max_{\beta_k^j, a_{i,k}} \quad & \sum_{i,k} a_{i,k} \left[\sum_{j>0} \left(y_{i,j} \beta_k^j . \mathbf{X}_i \right) - \ln \left(1 + \sum_{j>0} e^{\beta_k^j . \mathbf{X}_i} \right) \right] \\
\text{s. t.} \quad & \\
& \sum_k a_{i,k} = 1, \ i = 1, \ldots, N. \\
& a_{i,k} \in \{0, 1\}, \ i = 1, \ldots, N, \ k = 1, \ldots, K. \\
& \beta_k^j \in \mathbb{R}^m, \ k = 1, \ldots, K \ j = 1, \ldots J
\end{aligned} \tag{2.12}$$

We use similar techniques to Section 2.4 to reformulate this problem into a mixed-integer concave optimization with linear constraints. We start the derivation with the simple case where $J = 2$ and then show that the same approach can be used for any value of J.

2.5.3 Reformulation when customers have two choices

Let us consider the case where $J = 2$. The customer can choose between {"not to buy","buy item 1","buy item 2"}. The objective function of problem (2.12) becomes:

$$\sum_{i,k} a_{i,k} \left[y_{i,1} \beta_k^1 . \mathbf{X}_i + y_{i,2} \beta_k^2 . \mathbf{X}_i - \ln \left(1 + e^{\beta_k^1 . \mathbf{X}_i} + e^{\beta_k^2 . \mathbf{X}_i} \right) \right]$$

Proposition 1.4. *The joint clustering and multinomial logistic problem with two choices*

and K clusters can be reformulated into:

$$\begin{aligned}
&\max_{\beta_k^j, a_{i,k}, \delta_i, v_i} \quad \sum_i (1 - y_{i,0})\delta_i - \ln(1 + e^{\delta_i} + e^{v_i}) \\
&s.\ t. \\
&\qquad (-1 + 2y_{i,0})\delta_i \geq y_{i,0}\beta_k^{\mathbf{1}}.\mathbf{X}_i - \sum_{l>0} y_{i,j}\beta_k^j.\mathbf{X}_i, \ \textit{if } a_{i,k} = 1 \\
&\qquad v_i \geq y_{i,0}\beta_k^{\mathbf{2}}.\mathbf{X}_i + (1 - y_{i,0}) \sum_{j>0} (1 - y_{i,j})\beta_k^j.\mathbf{X}_i, \ \textit{if } a_{i,k} = 1 \qquad (2.13) \\
&\qquad \sum_k a_{i,k} = 1, \ i = 1, \ldots, N. \\
&\qquad a_{i,k} \in \{0, 1\}, \ i = 1, \ldots, N, \ k = 1, \ldots, K. \\
&\qquad \beta_k^j \in \mathbb{R}^m, \ k = 1, \ldots, K \ j = 0, \ldots J
\end{aligned}$$

This formulation has a strictly concave objective function and linear constraints.

Proof. The goal is to remove the binary decision variables $a_{i,k}$ from the objective function to make the problem more tractable. This is done in two steps.

Let us focus on a customer i and on cluster k such that $a_{i,k} = 1$. Let us distinguish two cases:

- If $y_{i,0} = 1$ then $y_{i,1} = y_{i,2} = 0$ and the objective function can be rewritten as

$$-\ln(1 + e^{\beta_k^1.\mathbf{X}_i} + e^{\beta_k^2.\mathbf{X}_i})$$

  Following the approach used in Section 2.4, we introduce the auxiliary variables δ_i and v_i. Note that we need two variables here instead of one because $J = 2$. Using the change of variable $\delta_i = \beta_k^1.\mathbf{X}_i$ and $v_i = \beta_k^2.\mathbf{X}_i$ we get

$$\begin{aligned}
&\max \quad -\ln(1 + e^{\delta_i} + e^{v_i}) \\
&\text{s. t.} \quad \delta_i \geq \beta_k^1.\mathbf{X}_i \qquad (2.14) \\
&\qquad\quad v_i \geq \beta_k^2.\mathbf{X}_i
\end{aligned}$$

  Note that the equality constraints can be relaxed because the objective is decreasing

in both variables.

- If $y_{i,0} = 0$ then $y_{i,1}$ or $y_{i,2}$ is equal to 1. Let us define: $\delta_i = y_{i,1}\beta_k^1.\mathbf{X}_i + y_{i,2}\beta_k^2.\mathbf{X}_i$ and $v_i = (1 - y_{i,1})\beta_k^1.\mathbf{X}_i + (1 - y_{i,2})\beta_k^2.\mathbf{X}_i$. (Let $j, \bar{j} \in [1, \ldots, J]$, the previous definitions enforce that if the customer chooses option j and rejects choice $\bar{j}$ then $\delta_i = \beta_k^j.\mathbf{X}_i$ and $v_i = \beta_k^{\bar{j}}.\mathbf{X}_i$.)

  and the maximization problem can be reformulated:

$$\begin{aligned} \max \quad & \delta_i - \ln(1 + e^{\delta_i} + e^{v_i}) \\ \text{s. t.} \quad & \delta_i \leq y_{i,1}\beta_k^1.\mathbf{X}_i + y_{i,2}\beta_k^2.\mathbf{X}_i \\ & v_i \geq (1 - y_{i,1})\beta_k^1.\mathbf{X}_i + (1 - y_{i,2})\beta_k^2.\mathbf{X}_i \end{aligned} \tag{2.15}$$

  where, again, the equality constraints have been relaxed because the objective is increasing in δ and decreasing in v.

Similarly to Section 2.4, we can put these two cases together using linear constraints and obtain formulation (2.13).

Finally, as illustrated in the previous Section we can translate the "If $a_{i,k} = 1$ then..." constraints into linear constraints by introducing a "big M" parameter.

Again, the objective function of problem (2.13) is strictly concave and the constraints are linear. Similarly as before, we can then add clustering constraints to avoid overfitting. We will not present the details here as the implementation is exactly the same as when the choice is binary.

□

2.5.4 Reformulation when $J \geq 2$

The same exact approach can be used in the case where $J \geq 2$. The steps of the reformulation are similar to the case $J = 2$ but in addition we have to make a distinction between the customers that choose the no buy option and the others. We then add auxiliary variables for every pair (customer, choice) and we take advantage of the monotonicity of the objective function to relax the change of variable equalities into inequalities.

Proposition 1.5. *The joint clustering and multinomial logistic problem with J choices and K clusters can be reformulated into:*

$$
\begin{aligned}
&\max_{\beta_k^j, a_{i,k}, \delta_i, v_i^j} \quad \sum_i (1 - y_{i,0})\delta_i - \ln(1 + e^{\delta_i} + \sum_{l \geq 2} e^{v_i^j}) \\
&s.\ t. \\
&\qquad (-1 + 2y_{i,0})\delta_i \geq y_{i,0}\beta_k^1.\mathbf{X}_i - \sum_{j>0} y_{i,j}\beta_k^j.\mathbf{X}_i,\ \textit{if } a_{i,k} = 1 \\
&\qquad v_i^j \geq y_{i,0}\beta_k^1.\mathbf{X}_i + (1 - y_{i,0})\left[(1 - y_{i,j})\beta_k^j.\mathbf{X}_i + y_{i,l}\beta_k^1.\mathbf{X}_i\right],\ \textit{if } a_{i,k} = 1 \\
&\qquad \sum_k a_{i,k} = 1,\ i = 1, \ldots, N. \\
&\qquad a_{i,k} \in \{0, 1\},\ i = 1, \ldots, N,\ k = 1, \ldots, K. \\
&\qquad \beta_k^j \in \mathbb{R}^m,\ k = 1, \ldots, K\ j = 0, \ldots J
\end{aligned}
\tag{2.16}
$$

This formulation has a strictly concave objective function and linear constraints.

The proof can be found in Appendix A.2.

2.6 Data

In this Section, we use the data from SHOP.CA to build a Logistic Clustering to predict future consumption. First, we explain how we transformed our data set for this purpose.

We start by splitting our data into two time periods that we denote "Past" and "Future". The Past period is used to build the customers' features (X_i) and we use the Future period to determine the indicator of future purchases ($y_i \in \{0, 1\}$).

Definition 1.2. *We define*

- ***Past Period**: starts January 1st 2013 and ends October 31st 2013 (10 months)*
- ***Future Period**: starts November 1st 2013 and ends February 20th 2014 (4 months)*

On SHOP.CA's website, the mean return time (defined as the mean time between two consecutive purchases for a customer) is three months. We choose a four months period for the Future to capture a fraction large enough of returning customers. We start by considering

the set of customers that made at least one purchase and at least one social activity in the Past Period.

From the Past Period data we extracted social and transaction history features for every customer, the list of features we considered is reported in Table A.2 in the Appendix. Using the data in the Future period we build the customer specific indicator variable defined as

$$y_i = \begin{cases} 1 & \text{if customer } i \text{ makes at least a purchase in the Future period} \\ 0 & \text{otherwise} \end{cases}$$

2.6.1 Interested customers

It is common practice, when trying to predict customers' purchase behavior, to eliminate customers that are "not interested". Traditionally, the only tool online retailers use to define "interested customers" is transaction history. Thus, customers who have not done a purchase for a *long* period of time ("long" is defined with respect to the average return time to the website, for example) are considered as non interested (or as churners) and are discarded from the data used in the analysis. An example of a non interested customer is a customer who creates an account, makes a purchase and then never visits the website again. In e-commerce, this type of behavior is common because online retailers often offer important discounts on the first purchase to attract new clients. "Not interested" customers need to be removed from the data set to build an effective demand model. In our setting, we have several tools to define interested and non interested customers. We have access to transaction history, but also browsing history (customers log ins) and social interactions. This allows for a more accurate definition of interested customers. A customer with frequent log ins or frequent social interactions can be categorized as interested. This would not have been possible if we had access only to transaction data. Removing from the analysis non interested customers is a crucial step and can be assimilated to outliers' detection. First of all, it allows to remove customers on which the retailer has too little information to be able to predict their future behavior. Secondly, and this is particularly important when dealing with social features, it reduces the sparsity of the data. As discussed before, when dealing with multiple customers' features, it is easy to encounter data sparsity: most of the features have a really

high mode at 0. Non interested customers create sparsity. Thus, by removing them, we build a more balanced data set. This gives more room to predictive algorithms to build efficient and robust predictions for future purchase behavior.

Our definition of "interested customers" comes from a discussion with SHOP.CA and their experience on their customers' purchase behavior. It is motivated by three main principles. First of all, we want to use only Past data (transaction and social interactions). Then, we want to capture customers who are *active* enough *recently* in terms of *social interactions* and *purchases*. In order to quantify "*active*" and "*recently*", we face a trade-off between eliminating all non interested customers and having a data set large enough to train a predictive model. We use the following definition of interested customers. Note that the 4 months time period is motivated again from the mean return time on the website.

Definition 1.3. *We define as "interested customers" all the customers with, in the last 4 months of the Past period, at least:*

- *one purchase*
- *one social activity*
- *two active days*

Focusing only on interested customers, we create a data set of 503 customers where 37% of them make a purchase in the future. Note that the definition of interested customers is only based on Past data and allows to consider only customers who are active on the website recently and thus who are potential candidates for making purchases in the future. The vast majority of the customers we discarded do not make a purchase in the future. Figure A-3 summarizes the histograms of some social and transaction features for interested customers. Note that focusing on interested customers allows to decrease the sparsity of the data, but the distributions of social activities are still skewed towards 0. Histograms of the distributions of some key features of interested customers are reported in Figure A-3 in Appendix A.1.

Restricting the analysis to interested customers has thus two main advantages. First, it allows to create a more balanced data set where a significant amount of customers buy in the

future. This helps comparing the performance of different predictive algorithms. Secondly, the data set becomes less sparse.

2.7 Implementation and results

In this Section, we solve the formulation presented in Section 2.4 for the set of interested customers (Definition 1.3). We then analyze its predictive performance and compare it to other predictive models that we use as benchmarks. We show that using clustering and incorporating social features increases significantly the predictive power. Finally, we analyze the sensitivity of our model to the data with a bootstrap approach.

2.7.1 Joint clustering and logistic regression for the set of interested customers

We focus on the set of interested customers defined in Definition 1.3. Because of the sparsity of the data, we build $K = 2$ clusters.

Clustering features selection (x)

Customers are assigned to clusters using a threshold implementation according to a vector of features x that can incorporate social or transaction history from the Past Period. We want to choose the features present in x in order to build two clusters approximately balanced (i.e. with approximately the same number of customers in each cluster). As explained previously, most of interested customers have few social interactions. Thus, it was not possible to include the social features presented in Table A.2. Because of the structure of the threshold rules, the vast majority of the customers would have been assigned to cluster "Low". Thus we include only the feature "Number of Purchases" in the clustering vector x. This choice is exclusively motivated by the sparsity issue. With a richer data set in terms of social interactions, one can try different combinations of features for the vector x in order to have the best prediction fit.

Regression features selection (X)

To select the features to be included in the logistic model (X) we use a greedy backward selection strategy. We start by considering in vector X all the features presented in Table A.2 and get the threshold value Γ and the logistic coefficients β_1 and β_2. Then, *for each cluster separately*, we run a logistic regression and compute the standard deviation and p-value for every feature. We use these p-values to eliminate in an iterative way the features that have p-values above 0.05 *for both clusters*. Note that the coefficients β_1 and β_2 found by the joint formulation and the logistic regressions are the same, we use the second stage regression only to get p-values and select significant features.

Solving the joint formulation

As mentioned in Section 2.4, the objective function of problem (2.3) is the sum of strictly concave functions of one variable: $\sum_i f_i(\Delta_i)$, where $f_i(x) = y_i x - \ln(1+e^x)$. The constraints of problem (2.3) are linear. In order to efficiently solve this problem, we approximate the functions f_i by piecewise-linear functions and transform problem (2.3) into a linear mixed-integer program. Note the specific shape of the functions f_i are illustrated in Figures 2-3 and 2-4. f_1 and f_0 have linear asymptotes in $+$ and $-$ infinity, this allows for accurate approximations by piece-wise linear functions, without having to introduce a large number of breakpoints (only 3 are used in Figures 2-3 and 2-4).

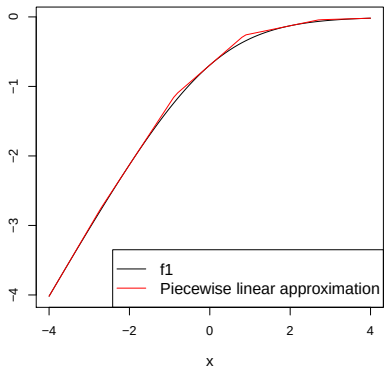


Figure 2-3: $f_1(x) = x - \ln(1 + e^x)$

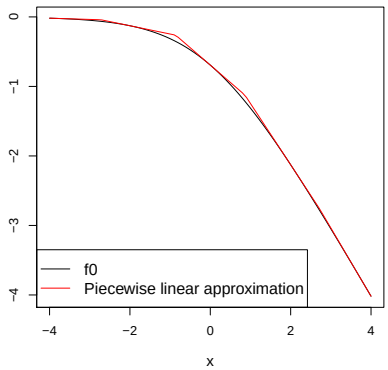


Figure 2-4: $f_0(x) = -\ln(1 + e^x)$

Solving the joint clustering and logistic regression problem takes under 10 minutes with a laptop with Intel Core i5-2430M CPU with 2.40 GHz and 2.40 GHz processor, 4 GB RAM and 64-bit Windows operating system. This is a reasonable run time for the demand estimation problem as it does not need to be updated as frequently.

Results

With the data set of "interested customers" and using *Number of Past Purchases* for clustering we find that the threshold value $\Gamma = 3$. Thus, customers with less than 3 Past Purchases are assigned to cluster Low and customers with at least 3 Past Purchases are assigned to cluster High. Out of the 503 interested customers, 212 customers are in cluster Low and 291 are in cluster High. This creates balanced clusters in terms of number of customers.

The regression coefficients and significant variables for each cluster are reported in Tables 2.1 and 2.2. Overall the variables *Past Purchases, Days since last log in, Reviews* and *Successful Referrer* are significant. Note that the signs of the coefficients are consistent: the farther in time a customer's last log in, the lower the probability to buy for that customer. Past Purchases, Reviews and Successful Referrer have a positive impact on the probability to buy. Also note that the two clusters do not have the same set of significant features. This illustrates the flexibility of the Logistic clustering algorithm that is able to capture the different behaviors of the population segments.

	Estimate	Std. Error	z value	$\mathbb{P}(>\|z\|)$
Intercept	-0.9135	0.3473	-2.63	0.0085**
Days since last log in	-0.0133	0.0065	-2.03	0.0428*
Reviews	0.0924	0.0509	1.81	0.0694*
Successful Referrer	1.2384	0.4115	3.01	0.0026**
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05				

Table 2.1: Regression results for cluster Low

Performance

There are several possible ways to evaluate the performance of a probabilistic classifier. We analyze the predictive power of our model looking at its confusion matrix and ROC curve.

	Estimate	Std. Error	z value	$\mathbb{P}(>\|z\|)$
Past Purchases	0.0017	0.0004	4.25	$< 10^{-4}$***
Days since last log in	-0.0290	0.0041	-7.07	$< 10^{-11}$***
Successful Referrer	0.7338	0.2494	2.94	0.0033**
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05				

Table 2.2: Regression results for cluster High

Confusion matrix A logistic model predicts a probability p_i to buy for every customer. We can transform it into a binary classifier (that predicts a binary value: whether the customer is going to make a purchase in the future or not) by choosing a threshold value γ and predicting that all the customers with $p_i > \gamma$ are buyers. In this case we choose a priori the same threshold value for the two clusters: $\gamma = 0.5$. This is validated for cluster High by the ROC curve plotted in Figure 2-6 ("top left corner" of the curve). Note that the hyper parameter γ can be tuned (and different values can be chosen for the two clusters) using cross-validation. We do not use this approach here to avoid overfitting the data. We can evaluate the predictive performance of a binary classifier looking at its confusion matrix and at some associated metrics.

Tables 2.3 and 2.4 represent the confusion matrix and the accuracy, specificity, sensitivity and precision for the two clusters. A detailed presentation of how a confusion matrix is built and definitions of its associated metrics can be found in Appendix A.3. In a confusion matrix, the rows represent the actual customer behavior and the columns represent the prediction. For example, in cluster Low there are 5 customers who did not make a purchase ($y_i = 0$) but for whom we predicted that they were buyers. Accuracy, specificity, sensitivity and precision are computed from the confusion matrix and measure the quality of the prediction conditioned on a column or a row of the table (see Appendix A.3). A perfect classifier achieves 100% score for each of these metrics.

We can see that cluster Low is unbalanced: 76% of its customers do not make a purchase. Recall that cluster Low is the set of less frequent customers with less than three Past Purchases, it makes sense that a low percentage of them makes a purchase in the future. We also have less information about these customers, and this makes the prediction more challenging. Our model correctly classifies most of the customers as non-buyers (156 out of 212) and has high scores for accuracy, specificity and precision. The only low score is the

sensitivity that captures the number of correctly classified customers among the buyers (here 13 out of 38+13). This is not surprising because among non frequent customers, the model probably does not have enough information to correctly identify all the buyers.

Cluster High is balanced: almost half of the customers make a purchase in the future. Cluster High is the cluster of frequent customers, the model has more information about the customers and is able to make more accurate predictions on customers' future behavior. We observe good scores for the four metrics we considered.

In summary, the in sample confusion matrix suggests a good overall performance of the algorithm. Further analysis is done by comparing the performance of our algorithm relative to two benchmarks (2.7.2) and analyzing the sensitivity of the parameters (2.7.3).

		Prediction	
		0	1
y_i	0	156	5
	1	38	13

Proportion of non-buyers	76%
Accuracy	80%
Specificity	97%
Sensitivity	25%
Precision	72%

Table 2.3: Confusion Matrix for cluster Low

		Prediction	
		0	1
y_i	0	129	26
	1	43	93

Proportion of non-buyers	53%
Accuracy	76%
Specificity	83%
Sensitivity	68%
Precision	78%

Table 2.4: Confusion Matrix for cluster High

AUC and ROC A logistic model predicts a probability for every customer and is not a simple binary classifier. A way to estimate the predictive performance of these probabilities that does not require the choice of a specific threshold value γ is the ROC curve (Receiver Operating Characteristic). The ROC curve represents, for different values of $\gamma \in (0, 1)$ the true positive and false positive rates. A random classifier (that predicts 1 with 50% probability) has a ROC curve aligned with the first diagonal. A perfect classifier has a false positive rate of 0 and a true positive rate of 1 (top left corner of the plot) for every value of

γ. Thus, looking at the ROC curve is a good way of evaluating the performance of the model for different values of $\gamma > 0$. A bad probabilistic classifier has a ROC curve close to the first diagonal, a good one gets close to the top left corner. A quantitative way of evaluating a ROC curve is computing the Area Under Curve (AUC), that geometrically is the area under the ROC curve and represents the probability that, given two customers one with $y_i = 1$ and the other $y_i = 0$, the model is able to correctly determine which one is which. Figures 2-5 and 2-6 represent the ROC curves for the two clusters. The AUC for cluster Low is 69% and 81% for cluster High. Again, we can notice a better predictive performance in cluster High, this is due to the sparsity of the data for non-frequent customers. Overall, the ROC curves and AUC scores show a good performance of the probabilistic classifier. Note that, on Figures 2-5 and 2-6 the curve are labeled by the corresponding values of γ. To build an effective binary classifier we need to choose the classifier with the highest pair (False Positive Rate, True Positive Rate). For both plots $\gamma = 0.5$ seems a good choice, and this validates the choice done in building the confusion matrix in the previous paragraph.

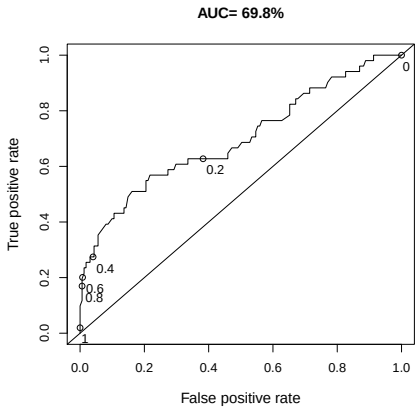


Figure 2-5: ROC curve for cluster Low

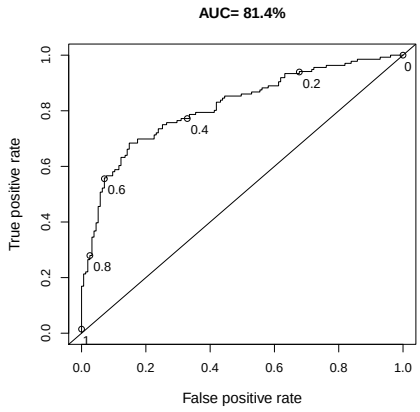


Figure 2-6: ROC curve for cluster High

2.7.2 Comparing to alternative approaches

In this Section, we compare the performance of our predictive model to a different benchmark. We want to evaluate the impact of using social features and of clustering customers on the predictive performance.

We define:

1. **Baseline**: a simple model that predicts the most frequent outcome for every customer (here as the majority of the customers do not buy in both clusters it predicts that nobody buys in the future)

2. **Benchmark**: joint clustering and logistic regression that *does not use social features* (only transaction history)

3. **Aggregated model**: a model that does not use clustering but predicts customers' probability to buy using a single logistic regression model (it can incorporate social features)

A detailed description of these models is presented in Appendix A.4.

We want to compare the out of sample accuracy of logistic clustering to the three benchmarks. In order to do that,we randomly split the data set of interested customers into a training set of 302 customers and a test set of 201 customers. In order to have a consistent baseline, we keep the percentage of buyers and the ratios between clusters Low and High constant in the the training and test sets. We train the four models on the training set and report their out of sample accuracy (computed on the test set) in Table 2.5. Notice that the accuracy of the Logistic Clustering (rounded to the nearest integer) does not change from the result presented in the confusion matrix. This suggests that Logistic Clustering is a robust model that can be consistently generalized to include new data points. We can see that Logistic Clustering significantly outperforms the Baseline and the Benchmark. On the entire data set, Logistic Clustering has correctly classified 15% more customers than the Baseline. Note that Logistic Clustering also outperforms the Aggregated model, but the difference between their performances is smaller. The intuition is that, with a clustering step, we are able to build a more accurate and robust model that captures the different aspects influencing the different segments of the population.

We can conclude that incorporating social features for demand prediction significantly increases the predictive performance. Furthemore, smartly clustering customers into categories creates a more robust and flexible model that is able to capture segments' specificity. Note that with a richer data set better performance can be achieved and the impact of clustering can become even more valuable.

Model	Cluster Low	Cluster High	Entire data set
Logistic clustering	**79%**	**77%**	**78%**
Baseline	76%	53%	63%
Benchmark	75%	64%	69%
Aggregated model	75%	74%	74%

Table 2.5: Out of sample accuracy of different models

2.7.3 Sensitivity Analysis

We have performed the analysis of the Logistic Clustering algorithm with a set of 503 customers because of data sparsity. This number of customers is not really large and it is important to check the sensitivity of our results with respect to the data. In Section 2.7.2, we divide the set of interested customers into a training and a test set and we compute the out of sample accuracy. In what follows, we use a different approach based on bootstrap to evaluate the sensitivity of the Logistic Clustering Parameters to the data.

Bootstrap approach

for $b = 1 \dots B$ bootstrap repetition

- Generate a bootstrap sample of size 503 with replacement from the set of interested customers
- run the Logistic Clustering algorithm
- store the value of the threshold $\hat{\Gamma}_b$ and the regression coefficients $\hat{\beta}_{1b}$ and $\hat{\beta}_{2b}$

This allows us to approximate the distribution of $\hat{\Gamma}$, $\hat{\beta}_1$ and $\hat{\beta}_2$ and their sensitivity to the data.

We simulate $B = 100$ bootstrap iterations and we reproduce our results in Figure 2-7. The top plots are box-plots of the logistic regression coefficients for significant features for the two clusters. In a box-plot, the thick horizontal line represents the median and the box represents the 25 and 75 quantiles. The Inter Quartile Range is defined as the distance between the 25 and the 75 quantiles. Outliers (points which distance from the median is

larger than 1.5 times the inter quartile range) are represented by a circle. We added a red horizontal line representing the value estimated with the original data set (from Tables 2.1 and 2.2). First of all, notice that the red line falls inside the box and in most of the cases is close to the median. This suggests that our model is robust to the data. Secondly, the signs of the coefficients of the significant variables are consistent. For example, for *Days since last log in* in cluster High, the estimated coefficient from Table 2.2 is negative and in the box-plot only points defined as outliers are positive. This is another indicator of the robustness of the model. Finally, note that, thanks to the bootstrap approach, we are able to estimate the distribution of the threshold $\hat{\Gamma}$. We can see that, in our simulations, $\hat{\Gamma}_b$ takes values between 2 and 7 with more than half of the values being between 3 and 4. This is another indicator of the robustness of our model.

To conclude, more data would have been extremely valuable for the performance of our model. Nevertheless, a sensitivity analysis based on bootstrap resampling shows that the estimated parameters are robust with respect to the data variability.

2.8 Conclusions

We considered a set of customers characterized by their transaction and social interactions history. The goal of this research was to define segments in the population using transparent and hierarchical rules in order to have a better demand estimation. In Section 2.4, we showed that the problem of joint clustering and logistic regression can be formulated as a strictly concave mixed-integer problem and can be solved efficiently by commercial software. In Section 2.5, the same approach is extended to the multinomial logit model where customers can choose between a set J of items to purchase. Finally, in Section 2.7, we apply the Logistic Clustering algorithm to SHOP.CA data. We analyze its performance and compare it to alternative predictive models. Our model has a 78% accuracy on the set of interested customers and significantly outperforms the classical models we use as benchmarks. We show that adding social features to the model significantly improves the predictive accuracy. For example, in this specific application, we show that the number of reviews written, the number of successful recommendations sent and the days since the last log in are good

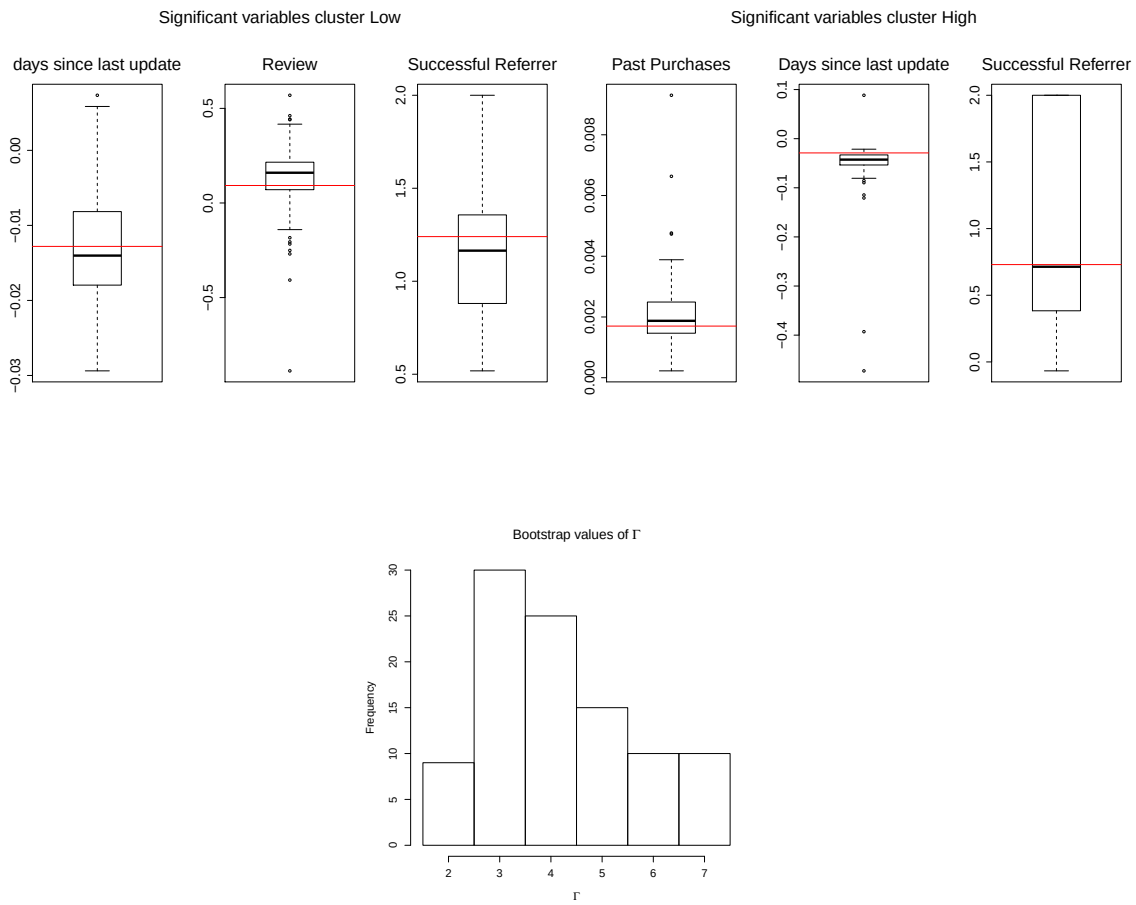


Figure 2-7: Bootstrap sensitivity results

predictors of the customer future purchase behavior. This suggests that incorporating social interactions in demand models can be extremely valuable for online retailers. In addition, when facing heterogeneous customers, splitting customers into segments is extremely valuable and allows to capture the differences in purchase behaviors across the population segments. In our application, we implemented Threshold Rules to define the clusters but this is not a constraint of the algorithm. Any polyhedral separator can be used. To conclude, even if more testing with different data sets is needed, we believe that the Logistic Clustering algorithm is a valuable and flexible tool to jointly find population segments and estimate their demand function with a logit choice model.

Chapter 3

Price sensitivity estimation with missing data

3.1 Introduction and Motivation

A key aspect in e-commerce is understanding how customers respond to discounts (that we will call rewards here). There are multiple possible sources of rewards: coupons and promo codes offered on social networks, targeted promotions for a short period of time (Cyber Monday for example), cash-back earned after a previous purchase... Not only rewards come from different sources, they can also be redeemed in different ways: they range between a fixed price discount on a specific category or item ($10 off a specific T-shirt or brand) to a percentage of discount applied to any purchase. These different vehicles for rewards have different impacts on customers. In traditional retailing, the promotions vehicles used simultaneously are often limited, it is possible to keep track of the rewards used and the number of customers that received them in order to evaluate the impact of each of these vehicles. In online retailing, with a large amount of items on sale and different promotion vehicles used simultaneously, it is impossible to keep track of which rewards were offered to which customers. In a given day, multiple promo codes with different values can be offered on social networks, and different promotions can be active on different items. In this context, it is impossible to know which customers are "aware" of which rewards. Nevertheless, when a customer makes a purchase, the retailer keeps track of the type of rewards used

and their amount. This situation can be described as a problem of missing data: for the customers that make a purchase, the retailer knows exactly what is amount of rewards they used and what was its source, but for non-buyers the retailer does not know what is the amount of discount they were offered. Understanding the impact of different types of rewards on different customers is key to build effective pricing strategies. The problem of missing or incomplete data has been widely studied in the Revenue Management literature. The classical example is an airline reservation website where all the transactions are recorded but there is no record of "the outside-option": the customers that visit the website, observe the prices and decide not to buy or to buy from a different carrier (see [9] or [13]).

In our specific problem, the problem is twofold. We first need to understand how rewards are allocated between customers. For example, if rewards are offered through social networks or friends' recommendations, social customers are more likely to receive these offers, while if they are offered through the retailer website, frequent customers will see them. After modeling the distribution of rewards, we need to understand how customers respond to these rewards by adding a price sensitivity component in the customer demand function.

Contributions: In a traditional parametric approach, the EM algorithm can be used to solve this incomplete data problem. The main contribution of this work is a generalization of the EM algorithm that allows a non-parametric estimation of the distribution of rewards. This approach, denoted by NPM ("Non Parametric Maximization"), is more flexible and robust and can be applied without restrictive hypothesis on the shape of the distribution of rewards. With a logistic demand function, we show that the NPM is a consistent estimator of the price sensitivity and distribution of rewards. Furthermore, with extensive simulations, we show that the NPM converges significantly faster (3 times faster on average) than the EM algorithm.

3.2 Model description

Consider a retailer selling a product online. He faces a set of customers where each customer is defined by X_i a set of observable (by the retailer) characteristics: (past purchases, social

activities, social interactions, friends etc.). Furthermore, each customer is assigned to a level of reward (that corresponds to a percentage of discount). These rewards can come from multiple sources: cashback from previous purchases, promotions in a special period of time, coupons or promo codes promoted through social media etc. We assume that active consumers (in terms of social networks and past purchases) are more likely to receive high level of rewards. Customers that are very active on social networks are more likely to be aware of special promotions or promo codes posted by the retailer on these platforms. Other customers may have cashback earned on previous purchases. Mathematically, we assume that every customer's rewards level, denoted by R_i, is random and lies on a finite and discrete set ($\{0\%,1\%,5\%,10\%\}$ for example). We also assume that the distribution of R_i depends on the customers' characteristics.

3.2.1 Model

We assume that the decision process of customer i is in two steps:

1. The customer is assigned to a random reward level R_i. We assume that R_i lies in a discrete and finite set $\mathcal{R}$ and that distribution of R_i depends on a vector of customer features denoted X_i . We don't assume a specific form for this distribution and we denote $\mathbb{P}(R_i = r|X_i) = f_{X_i}(r)$.

2. After observing his reward level R_i, the customer decides whether to make a purchase according to a *logistic model*.

3.2.2 Estimation Problem

Let us assume that the retailer wants to estimate customers' purchase behavior using only transaction data. Given a dataset $\mathcal{D}$, this means that for every customer i we observe his vector of characteristics and whether he decides to make a purchase. For the customers that made a purchase, we also observe the level of rewards they received R_i. For the customers that did not make a purchase we do not observe R_i. This is thus an estimation problem with missing data.

Define

- i: index of a customer
- N: total number of customers (data)
- X_i: vector of features of customer i (data)
- R_i: reward offered to customer i (data only for buyers)
- y_i: binary variable that indicates whether the customer makes a purchase (data)
- $\mathcal{D}_b$: subset of buyers (subset of data for which we have complete information)

Let us also assume that R lies in a discrete set $R_i \in \mathcal{R} = \{r_1, \ldots, r_K\}$. Let us denote :

$$\mathbb{P}(R_i = r|X_i) = f_{X_i}(r)$$

and

$$\mathbb{P}(y_i = 1|R_i, X_i) = \frac{e^{\beta.X_i + \beta_r R_i}}{1 + e^{\beta.X_i + \beta_r R_i}}$$

Assumption 2. *We assume that R_i and X_i are independent.*

We introduce this assumption to avoid colinearity between the logistic regression features (X_i, R_i). A sufficient condition is that X_i is a set of independent features that can be split into two disjoint subsets X_i^1,X_i^2 where f depends only on X^1 and $\mathbb{P}(y_i = 1|R_i, X_i)$ depends only on X^2. This condition is, of course, not necessary but avoids colinearity in the regressors and simplifies the analysis. In the rest of this chapter, we will assume that assumption 2 holds.

We want to estimate the distribution function f and the parameters of the logistic regression(β, β_r) using only transaction data, this is an estimation problem with missing data.

We will use the following definition of consistency:

Definition 2.1. *Let N be the number of customers. $\hat{\theta}_N$ is a consistent estimator of θ_0 if $\hat{\theta}_N$ converges to θ_0 in probability:*

$$\hat{\theta}_N \xrightarrow[N\to\infty]{\mathbb{P}} \theta_0$$

We will need to compute local averages of functions, we will use the following notation to represent the local average of the function f within a neighborhood of x with size ϵ.

$$\hat{\mathbb{E}}_{X_i \simeq x}(f) \stackrel{d}{=} \frac{\sum_i \mathbf{1}_{|X_i - x| < \epsilon} f(X_i)}{\sum_i \mathbf{1}_{|X_i - x| < \epsilon}}$$

In the following, we will not explicitly denote the value of ϵ and keep in mind that it is a parameter that has to be tuned.

3.3 Motivation: missing data approach

In this Section, we build the likelihoods of complete and incomplete data sets.

3.3.1 Complete data likelihood

If we consider a complete data set, i.e. where the retailer observes the rewards for every customer and whether they make a purchase or not, the likelihood for a data point (x, r, y) can be decomposed in two steps:

- If a customer has a features' vector $X_i = x$ then the corresponding probability of receiving the reward level r is $\mathbb{P}(R_i = r | X_i = x) = f_x(r)$

- If a customer has a features' vector $X_i = x$ and received a reward level $R_i = r$ then the corresponding purchase probability is given by $\mathbb{P}(y_i = 1 | X_i = x, R_i = r) = \frac{e^{\beta.x + \beta_r r}}{1 + e^{\beta.x + \beta_r r}}$ and $\mathbb{P}(y_i = 0 | X_i = x, R_i = r) = \frac{1}{1 + e^{\beta.x + \beta_r r}}$.

- Finally, the likelihood of a data point (x, r, y) is

$$y f_x(r) \frac{e^{\beta.x + \beta_r r}}{1 + e^{\beta.x + \beta_r r}} + (1 - y) f_x(r) \frac{1}{1 + e^{\beta.x + \beta_r r}}$$

  As $y \in \{0, 1\}$ the first term of the sum represents a buyer and the second term represents a non-buyer.

Thus, the complete data log-likelihood becomes

$$\begin{aligned}\mathcal{L}_c(\mathcal{D},\beta,f) =& \sum_{i=1}^{N} y_i \log\left(f_{X_i}(R_i)\frac{e^{\beta.X_i+\beta_r R_i}}{1+e^{\beta.X_i+\beta_r R_i}}\right)\\ &+(1-y_i)\log\left(f_{X_i}(R_i)\frac{1}{1+e^{\beta.X_i+\beta_r R_i}}\right)\end{aligned} \tag{3.1}$$

3.3.2 Incomplete data likelihood

When the rewards offered to non-buyers are missing, the likelihood for buyers can be built using the previous approach, while for non-buyers the expression is slightly different.

- For buyers, the data is complete thus the likelihood of a data point $(x, r, y = 1)$ is $f_x(r)\frac{e^{\beta.x+\beta_r r}}{1+e^{\beta.x+\beta_r r}}$.

- For non-buyers, we do not observe the reward level R_i and the likelihood of a data point $(x, \varnothing, y = 0)$ is given by the law of iterated expectations:

$$\mathbb{P}(y=0|x) = \mathbb{E}\left[\mathbb{P}(y=0|x,R)\right]$$

  and the likelihood for a non-buyer becomes:

$$\log\left(\sum_{r\in\mathcal{R}} f_{X_i}(r)\frac{1}{1+e^{\beta.X_i+\beta_r r}}\right)$$

The incomplete data log-likelihood is:

$$\begin{aligned}\mathcal{L}_i(\mathcal{D},\beta,f) =& \sum_{i=1}^{N} y_i \log\left(f_{X_i}(R_i)\frac{e^{\beta.X_i+\beta_r R_i}}{1+e^{\beta.X_i+\beta_r R_i}}\right)\\ &+(1-y_i)\left(\log\left(\sum_{r\in\mathcal{R}} f_{X_i}(r)\frac{1}{1+e^{\beta.X_i+\beta_r r}}\right)\right)\end{aligned} \tag{3.2}$$

To illustrate the expression of this likelihood, let us consider a simple example where we assume a parametric shape for f: $f_{X_i,\alpha}$ where α is a set of parameters. Let us assume that we are just interested in whether a customer receives a reward or not: $\mathcal{R} = \{0, 1\}$. Let us also assume that, the probability of receiving a reward is a linear function of the features of the customer:

$$\mathbb{P}(R_i = 1|X_i) = \alpha.X_i$$

where α is a vector. Then we have

$$f_x(1) = \alpha.x \text{ and } f_x(0) = 1 - \alpha.x$$

In this setting, the incomplete data likelihood $\mathcal{L}_i$ can be written in terms of the data and the pair (β, α) and we can estimate these parameters by maximizing the incomplete data log-likelihood with respect to (β, α). Unfortunately, with a large number of customers and a multidimensional vector of features, this approach can be numerically intensive. In addition, we can show that, in general, $\mathcal{L}_i$ is neither concave nor quasi-concave.

Proposition 2.1. *$\mathcal{L}_i(\mathcal{D}, \beta, f)$ is not quasi-concave.*

Proof. Consider the simple case where there are no customer features and $\mathcal{R} = \{0, 1\}$. With binary reward levels, $R_i = 0$ means that the customer is not offered a reward and $R_i = 1$ means that the customer is offered a reward. In this case all the customers are the same and they all have the same probability of receiving rewards level 1. Then let $\alpha = \mathbb{P}(R = 1)$ and $1 - \alpha = \mathbb{P}(R = 0)$ (this does not depend on i).

Using the second term of equation 3.2, the log-likelihood for non-buyers becomes:

$$l(\alpha, \beta_r) = \log\left(\alpha \frac{1}{1 + e^{\beta_r}} + \frac{1 - \alpha}{2}\right)$$

This is not a concave function of (α, β_r). In fact if $(\alpha_1, \beta_{r1}) = (0, 1)$ and $(\alpha_2, \beta_{r2}) = (1, 0)$, let (α_3, β_{r3}) be an intermediate point: $(\alpha_3, \beta_{r3}) = (\frac{\alpha_1 + \alpha_2}{2}, \frac{\beta_{r1} + \beta_{r2}}{2}) = (.5, .5)$. Then we have $l(\alpha_1, \beta_{r1}) = l(\alpha_2, \beta_{r2}) = -\log(2) \simeq -0.69$ and $l(\frac{\alpha_1 + \alpha_2}{1}, \frac{\beta_{r1} + \beta_{r2}}{2}) = -0.8237792$, which is smaller than the previous two. Therefore, concavity is violated.

In the same setting without features, it is possible to find numerical examples with more than one customer where the likelihood is not quasi-concave. Consider 8 customers with 7 non-buyers and one buyer that does not receive a reward. Then $\mathcal{L}_i$ becomes:

$$l(\alpha, \beta_r) = \log(\frac{1 - \alpha}{2}) + 7 \log\left(\alpha \frac{1}{1 + e^{\beta_r}} + \frac{1 - \alpha}{2}\right)$$

If you consider a parameter $x \in [0, 1]$ and the pair $(\alpha, \beta_r) = (x, 1 - x)$, then the slice of the likelihood function defined by $x \to l(x, 1 - x)$ is not concave as shown in Figure 3-3.

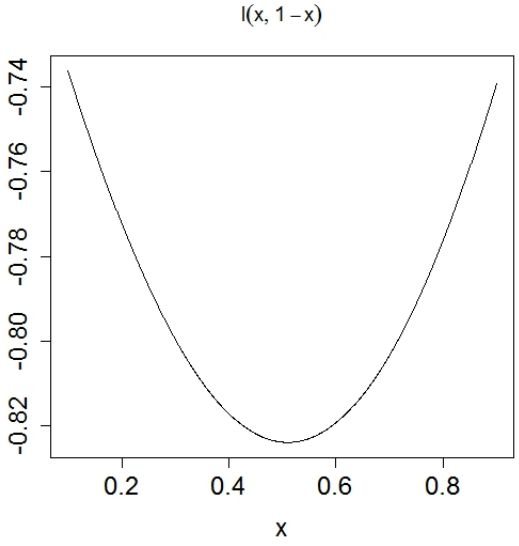


Figure 3-1: Non concavity of the incomplete data likelihood: $l(x, 1-x)$

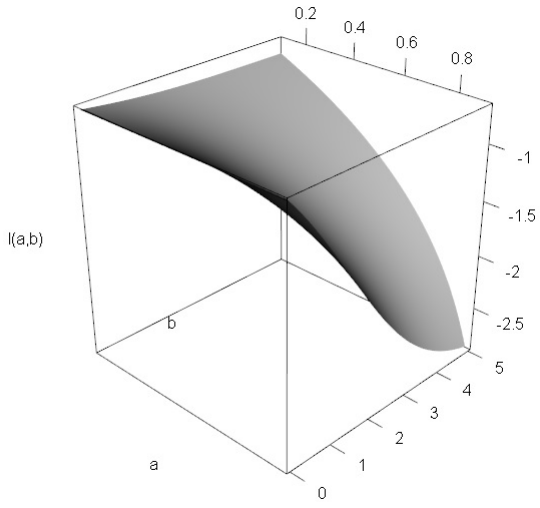


Figure 3-2: Surface of the likelihood for non-buyers

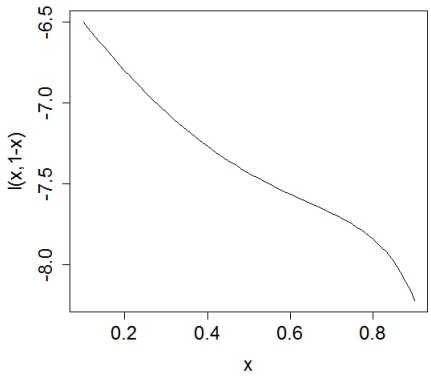


Figure 3-3: Non concave slice of likelihood

□

3.4 EM algorithm

The Expectation-Maximization (EM) algorithm is a classical approach to parameter estimation with missing data introduced by Dempster et al. in 1977 ([8]). Consider a data set $\mathcal{D}_c$ and assume that some data is missing and we can only observe a censored data set $\mathcal{D}$. In

our case, $\mathcal{D}_c$ is a complete data set where we observe the rewards for every customer and $\mathcal{D}$ is the incomplete data set where the rewards for non-buyers are missing. The EM algorithm turns out to be particularly useful in the case where the log-likelihood of the incomplete data is hard to maximize (it is not quasi-concavity and has multiple local optima...) while the complete data log-likelihood has a simple form. This is the case in our setting, we have shown that the incomplete data likelihood is not quasi-concave in general and it is easy to prove that the complete data likelihood is concave. The EM algorithm has been used in the Operations Management literature to take into account censored or unobserved data in several applications. In [9], Vulcano et al. estimate the customers' arrival process on an airline booking website, Phillips et al. ([10]) use the EM algorithm to estimate unobserved reserve prices and willingness-to-pay in a business to business negotiation, Jagabathula et al. ([11]) focuses on customers' willingness-to-pay in a revenue management setting. The EM algorithm is also widely used in Machine Learning for clustering problems as it provides a fast and robust approach to estimate mixture of Gaussians.

Let θ be the set of parameters to estimate. Consider the complete and incomplete data sets $\mathcal{D}_c$ and $\mathcal{D}$ and their associated log-likelihoods $\mathcal{L}_c(\mathcal{D}_c, \theta)$ and $\mathcal{L}_i(\mathcal{D}, \theta)$. Assume that $\mathcal{L}_i$ is hard to maximize while $\mathcal{L}_c$ has a simple form. The idea of the EM algorithm is to replace the complicated incomplete data likelihood $\mathcal{L}_i$ by the expectation of the complete data log-likelihood. The EM algorithm starts with an initial value for the parameters θ^0. For every iteration k, in the Expectation step "E", it computes the expectation of the complete data log-likelihood given θ^k: $\mathbb{E}_{\theta^k}\left(\mathcal{L}_c(\mathcal{D}_i, \theta)\right)$. This expected log-likelihood has the same simple form as $\mathcal{L}_c$ and can thus be directly maximized to find updated parameters θ^{k+1} in the Maximization step "M".

In summary, the steps of the EM algorithm are:

- **Initialization:** Set $\theta = \theta^0$

  On the k^{th} iteration:

- **E step:** compute $e(\theta^k, \theta) = \mathbb{E}_{\theta^k}\left[\mathcal{L}_c(\mathcal{D}, \theta)\right]$

- **M step:** update θ by setting

$$\theta^{k+1} = \text{argmax}_\theta \quad e(\theta^k, \theta)$$

These steps are repeated until convergence of (θ^k).

The EM algorithm is a simple and fast way of building an estimator of θ. Dempster ([8]) proves that, under some regularity conditions on $\mathcal{L}_i$, $\mathcal{L}_i$ increases after each iteration of the EM algorithm. If the incomplete log-likelihood function is continuous in θ then all limit points of the EM algorithm are stationary points of the incomplete log-likelihood function ([15]). Thus, if the EM algorithm converges, it converges to a stationary point. By starting with different values θ^0, running the EM algorithm until convergence (if any), and finally considering the estimation of θ with the highest likelihood value, we obtain a maximum likelihood estimator of θ, that is consistent and efficient. In summary, the EM algorithm is a simple and robust way of estimating parameters with missing or unobserved data. Nevertheless, in some applications, it has been criticized for requiring a large number of iterations before reaching convergence of the parameters ([13]). Having to run the algorithm from different starting points to avoid reaching local optima of the likelihood function may not be a practical approach.

In our setting, the EM algorithm can be applied to estimate jointly β,β_r and f if we assume that f has a parametric shape. Let α be the parameters of f and $\theta = (\beta, \beta_r, \alpha)$. Then

$$\mathcal{L}_i(\mathcal{D}, \theta) = \sum_{i=1}^{N} y_i \log\left(f_{X_i,\alpha}(R_i) \frac{e^{\beta.X_i + \beta_r R_i}}{1 + e^{\beta.X_i + \beta_r R_i}} \right) + (1 - y_i) \left(\log\left(\sum_{r \in \mathcal{R}} f_{X_i,\alpha}(r) \frac{1}{1 + e^{\beta.X_i + \beta_r r}} \right) \right)$$

and

$$\begin{aligned} e(\theta^k, \theta) = \mathbb{E}_{\theta^k} \mathcal{L}_c(\mathcal{D}, \theta) \quad &= \quad \sum_{i=1}^{N} y_i \log\left(f_{X_i,\alpha}(R_i) \frac{e^{\beta.X_i + \beta_r R_i}}{1 + e^{\beta.X_i + \beta_r R_i}} \right) \\ &\quad + (1 - y_i) \mathbb{E}_{\theta^k} \left[\log\left(f_{X_i,\alpha}(R_i) \frac{1}{1 + e^{\beta.X_i + \beta_r R_i}} \right) \right] \end{aligned}$$

where the expectation is taken over the unobserved variable R_i for non-buyers. Note that e and $\mathcal{L}_i$ are identical for buyers ($y_i = 1$) as there is no missing data for these customers but differ for non-buyers ($y_i = 0$). In e, there is no sum over $r \in \mathcal{R}$ inside the logarithm, and this is the reason why maximizing e is a concave optimization problem while maximizing $\mathcal{L}_i$ may not be one.

In order to compute the expectation $\mathbb{E}_{\theta^k}$ we need to compute $\mathbb{P}(R_i = r|y_i = 0, X_i, \theta^k)$. This can be done using Bayes' rule:

$$\begin{aligned} p^k_{r,i} = \mathbb{P}(R_i = r|y_i = 0, X_i, \theta^k) &= \frac{\mathbb{P}(y_i = 0|R_i = r, X_i)\mathbb{P}(R_i = r|X_i)}{\mathbb{P}(y_i = 0|X_i)} \\ &= \frac{\frac{1}{1+e^{\beta^k.X_i+\beta^k_r r}} f_{X_i,\alpha^k}(r)}{\sum_{s\in\mathcal{R}} f_{X_i,\alpha^k}(s)\frac{1}{1+e^{\beta^k.X_i+\beta^k_r s}}} \end{aligned}$$

Overall the expected log likelihood becomes:

$$\begin{aligned} e(\theta^k, \theta) = &\sum_i y_i \left[\log\left(f_{X_i,\alpha}(R_i)\right) + \beta.X_i + \beta_r R_i - \log(1 + e^{\beta.X_i+\beta_r R_i})\right] \\ &+ \sum_i (1 - y_i) \sum_{r\in\mathcal{R}} \left[p^k_{r,i} \log\left(f_{X_i,\alpha}(r)\right) - p^k_{r,i} \log(1 + e^{\beta.X_i+\beta_r r})\right] \end{aligned}$$

We implemented the EM algorithm with simulated data in a parametric setting. The results can be found in Section 3.6.

3.5 NPM algorithm

The EM algorithm is a classical approach used for parameter estimation with missing data. For our application, if the function f is parametric, it is a fast and efficient method for jointly estimating f and β. There are theoretical guarantees for the convergence of the estimator under regularity conditions of the incomplete data likelihood function. We present here an alternative approach that does not require any parametric assumption on f.

Recall that $f_{X_i}(r)$ represents the probability that a customer with a feature vector X_i receives the reward level r. This model assumes that the distribution of rewards depends on the customer's characteristics because more active and social customers are more likely

to be aware of promotions and promo codes. While this dependency is intuitive, assuming a specific parametric representation for f seems to be a strong hypothesis and can be difficult to justify. When modeling customer behavior, the risk of mispecification is high and a non-parametric analysis is extremely appealing due to its flexibility and robustness.

The Non Parametric Maximization ("NPM") algorithm extends the EM algorithm approach to the case where f is non-parametric. As in the EM algorithm, an iterative approach is used and the "difficult" maximization part of the incomplete data log-likelihood is replaced by a simpler maximization problem. The intuition behind this algorithm is that, even though the incomplete data likelihood is not guaranteed to be concave in (f, β, β_r), it is concave in (β, β_r) when f is fixed. Thus, if we have an estimation of f, we can recover an estimation of β by maximizing the incomplete data likelihood only with respect to (β, β_r) which is a simple concave optimization problem.

3.5.1 Description

In what follows we will denote as β the vector (β, β_r). The NPM algorithm starts with initial estimates of the coefficients $\beta^{(0)}$. Without prior knowledge of the impact of the features, a good starting point is $\beta^{(0)} = 0$. This sets the probability of making a purchase to 50% for every customer (An intuitive explanation of why $\beta^{(0)} = 0$ is an appropriate starting point is presented later). It then uses this initial estimate to build a non-parametric estimator $f^{(0)}$ of the function f. This is the Non Parametric "NP" step. Then, in the Maximization step "M", it replaces f by this first estimate $f^{(0)}$ in the incomplete data log-likelihood and maximizes the latter with respect to β to update its estimation $\beta^{(1)}$. The process is then repeated until convergence of (f, β).

In summary, the steps of the NPM algorithm are:

- **Initialization:** Set $\beta = \beta^{(0)}$ ($\beta^{(0)} = 0$ is a good initial guess)
  On the k^{th} iteration:

- **NP step:** $f^{(k)}$: non parametric estimation of f knowing $\beta^{(k)}$

- **M step:** replace f by $f^{(k)}$ in $\mathcal{L}_i$ and set

$$\beta^{(k+1)} = \text{argmax}_\beta \mathcal{L}_i(\mathcal{D}, \beta, f^{(k)})$$

- The process is repeated until convergence of (f, β).

"NP" step: Non Parametric Estimation of the function f

In the NP step, we use the data and an estimation of β to build a non-parametric estimation of the function f. Recall that R_i lies in a discrete set $\mathcal{R}$. For the sake of simplicity in the explanation, we will consider, without loss of generality, the case where the rewards are binary: $\mathcal{R} = \{0, 1\}$. This means that we are just interested in whether a customer receives a reward or not. We do not consider the possibility of having a range of rewards (0,5%,10%...). Nevertheless, the same approach can be similarly used for a general discrete and finite set $\mathcal{R}$. The description of the NPM algorithm with a general set $\mathcal{R}$ can be found in Appendix B.1.

If the rewards are binary, for every customer we are interested only in the quantity $f_{X_i}(1) = \mathbb{P}(R_i = 1|X_i)$ because we can get $f_{X_i}(0)$ from $f_{X_i}(0) = 1 - f_{X_i}(1)$.

Naive approach A naive approach to build an estimator of f uses only the available data, i.e. the rewards received by buyers. Let us consider the subset of buyers $\mathcal{D}_b$. For this subset of customers we observe the rewards they receive. We can thus build a non-parametric estimator of $\mathbb{P}(R_i = 1|i \in \mathcal{D}_b, X_i) = \mathbb{P}(R_i = 1|y_i = 1, X_i)$ using $\mathcal{D}_b$. With a moving average estimator for example, we can approximate $\mathbb{P}(R_i = 1|i \in \mathcal{D}_b, X_i = x)$ by the proportion of buyers that received a reward in the neighborhood of x.

Filling missing data using averages of observable data is a common practice in marketing. In the econometrics literature, [12] focuses on supermarket sales data, where customers can choose between different brands but only the price of the purchased option is recorded. They show that there is a systematic bias in filling the missing price of a brand by the average price of observed transactions of this brand. If customers are price sensitive, they tend to purchase cheap options. Looking only at transaction data for a product means conditioning

on the fact that this product has been purchased, thus that it was cheap. Averaging over observed prices leads to a systematic underestimation of the price.

We observe the same phenomenon in our application because we assume that rewards have a positive impact on the customer purchase behavior ($\beta_r > 0$). Intuitively, if a customer receives a reward he is more likely to make a purchase. Thus, among buyers, there is a higher proportion of customers that received a reward compared to the whole population. By computing the proportion of customers that receive a reward among buyers we overestimate the probability $f(1)$. It is not possible to use only buyers data to estimate the distribution of rewards.

Proposition 2.2. *The naive approach leads to an overestimation of the probability of receiving a reward:*

$$\mathbb{P}(R_i = 1|y_i = 1, X_i) > \mathbb{P}(R_i = 1|X_i) \text{ for any feature vector } X_i$$

Proof. We assume that $\beta_r > 0$, thus

$\mathbb{P}(y_i = 1|R_i = 1, X_i) = \frac{e^{\beta.X_i+\beta_r}}{1+e^{\beta.X_i+\beta_r}} > \mathbb{P}(y_i = 1|R_i = 0, X_i) = \frac{e^{\beta.X_i}}{1+e^{\beta.X_i}}$, since $e^{-\beta_r} < 1$.

Therefore,

$$\begin{aligned}\mathbb{P}(y_i = 1|R_i = 1, X_i) >& \mathbb{P}(y_i = 1|X_i) \\ =& \mathbb{P}(y_i = 1|R_i = 1, X_i) \times \mathbb{P}(R_i = 1|X_i) \\ &+ \mathbb{P}(y_i = 1|R_i = 0, X_i) \times (1 - \mathbb{P}(R_i = 1|X_i))\end{aligned}$$

Finally, using Bayes' rule:

$$\begin{aligned}\mathbb{P}(R_i = 1|X_i, y_i = 1) =& \frac{\mathbb{P}(y_i = 1|R_i = 1, X_i)}{\mathbb{P}(y_i = 1|X_i)} \mathbb{P}(R_i = 1|X_i) \\ >& \mathbb{P}(R_i = 1|X_i)\end{aligned} \tag{3.3}$$

Thus this naive approach leads to an overestimation of the distribution of rewards. □

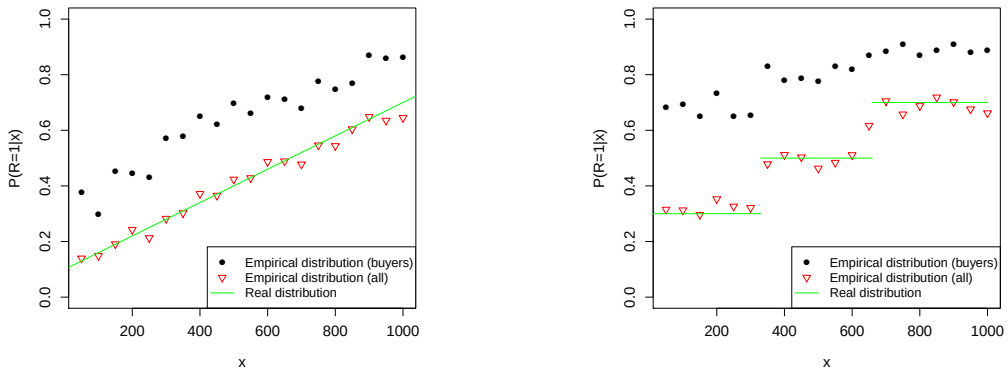


Figure 3-4: Naive estimation and true distribution for different shapes of f

In Figure 3-4 we represent two shapes of function f (linear and piece wise constant) in a one dimensional case. Here customers are characterized by a one dimension vector X_i that is represented on the x-axis. (The range of x is taken to represent the variable "Past Purchases" introduced in Section 3.6). To every value of x is associated a probability of receiving a reward represented by the green line in Figure 3-4. The red dots represent the empirical distribution of rewards on a data set of 10000 customers (buyers and non-buyers), the black dots represent the estimation given by the naive approach (empirical distribution among buyers). We can see that in both plots the naive approach significantly overestimate the function f for all values of the feature vector x.

NPM approach With the example of the naive approach, we have seen that is not possible to recover an unbiased estimation using only data from buyers. Nevertheless, it is possible to use the data $\mathcal{D}$ **and** an estimation of β to build a non parametric estimator of f. In the "NP" step, we combine the non-parametric estimation given by the naive approach with Bayes' rule. Doing so, we are able to correct for the systematic bias of the naive approach and recover an estimation of f from the data and an estimation of β.

Let us start with Bayes' rule:

$$\mathbb{P}(R_i = 1|X_i, y_i = 1) = \frac{\mathbb{P}(y_i = 1|R_i = 1, X_i)}{\mathbb{P}(y_i = 1|X_i)}\mathbb{P}(R_i = 1|X_i) \tag{3.4}$$

In the previous equation, the left-hand side can be estimated from the data using the

naive approach. We will show that the right hand-side can be written as a simple function (that is denoted g) of the data, β and f. Thus, by inverting this function g, we can recover a non-parametric estimation of f.

First let us remark that:

1. $\mathbb{P}(R_i = 1|X_i) = f_{X_i}(1)$ is the quantity we want to estimate
2. We can build a non parametric estimator of $\mathbb{P}(R_i = 1|X_i, y_i = 1)$ using the subset of buyers $\mathcal{D}_b$ (see naive approach)
3. If we know the value of β then $\mathbb{P}(y_i = 1|R_i, X_i) = \frac{e^{\beta.X_i + \beta_r R_i}}{1+e^{\beta.X_i+\beta_r R_i}}$.
4.

$$\begin{aligned}\mathbb{P}(y_i = 1) &= \mathbb{P}(R_i = 1|X_i)\mathbb{P}(y_i = 1|R_i = 1, X_i) + (1 - \mathbb{P}(R_i = 1|X_i))\mathbb{P}(y_i = 1|R_i = 0, X_i) \\ &= f_{X_i}(1)\mathbb{P}(y_i = 1|R_i = 1, X_i) + (1 - f_{X_i}(1))\mathbb{P}(y_i = 1|R_i = 0, X_i)\end{aligned}$$

Let us combine the previous remarks. Using remark 3 we can build a non-parametric estimator of $\mathbb{P}(y_i = 1|R_i, X_i = x, \beta)$ with a kernel smoothing. For example with a window kernel we can approximate it by

$$\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r r}}{1 + e^{\beta X_i + \beta_r r}}\right)$$

Finally, the right hand side of equation (3.4) becomes

$$\frac{f_x(1)\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i+\beta_r}}\right)}{f_x(1)\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i+\beta_r}}\right) + (1 - f_x(1))\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)}$$

Let g be:

$$g(\mathcal{D}, \beta, x, \hat{f}) = \frac{\hat{f}\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i+\beta_r}}\right)}{\hat{f}\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i+\beta_r}}\right) + (1 - \hat{f})\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)} \tag{3.5}$$

Notice that, given $\mathcal{D}, \beta$ and x, g is a homography with respect to $\hat{f}$ (ratio of two affine

functions) and thus can be inverted easily in closed form. We can then build a non parametric estimator of $f_{X_i=x}(1)$ solving the equation with respect to $\hat{f}$:

$$\hat{\mathbb{P}}(R_i = 1|x, y_i = 1) = g(\mathcal{D}, \hat{\beta}, x, \hat{f})$$

where the left hand side is a non parametric estimator of $\mathbb{P}(R_i = 1|X_i = x, y_i = 1)$. This gives:

$$\hat{f}_x(1) = -\frac{\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)\hat{\mathbb{P}}(R_i = 1|x, y_i = 1)}{\left[\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right) - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)\right]\hat{\mathbb{P}}(R_i = 1|x, y_i = 1) - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right)}$$

Choice of a starting value for β: The NPM algorithm starts with an initial estimate of (β, β_r). Without any prior knowledge of the values of (β, β_r), $(\beta^{(0)}, \beta_r^{(0)}) = (0, 0)$ is a good starting point. This can be seen from the equation above that defines the update of $\hat{f}$ done in the NP step. Notice that, replacing (β, β_r) by $(0, 0)$ in the equation above, the expression is simplified and we get $\hat{f}_x(1)^{(0)} = \hat{\mathbb{P}}(R_i = 1|x, y_i = 1)$, which corresponds to the estimator given by the naive approach. Recall that the naive approach is the best approximation of f, without any a knowledge of the values of (β, β_r). Thus, this is a good starting point for the iterative algorithm.

M step: Estimation of the logistic regression coefficients

In the previous Subsection, we have shown how, using an estimation of the logistic coefficients β we can build a non parametric estimator of the function f. This is the "NP" step of the NPM algorithm. In this paragraph, we focus on the "M" step: using an estimator $\hat{f}$ of f, we build an estimator of the logistic regression coefficients β.

Assume you have access to an estimator $\hat{f}$ of f. Recall that the incomplete data log-likelihood is given in (3.2):

$$\mathcal{L}_i(\mathcal{D}, \beta, f) = \sum_{i=1}^{N} y_i \log\left(f_{X_i}(R_i)\frac{e^{\beta.X_i+\beta_r R_i}}{1+e^{\beta.X_i+\beta_r R_i}}\right) + (1-y_i)\left(\log(\sum_{r\in\mathcal{R}} f_{X_i}(r)\frac{1}{1+e^{\beta.X_i+\beta_r r}})\right)$$

Proposition 2.3. *Given $\mathcal{D}$ and f, $\mathcal{L}_i(\mathcal{D}, \beta, f)$ is a quasi-concave function of β.*

Proof. For one customer i, the log-likelihood is

$$y_i \left(\log \left(f_{X_i}(R_i) \right) + \beta . X_i + \beta_r R_i - \log \left(1 + e^{\beta . X_i + \beta_r R_i} \right) \right) + (1 - y_i) \left(\log(\sum_{r \in \mathcal{R}} f_{X_i}(r) \frac{1}{1 + e^{\beta . X_i + \beta_r r}}) \right)$$

- $\beta . X_i + \beta_r R_i$ is a linear function of β
- $-\log(1 + e^{\beta . X_i + \beta_r R_i})$ is a concave function of β
- $\log(\sum_{r \in \mathcal{R}} f_{X_i}(r) \frac{1}{1+e^{\beta . X_i + \beta_r r}})$ is quasi-concave in β as it is the logarithm of a sum of sigmoids

Thus, given $\mathcal{D}$ and f, the incomplete data log-likelihood is a quasi-concave function of β. □

The "M" step of the NPM algorithm is the following:
Assuming a previous estimation $\hat{f}$ of f, we build an estimator of β by maximizing the incomplete data log-likelihood with respect to β:

$$\hat{\beta} = \text{argmax}_{\beta} \mathcal{L}_i(\mathcal{D}, \beta, \hat{f})$$

3.5.2 Advantages

The NPM algorithm is an alternative to the EM algorithm that can be applied without any parametric assumption on the distribution function f. The NPM algorithm is thus a semi-parametric approach: the distribution of rewards f is estimated in a non-parametric way while the purchase behavior is assumed to follow a (parametric) logistic shape. There are several advantages to this approach.

A non parametric approach

Non-parametric approaches allow for flexibility in data estimation and do not rely on heavy distributional assumptions. This is particularly valuable when modeling customers' or retailers' behavior tend not to follow classical linear or normal distributions. It has been observed

([14]) that, when modeling human decision processes, non-parametric approaches are extremely valuable because the observed behavior does not verify the assumptions underlying classical parametric models. In our specific application, the function $f_X(.)$ represents the distribution of rewards received by a customer with features vector X. While we can assume that this distribution depends on the customer's characteristics and the probability of having high rewards is higher for "active" customers, it is difficult to justify a specific parametric shape for this relationship (linear, quadratic or piecewise constant for example). In this setting, the flexibility of non-parametric estimation allows to capture complex dependency between the different variables.

On the other hand, we have to note that non-parametric approaches are criticized because of the so-called "curse of dimensionality". They are extremely powerful and allow to estimate a more accurate relationship between the variables when dealing with a small (and small often means one) number of dependent variables, but become computationally intractable when the number of dependent variables increase. An intuitive understanding of the curse of dimensionality comes from the sparsity of the data in high dimensional spaces. For example, in the NP step of the NPM algorithm, we need to compute local averages in equation (3.5). If the dimension of the features vector X increases, more and more data points are needed to be able to find enough observations to perform meaningful local averages. Intuitively, if n data points are required for a non-parametric estimation in a 1-dimensional space, n^d data points are required to have a comparable training data density in a d-dimensional space. For more details on non-parametric estimation see for example [16].

Another drawback of non-parametric estimation, which is closely related to the curse of dimensionality, is that non-parametric approaches can tend to overfit the data and may not be able to generalize to other datasets. There is a tradeoff between localization (ability to capture all the information in the training set) and generalization (to other data sets). This is captured by the tuning parameters in the non-parametric approach, in the NPM algorithm it lies in the non-parametric smoothing technique used (local average, kernel density estimation, k-nearest neighbors, etc). Several parameter tuning techniques exist to avoid overfitting.

Computational advantages

The specific approach of the NPM algorithm creates interesting computational advantages. First of all, it decouples the estimation of the function f (done in the NP step) from the estimation of the logistic regression coefficients β (done in the M step). As mentioned before, this allows having two different estimation techniques: non-parametric for f and parametric for β. This is not possible in the EM algorithm where all the parameters are estimated together in a maximization step, and thus where a non-parametric component cannot be handled. This decoupling also allows for a faster maximization step. Both the EM and NPM algorithms, in the M step, maximize a concave likelihood function. In the case of the EM algorithm, the maximization is over the entire set of variables (f and β here) while in the NPM algorithm, the maximization is only over the variables β. This decreases the complexity of the maximization and can significantly reduce its running time. The complexities of the E and NP steps are comparable, with the difference that the E step does not require the computation of local averages.

Another important advantage of the NPM algorithm that has been observed in simulations is its convergence rate. In practical applications, the EM algorithm has been criticized for requiring a prohibitively large number of iterations before convergence of the estimation ([13]). After extensive simulations demonstrated in later Sections, we show that, in our application setting, the NPM algorithm requires significantly less iterations to converge. This is an extremely valuable asset when dealing with extremely large data sets in the case of online retailers and when estimates of customer behavior need to be updated frequently.

3.5.3 Theoretical results

Proof in the case where there are no customer features

In this Section, we consider the simple case where there is no vector of features X_i. This is equivalent to the case where all the customers are identical. In fact, in the latter case for every customer $\beta.X_i$ is a constant that can be considered as a common intercept. We will analytically prove that, in this simplified setting, the NPM algorithm converges and is a consistent estimator of β and f.

Setting Let us consider the case where all the customers are identical, i.e. there is no vector of features X_i. Let us also assume that rewards are binary: each customer can be assigned to reward level 0 or 1. ($R_i \in \{0, 1\}$). Let us denote:

- $\alpha_0 = \mathbb{P}(R_i = 1)$: this is the true value we want to estimate (this probability is the same for every customer)
- $\mathbb{P}(y_i = 1|R_i) = \frac{e^{\beta_{r0} R_i}}{1+e^{\beta_{r0} R_i}}$ with $\beta_{r0} > 0$
  For the sake of simplicity we do not consider an intercept in the logistic function: we assume that $\mathbb{P}(y_i = 1|R_i = 0) = 0.5$

NPM algorithm In this specific setting, the NPM algorithm becomes a parametric algorithm, as the distribution of rewards is constant across customers: $f_{X_i}(1) = \alpha_0$. We can replace f by the scalar α in the notations, and prove the convergence and consistency of the NPM estimator. Note that, even in this parametric setting, the NPM algorithm is different than the EM algorithm. The main underlying difference is that in the M step of the NPM algorithm the maximization is done only over the variable β_r and not on the distribution of rewards parameter α.

We first describe the steps of the NPM algorithm in this specific setting and then provide a proof for its convergence and consistency.

Initialization: We start with an initial estimate of β_r: $\beta_r^{(0)} = 0$.
In the k^{th} step:

NP step The NP step updates the estimation of α using $\beta_r^{(k)}$ and Bayes' rule. Let us recall that $\alpha^{(k)}$ is the value of α that solves

$$\hat{\mathbb{P}}(R_i = 1|y_i = 1) = g(\mathcal{D}, \beta^{(k)}, x, \alpha)$$

where g is defined by

$$g(\mathcal{D}, \beta, x, \alpha) = \frac{\alpha \hat{\mathbb{E}}_{X_i \simeq x} \left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i + \beta_r}} \right)}{\alpha \hat{\mathbb{E}}_{X_i \simeq x} \left(\frac{e^{\beta X_i + \beta_r}}{1+e^{\beta X_i + \beta_r}} \right) + (1 - \alpha) \hat{\mathbb{E}}_{X_i \simeq x} \left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}} \right)}$$

In the case where $X_i = 0$ the expression of g can be simplified. As there are no features there is no need to average over the values of X_i. Thus g becomes:

$$g(\mathcal{D}, \beta_r, x, \alpha) = \frac{\alpha \frac{e^{\beta_r}}{1+e^{\beta_r}}}{\alpha \frac{e^{\beta_r}}{1+e^{\beta_r}} + (1-\alpha)\frac{1}{2}}$$

Let us denote RB the empirical probability of having a reward given that the customer is a buyer:

$$RB = \frac{\sum_i y_i R_i}{\sum_i y_i}$$

then $\alpha^{(k)}$ satisfies

$$RB = \frac{\alpha^{(k)} \frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}}}{\alpha^{(k)} \frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}} + (1-\alpha^{(k)})\frac{1}{2}}$$

and thus, inverting the function g we get:

$$\alpha^{(k)} = \frac{-\frac{1}{2} RB}{\left(\frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}} - \frac{1}{2}\right) RB - \frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}}} \tag{3.6}$$

Note that if $\beta_r^{(0)} = 0$ then $\alpha^{(0)} = RB$, i.e. the initial estimation of α is the empirical estimation of $\mathbb{P}(R_i = 1 | y_i = 1)$ introduced in the naive approach. Thus $\alpha^{(0)} > \alpha_0$.

M step In this setting the incomplete data likelihood given in equation (3.2) becomes:

$$\mathcal{L}_i(\mathcal{D}, \beta, \alpha) = \sum_{i=1}^{N} y_i R_i \log\left(\alpha \frac{e^{\beta_r}}{1+e^{\beta_r}}\right) + y_i(1-R_i)\log\left((1-\alpha)\frac{1}{2}\right) + (1-y_i)\log\left(\frac{\alpha}{1+e^{\beta_r}} + \frac{1-\alpha}{2}\right)$$

Notice that only the first and last term of the sum are functions of β_r.

The M step updates the estimation of β_r given $\alpha^{(k)}$ by setting:

$$\beta_r^{(k+1)} = \text{argmax}_\beta \mathcal{L}_i(\mathcal{D}, \beta, \alpha^{(k)})$$

Convergence and consistency In this paragraph we provide an analytical proof of the convergence and consistency of the NPM estimates of β_r and α. The proof is in two steps. We first start showing that, starting from $\beta_r^{(0)} = 0$ the sequence $\left(\beta_r^{(k)}\right)_k$ is increasing and thus converging. We then show that the limit point of $\left(\beta_r^{(k)}, \alpha^{(k)}\right)$ is the maximum likelihood estimator of $\mathcal{L}_i$. This proves the consistency of the NPM algorithm.

Convergence

Proposition 2.4. *Starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ is a strictly increasing sequence and $\alpha^{(k)}$ is strictly decreasing.*

Proof. We describe the proof in four steps:

1. First, note that in the NP step $\alpha^{(k)}$ is defined by equation (3.6):

$$\alpha^{(k)} = \frac{-\frac{1}{2}RB}{\left(\frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}} - \frac{1}{2}\right)RB - \frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}}}$$

   where RB is estimated directly from data. The right hand side is a strictly decreasing function of $\beta_r^{(k)}$ thus, if $\beta_r^{(k)}$ increases, $\alpha^{(k)}$ strictly decreases.

2. Secondly, we can show that $\beta_r^{(1)} > \beta_r^{(0)} = 0$.

   In fact, using the result of Proposition 2.3, we know that, for a fixed value of α, $\mathcal{L}_i(\mathcal{D}, \beta, \alpha)$ is a concave function of β. Furthermore, we know that $\alpha^{(0)} = RB > \alpha_0$. Finally,

$$\begin{aligned}\frac{\partial \mathcal{L}_i}{\partial \beta_r}(\mathcal{D}, 0, \alpha^{(0)}) &= \sum_i \frac{1}{2} y_i R_i - (1-y_i)\alpha^{(0)}\frac{1}{2} \\ &= \sum_i \frac{1}{2} y_i R_i \times \left(1 - \frac{\sum_i (1-y_i)}{\sum_i y_i}\right) > 0\end{aligned}$$

   because $\alpha^{(0)} = RB = \frac{\sum_i y_i R_i}{\sum_i y_i}$ and $\frac{\sum_i (1-y_i)}{\sum_i y_i} < 1$ (because $\mathbb{P}(y_i = 1) > \mathbb{P}(y_i = 0)$ and for N large enough, $\frac{\#\text{non-buyers}}{\#\text{buyers}} < 1$).

Thus, $\mathcal{L}_i(\mathcal{D}, \beta_r, \alpha^{(0)})$ is a concave function of β_r and its derivative at 0 is positive, thus its maximum is strictly positive and $\beta_r^{(1)} > 0$.

3. If, for a given k, $\alpha^{(k-1)} < \alpha^{(k)}$ then $\beta_r^{(k)} > \beta_r^{(k+1)}$.

   This step can be derived using the first order conditions in the M step of the NPM algorithm.

$$\begin{aligned} & \beta_r^{(k+1)} = \text{argmax}_\beta \mathcal{L}_i(\mathcal{D}, \beta, \alpha^{(k)}) \\ \Longrightarrow \quad & \frac{\partial \mathcal{L}_i}{\partial \beta_r}(\mathcal{D}, \beta_r^{(k+1)}, \alpha^{(k)}) = 0 \\ \Longrightarrow \quad & \frac{\sum_i y_i R_i}{2\sum_i (1 - y_i)} = \frac{\alpha^{(k)} \frac{e^{\beta_r^{(k+1)}}}{1+e^{\beta_r^{(k+1)}}}}{\alpha^{(k)} \frac{1}{1+e^{\beta_r^{(k)}}} + (1 - \alpha^{(k)}) \frac{1}{2}} \end{aligned}$$

   The right hand-side of equation (3.7) is increasing in both β_r and α while the left hand-side depends only on data. The pairs $(\beta_r^{(k+1)}, \alpha^{(k)})$ are the solutions to this fixed point equation. We can thus conclude that if $\alpha^{(k-1)} < \alpha^{(k)}$ then $\beta_r^{(k)} > \beta_r^{(k+1)}$.

4. Finally, combining the two previous results, we can get the result by induction:

   - Initialization: $\beta_r^{(1)} > \beta_r^{(0)}$ using step 2.
   - Assume that, for a given k, $\beta_r^{(k)} > \beta_r^{(k-1)}$. Then, using step 1 we have that $\alpha^{(k-1)} < \alpha^{(k)}$ and, using step 3, we get that $\beta_r^{(k+1)} > \beta_r^{(k)}$.
   - We can then conclude that $\beta_r^{(k)}$ is an increasing sequence and $\alpha^{(k)}$ is a decreasing sequence.

□

Proposition 2.5. *Starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ and $\alpha^{(k)}$ converge (in $\mathbb{R} \cup \{\infty\}$).*

Proof. $\beta_r^{(k)}$ and $\alpha^{(k)}$ are monotonic sequences, thus they converge. □

Consistency From the previous paragraph we have that, starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ and $\alpha^{(k)}$ are converging sequences. Let us denote $\beta_r^{(\infty)}$ and $\alpha^{(\infty)}$ their limit points. We will show here that these limit points are β_{r0} and α_0.

Proposition 2.6. *In this simplified setting without customers' features, the NPM algorithm is a consistent estimator of* (β_{r0}, α_0).

Proof. The iterations of the NPM algorithm are defined by

$$\alpha^{(k)} = \frac{-\frac{1}{2}RB}{\left(\frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}} - \frac{1}{2}\right)RB - \frac{e^{\beta_r^{(k)}}}{1+e^{\beta_r^{(k)}}}}$$

and

$$\beta_r^{(k+1)} = \text{argmax}_\beta \mathcal{L}_i(\mathcal{D}, \beta, \alpha^{(k)})$$

First of all, let us notice that, by consistency of the maximum likelihood estimator, (β_{r0}, α_0) maximize the expected incomplete data likelihood:

$$(\beta_{r0}, \alpha_0) = \text{argmax}_{(\beta_r, \alpha)} \mathbb{E}\left[\mathcal{L}_i(\mathcal{D}, \beta, \alpha)\right]$$

$\mathcal{L}_i$ is a differentiable function, thus

$$\begin{cases} \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta_{r0}, \alpha_0) = 0 \\ \frac{\partial}{\partial \alpha} \mathbb{E}\left[\mathcal{L}_i\right](\beta_{r0}, \alpha_0) = 0 \end{cases} \tag{3.7}$$

Secondly, by definition of α we have that:

$$\alpha_0 = \frac{-\frac{1}{2}RB}{\left(\frac{e^{\beta_{r0}}}{1+e^{\beta_{r0}}} - \frac{1}{2}\right)RB - \frac{e^{\beta_{r0}}}{1+e^{\beta_{r0}}}}$$

With an abuse of notation, let us denote

$$\alpha(\beta_r) = \frac{-\frac{1}{2}RB}{\left(\frac{e^{\beta_r}}{1+e^{\beta_r}} - \frac{1}{2}\right)RB - \frac{e^{\beta_r}}{1+e^{\beta_r}}}$$

We then have $\alpha(\beta_{r0}) = \alpha_0$ and $\alpha(\beta_r^{(k)}) = \alpha^{(k)}$.

Using this notation, the limit points $(\beta_r^{(\infty)}, \alpha^{(\infty)})$ are a fixed point for the system of

equations:

$$(\beta_r, \alpha) \text{ such that } \begin{cases} \alpha(\beta_r) = \alpha \\ \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta_r, \alpha) = 0 \end{cases} \tag{3.8}$$

We already know that (β_{r0}, α_0) verifies this system of equations. Showing that this system of equations has an unique fixed point, we prove that $(\beta_{r0}, \alpha_0) = (\beta_r^{(\infty)}, \alpha^{(\infty)})$ which means that the NPM is a consistent estimator.

Let us consider the function that associates to β_r the derivative of the expected likelihood with respect to the first variable, evaluated at $(\beta_r, \alpha(\beta_r))$.

$$h : \beta_r \longrightarrow \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta_r, \alpha(\beta_r))$$

Then:

- $h(\beta_{r0}) = 0$
- h is a decreasing function

$$h(\beta) = \frac{1}{1+e^{\beta}} \left(\mathbb{E}(y_i R_i) - \mathbb{E}\left((1-y_i) \frac{\alpha(\beta) \frac{e^{\beta}}{1+e^{\beta}}}{\alpha(\beta)\left(\frac{1}{1+e^{\beta}} - \frac{1}{2}\right) + \frac{1}{2}} \right) \right)$$

because the first term does not depend on β and $\beta :\to \frac{\alpha(\beta) \frac{e^{\beta}}{1+e^{\beta}}}{\alpha(\beta)\left(\frac{1}{1+e^{\beta}} - \frac{1}{2}\right) + \frac{1}{2}}$ is an increasing function of β.

Thus β_{r0} is the only root of h thus (β_{r0}, α_0) is the only fixed point of the system of equations (3.8), $(\beta_{r0}, \alpha_0) = (\beta_r^{(\infty)}, \alpha^{(\infty)})$ and the NPM algorithm is consistent.

□

Results with synthetic data: no features In the simple case where all the customers are identical, we simulate synthetic data with different values of (α, β_r) and different number of customers (N) and analyze the performance of the NPM algorithm in terms of convergence to the true value and running time.

We build 18 different data sets combining different values of the problem's dimensions:

- the number of customers $N \in \{1000, 10000\}$
- the proportion of rewards $\alpha \in \{0.2, 0.5, 0.7\}$
- the price sensitivity $\beta_r \in \{0.5, 2, 3\}$

For every combination of parameters (N, α, β_r) we run the NPM algorithm until the difference between two consecutive log-likelihood is less than 10^{-6} and the difference between two consecutive values of α and β is less than 10^{-2}.

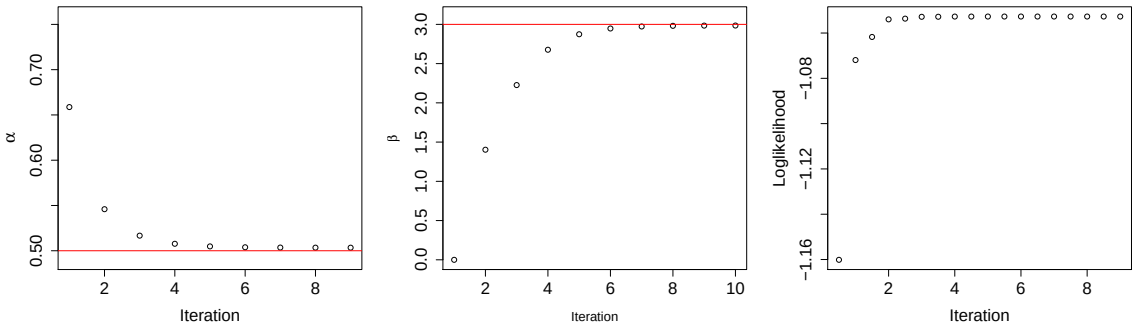


Figure 3-5: NPM algorithm for $N = 10000, \alpha_0 = 0.5, \beta_{r0} = 3$

α	β	iterations	Running time(s)
0.2	0.5	17	0.05
0.2	2.0	19	0.05
0.2	3.0	17	0.06
0.5	0.5	7	0.02
0.5	2.0	8	0.02
0.5	3.0	9	0.03
0.7	0.5	5	0.03
0.7	2.0	7	0.01
0.7	3.0	6	0.02

Table 3.1: Iterations and running time for $N = 1000$

α	β	iterations	Running time(s)
0.2	0.5	14	0.30
0.2	2.0	18	0.36
0.2	3.0	22	0.50
0.5	0.5	7	0.19
0.5	2.0	9	0.22
0.5	3.0	10	0.29
0.7	0.5	5	0.14
0.7	2.0	6	0.14
0.7	3.0	7	0.19

Table 3.2: Iterations and running time for $N = 10000$

General case with β known

The result proved in the last paragraph can be extend to a more general case including customers' features X_i.

α	β_r	$\hat{\alpha}$	$\hat{\beta}_r$	% error α	% error β_r
0.20	0.50	0.19	0.53	1.09	6.26
0.20	2.00	0.20	1.80	0.17	9.78
0.20	3.00	0.21	2.75	0.25	8.33
0.50	0.50	0.50	0.54	0.49	7.74
0.50	2.00	0.50	1.95	0.08	2.26
0.50	3.00	0.52	2.60	0.52	13.39
0.70	0.50	0.70	0.52	0.37	4.31
0.70	2.00	0.69	2.14	0.49	6.80
0.70	3.00	0.70	3.15	0.01	4.84

Table 3.3: $N = 10000$, estimated values and percentage errors

Proposition 2.7. *Assume that*

- $\mathbb{P}(y_i = 1|X_i, R_i) = \frac{e^{\beta.X_i+\beta_r R_i}}{1+e^{\beta.X_i+\beta_r R_i}}$
- $f_x(r) = \mathbb{P}(R_i = r|x)$ *and that* x *is a vector that can take a finite number of values*
- *the value of* β *is known*

Then the NPM algorithm is a consistent estimator of (β_r, f).

The second assumption is often realized in practice. Assume that f depends only on a subset of features present in X. In our application, f can depend only on the social features (number of friends, reviews etc.). These features take discrete and finite values.

The strongest assumption is the third one. X_i is the set of features that can be observed for each customer. Assuming that X and R are independent, we can get a good estimate of β (except the intercept term) by running a classical logistic regression of y_i on X_i. The only term that cannot be computed this way is β_r. Thus, in our context, it is reasonable to assume that the NPM algorithm starts with a good approximation of β.

The proof of 2.7 follows the same outline as in the no feature case. We first prove the convergence of β_r and f by proving that $\beta_r^{(k)}$ is increasing and $(f_x^k(1))_k$ is decreasing in k for any given value of x. We then prove the consistency of the estimator by proving that it is solution to a fixed point equation with an unique solution. The detail of the proof are reported in Appendix B.2.

General case

We study the general case,where β is not assumed to be known, using simulations in the next Section. Computationally, we observe that the NPM algorithm is a consistent estimator (that converges fast compared to the EM algorithm).

3.6 Results on simulated data

We presented three different approaches to jointly estimate the distribution of rewards f and the logistic regression coefficients (β, β_r):

- Direct maximization of the incomplete data likelihood ("DM")
- EM algorithm
- NPM algorithm

Recall that the incomplete data likelihood is not always concave. This makes the direct maximization computationally intensive and, we do not have guarantees that the solution we find is the global maximum.

In this Section, we compare the performances of the three approaches in terms of estimation accuracy and running time in a setting where the three approaches can be applied. We simulate $N = 1000$ customers with two features each and we apply the three approaches to this synthetic data set.

3.6.1 Simulation framework

We simulate $N = 10000$ independent customers with two features. The definition of these features is closely related to the significant features in the logistic clustering part.

Each customer is characterized by :

- d_i (days since last purchase): uniformly distributed on $[0 : 100]$
- p_i (Past Purchases): uniformly distributed on $10 \times [0 : 100]$

- the rewards levels are binary ($R_i \in \{0, 1\}$) and depend only on p_i: $\mathbb{P}(R_i = 1|X_i) = r(p_i)$ where f is a piece-wise constant function

$$f(p) = \begin{cases} \alpha_1 \text{ if } p \leq 330 \\ \alpha_2 \text{ if } 330 < p \leq 660 \\ \alpha_3 \text{ if } 660 < p \end{cases} \tag{3.9}$$

- customers decide whether to make a purchase according to a logistic function that depends on d_i and R_i: $\mathbb{P}(y_i = 1|R_i, d_i) = \frac{e^{\beta_0 + \beta_d d_i + \beta_r R_i}}{1 + e^{\beta_0 + \beta_d d_i + \beta_r R_i}}$
- d_i and p_i are independent to satisfy assumption 2.

For each instance generated, we compare the performance of the NPM algorithm, the EM algorithm and the direct maximization of the likelihood.

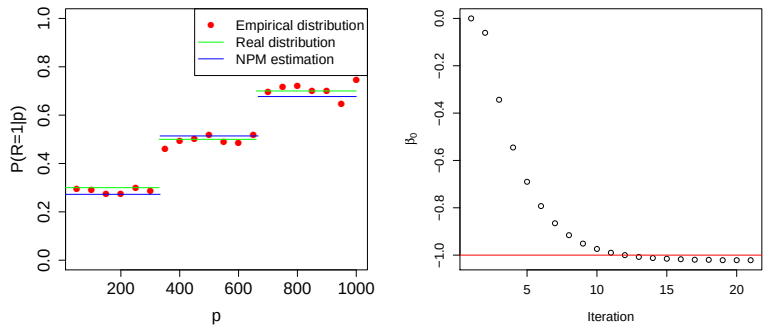
Convergence of the NPM algorithm

Figure 3-6 illustrates the convergence of the NPM algorithm in a particular instance, we can notice that the convergence rate is fast: in 20 iterations the algorithm converges and the estimated values are less than 5% far from the true parameters. It is important to remember that here we consider a setting where the distribution of rewards is parametric and where we give the same parametric assumptions (piece-wise structure of the function f) as an input for the NPM algorithm. Relaxing this hypothesis, a full non-parametric estimation of the distribution of rewards is possible.

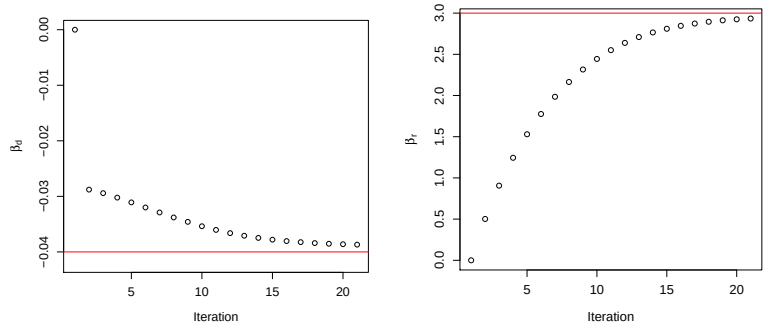
In Figure 3-7, we compare the convergence rate of the EM and NPM algorithms on the same instance as before, we can see that the NPM algorithm convergence significantly faster: the estimated $\hat{\beta}_r$ gets at less than 5% from the true value in 20 iterations while the EM algorithm requires 100 iterations to get to the same accuracy.

3.6.2 Comparing performances

In this Section, we compare the performances of the three methods in terms of accuracy and running time. We consider random values for (α, β, β_r), generate data sets of 100 000



(a) α (b) β_0



(c) β_d (d) β_r

Figure 3-6: Parameter estimation for $N = 10000, \alpha = (0.3, 0.5, 0.7), (\beta_0, \beta_d, \beta_r) = (-1, -0.04, 3)$, 20 iterations

customers and apply the three methods for parameter estimation. In Table B.1 in Appendix B.3 we reported the estimation accuracy, the number of iterations and the running time for 20 instances.

	% error β_0	% error β_d	% error β_r	# Iterations	Running Time (s)
NPM	10.4	3.73	5.69	12.0	63.6
EM	8.0	3.39	4.03	47.5	541.4

Table 3.4: Average absolute percentage error and number of iterations

In Table 3.4 we compare the NPM and the EM algorithm in terms of average absolute percentage errors on the estimation of β and average number of iterations before convergence. The average is taken over 80 instances (including the ones in Table B.1). We can notice that the two algorithms have similar accuracy but that the NPM algorithm converges significantly faster: it requires on average almost 4 times less iterations and a 8.5 times shorter running

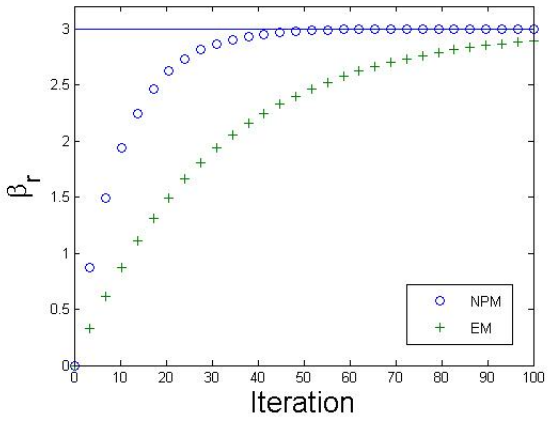


Figure 3-7: Comparing convergence rate of the EM and NPM algorithm

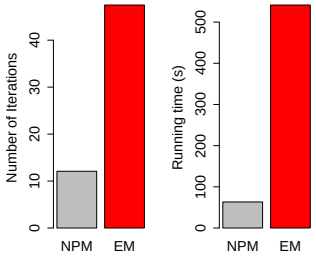


Figure 3-8: Average Number of Iterations and Running Time for NPM and EM

time. We observed this difference in speed consistently: in 96% of our simulations, the NPM algorithm requires less iterations than the EM algorithm.

In conclusion, on synthetic data, the NPM algorithm has a good performance- comparable to the EM algorithm- in terms of estimation accuracy but is significantly faster. In most of the cases, with 100 000 customers and 6 parameters to estimate it converges in less than 20 iterations. The NPM algorithm significantly outperforms the EM algorithms in terms of convergence rate.

NPM algorithm without any parametric assumption on f In the previous paragraph, we wanted to compare the performances of EM and NPM. The EM algorithm can only be used in settings where the function f has a parametric shape. Thus, we simulated a piece-wise constant distribution of rewards (defined in equation (3.9)). But the strength of the NPM algorithm is that it can be used in a general setting without any parametric

assumption. The non-parametric approach of the NPM allows to get a good estimation of f without any a priori information on its shape. To illustrate this, we use the simulation framework presented in Subsection (3.6.1) where we replace f by a bell shaped function. We simulate $N = 100,000$ customers. We then apply the NPM algorithm and use local averages as the non-parametric estimator in the NP step. Figure 3-9 shows the result of the estimation of f. The algorithm converges after 60 iterations. We can see that NPM allows a good estimation of the distribution of rewards without parametric assumptions.

We used here local average to provide an estimation of the function f. One could argue that we could get the same estimator from the EM algorithm if we make the problem parametric by dividing the range of possible values of p into small intervals. Doing so, we add a large number of additional variables α to the EM algorithm. Recall that in the EM algorithm, the Maximization step is over all the variables (and not only β), thus this will increase significantly the complexity of the M step. To conclude, the flexibility of the NP step and the fact that it decouples estimation of f and β, make the NPM algorithm extremely attractive and efficient estimator when the shape of the function f is unknown.

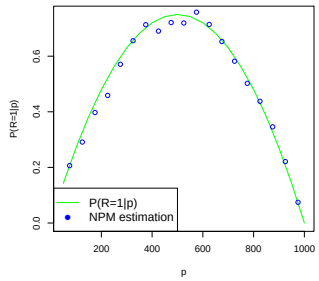


Figure 3-9: Non Parametric estimation of the function f

3.7 Conclusion

In this chapter, we studied the problem of customers' price sensitivity estimation in a setting of missing data. We first apply EM, a well-known parametric algorithm, to estimate the parameters of our model and then propose a novel non-parametric approach denoted NPM. The NPM approach allows for a more flexible and robust estimation of the distribution of

rewards using a non-parametric approach. We provide an analytical proof of the convergence and consistency of the NPM algorithm in two simplified settings. Using simulations and synthetic data, we validate its convergence on data sets with general customer' features and show that the NPM algorithm converges significantly faster (4 times less iterations and 8 less running time) than the EM algorithm. These two aspects make the NPM algorithm extremely attractive for its flexibility and convergence rate.

Chapter 4

Optimal Pricing Strategies

4.1 Introduction

In the two previous Chapters, we have built a demand estimation framework that identifies segments in the customers' population and jointly estimates their demand with a distinct logit choice model for each segment. The logit model incorporates customers' specific characteristics (previous purchase behavior and social activity) and a price sensitivity component. One last aspect is missing in the demand function: social influence. If an influential customer buys an item, how will this affect the purchase behavior of his friends? How customers' purchase behavior is influenced by a transparent rewards policy? In this Chapter, we first present a model to describe the social influence among the population segments. We then develop an optimization framework for the pricing problem. We identify the levels of rewards that should be given to each badge category in order to maximize the total revenue. We then formulate the pricing optimization problem and propose a dynamic programming approach to solve it efficiently for a general shape of demand function. Finally, we focus on the case where customers are symmetric and develop insights on the performance and behavior of optimal pricing policies.

Literature Review

Social networks have been deeply studied in marketing in the last decades. Viral marketing is defined by the Oxford English Dictionary as *"a method of product promotion that relies on getting customers to market an idea, product, or service on their own by telling their friends about it, usually by e-mail"*. In other words, on a pricing perspective, assume that after purchasing an item customers recommend it to their friends and that this encourages the latter to purchase the same item. Then, it is extremely profitable for the retailer to identify "influencers" (central customers with many friends for example) and offer them an important price incentive (even offering the item for free sometimes) to make sure that they buy the item and recommend it to their friends. The classical example comes from the fashion industry, where VIP (famous singers, actors etc.) are offered clothes for free in exchange of the advertisement the brand receives.

Different approaches have been used to model this social influence. Their common underlying structure is a connected graph where nodes represent customers and edges represent friendship relations. An approach, inspired from the economics literature, is to consider that every customer decides his purchase behavior by maximizing a utility function. This utility is customer specific and depends on his friends consumption and on the item's price (that can be different for different customers). This gives rise to a bilevel optimization problem where, in the lower level, the customers simultaneously choose their consumption by maximizing their utility given the price and, in the upper level, the retailer decides what price to offer to every customer. [17] and [19] study the case where customers' consumption is continuous while [18] focuses on when consumption is binary (customers can only choose whether to buy or not). This approach gives important insights from a theoretical point of view but, on an application perspective, it is difficult to justify a specific shape of utility function and it is even harder to estimate a distinct function for every customer.

Another possible approach relies on modeling the spread of influence as a diffusion process that can be deterministic (heat diffusion model) or probabilistic (cascade model). The reference paper for a probabilistic diffusion process is [21], where, after proposing different diffusion processes, they tackle the problem of finding the set of "VIP" to which the product

is offered for free initially in order to maximize the total consumption over the entire network. The general problem is NP-hard but heuristics are provided with good performance guarantees (see [22] for example).

Finally, [20] proposes a discrete choice model decision process in two steps. Customers face a finite set of choices: neighborhoods in which to live in their application. In the first step, customers make a decision according to a discrete choice model depending only on customer characteristics (income, family size etc.) and option characteristics (price, distance from schools etc.). In the second step, customers observe what their friends chose in the previous step and can adjust their decision. The social influence is captured in this second step.

4.2 Model

We propose a model to incorporate social influence in the demand function and then formulate the pricing problem. We do not consider individual pricing (as studied in [17] or [19]) but follow SHOP.CA business constraints: every customer in the same badge category has to receive the same reward level.

In this Chapter, to follow standard notations we will talk about prices rather than rewards. Rewards levels are percentages of discount, thus price and reward follow the simple equation $p_i = p_0(1 - r_i)$ where p_i is the price offered to customer i, p_0 is the base price (without discount) and r_i is the reward level of customer i.

4.2.1 Modeling Social Influence

Recall SHOP.CA business model illustrated in Figure 1-1. Customers are clustered into badge categories according to transparent rules. Furthermore, they know what is the rewards level they receive and what are *the rewards levels the other clusters receive.*

We first consider a setting with $K = 2$ clusters and introduce and motivate our social influence model. We then generalize our approach to a general number of clusters K. Consider a set of customers, divided into two clusters. Cluster 1 corresponds to cluster Low defined

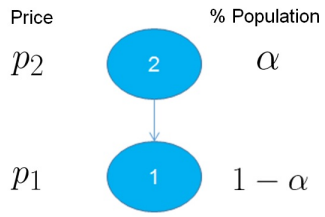


Figure 4-1: Sketch of social influence in a setting with two clusters

in Chapter 2. Cluster 2 corresponds to cluster High: the set of most influential and socially active customers. Assume that α percent of the customers are in cluster 2 and $1-\alpha$ percent are in cluster 1. Let us denote p_1 the price assigned to customers in cluster 1 and p_2 the price paid by customers in cluster 2. Assume that the influential customers receive a lower price than the non-influential.First of all, we assume that the social influence depends on the price discrimination $(p_2 - p_1)$. The intuition is the following: customers in the lower category are aware of the different prices and they know that by being more active (make more purchases, write more reviews, share more items etc.) they can reach this additional level of discount. This will increase the value they give to an additional purchase, or in other words, this will incentivize them to buy. On the other end, influencers do not receive any additional benefit from the social interactions (except the discounted price they pay). We also assume that the social influence depends on the fraction of customers in cluster 2. This is motivated by two main reasons. First of all, if the population in cluster 2 increases, this means that there are more socially active customers that send recommendations, share purchases, write reviews. This will influence the rest of the population to buy more. Secondly, on a price incentive point of view, a customer in cluster 1 who has a large fraction of his friends in cluster 2- who receive a higher discount than him- will be incentivized to accomplish the required actions to get to cluster 2. In summary, we make the following assumptions:

Assumption 3. *The demand of customer i in cluster $k \in \{1, 2\}$ depends on:*

- *X_i: the customer's specific features described in Chapter 2 (transaction history, social interactions ...).*

- *p_k: price offered to cluster k*

- *An additional social influence term is present only for cluster* 1. *We assume that the social influence depends on* α *fraction of "influencers" in the population and on* $p_1 - p_2$ *the additional level of discount offered to these influencers. Thus this influence term can be written* $f(\alpha, p_1 - p_2)$ *where* f *is increasing in both arguments. We make the further assumptions:*

  - *the effects of* α *and* $p_1 - p_2$ *are independent*
  - f *is a concave function of* α
  - f *is approximated by a linear function of* $p_1 - p_2$

  Finally we can write $f(\alpha, p_1 - p_2) = \sigma(\alpha)(p_1 - p_2)$ *where* σ *is a concave function with* $\sigma(0) = 0$ *and* $\sigma(1) = 1$.

It is common practice to assume that the influence is a concave function of the population size. In fact, intuitively, the marginal effect of having one additional influencer is decreasing with the number of influencers. [23] proves this property at a global scale (cluster level) starting from the local assumptions (customer level) of [21].

If the demand function follows a logistic model we then have the following definition.

Definition 3.1. *The demand function of customer* i *in cluster* 1 *is given by*

$$f_i^1(X_i, p_1, p_2) = \frac{e^{\beta_1 . X_i - \beta_{r,1} p_1 + \beta_{r,1}\gamma_1 \sigma(\alpha)(p_1 - p_2)}}{1 + e^{\beta_1 . X_i - \beta_{r,1} p_1 + \beta_{r,1}\gamma_1 \sigma(\alpha)(p_1 - p_2)}} \tag{4.1}$$

where σ *is a concave and increasing function that satisfies* $\sigma(0) = 0$ *and* $\sigma(1) = 1$ *and* $\gamma_1 \in [0, 1]$ *is the cross-cluster influence coefficient.*

The demand function of customer i *in cluster* 2 *is given by*

$$f_i^2(X_i, p_2) = \frac{e^{\beta_2 . X_i - \beta_{r,2} p_2}}{1 + e^{\beta_2 . X_i - \beta_{r,2} p_2}} \tag{4.2}$$

Note that the coefficients β and β_r are cluster specific (as in Chapter 2). The social influence is captured by the term $\beta_{r,1}\gamma_1 \sigma(\alpha)(p_1 - p_2)$ in equation (4.1). We introduced the coefficient γ_1 that measures the intensity of the social influence of cluster 2 on cluster 1. We assume that, if $p_1 - p_2 > 0$, i.e. the influencers receive a lower price, there is a positive social

influence, thus $\gamma_1 > 0$. We also assume that the price sensitivity is always greater than the social influence, thus $\beta_{r,1} < \beta_{r,1}\gamma_1$ i.e. $\gamma_1 < 1$. Also note that if $p_1 = p_2$, i.e. no additional discount is offered to cluster 2, then there is no social influence.

For settings with $K > 2$ clusters, we use the same model with an additional *myopic assumption*. We assume that each customer is influenced only by one cluster. This gives rise to two possible structures of influences.

4.2.2 Two special structures

We focus on two structures of social influence: the nested model, where clusters are hierarchically ordered and the VIP model where a cluster of "VIP" influences the rest of the population.

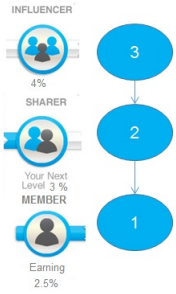


Figure 4-2: Nested model with 3 clusters

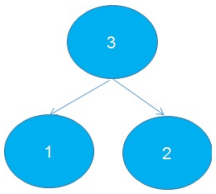


Figure 4-3: VIP model with three clusters

Nested model

The nested model is illustrated in Figure 4-2. Its structure comes from SHOP.CA business model. The lowest cluster (1) corresponds to the badge level "Member", the second cluster (2) is "Sharer" etc. With our myopic assumption, we consider that cluster k is only influenced by cluster $k+1$. This is equivalent to assuming that customers are myopic and think "one step at a time". If a customer is in cluster k, he will only consider the price offered to him and to the cluster just above him to make his purchase decision. He does not consider the additional level of discounts he would get if he was two clusters above. This assumption relies on the fact that a large number of actions are required to get from one badge level

to another (reviews, shares, friends) and thus customers consider one step at a time. The demand of customer i in cluster k is defined by:

$$f_i^k(X_i, p_k, p_{k+1}) = \frac{e^{\beta_k.X_i - \beta_{r,k}p_k + \beta_{r,k}\gamma_k\sigma(\alpha_{k+1})(p_k - p_{k+1})}}{1 + e^{\beta_k.X_i - \beta_{r,k}p_k + \beta_{r,k}\gamma_k\sigma(\alpha_{k+1})(p_k - p_{k+1})}} \tag{4.3}$$

where α_k is the fraction of customers in cluster k and we set $\alpha_{K+1} = 0$.

VIP model

The VIP model structure is illustrated in Figure 4-3. It is characterized by a cluster of "VIP" (cluster K) that influences all the other clusters $(1, \ldots, K-1)$. The typical example is a network with a group of extremely popular people (famous artists for example) that are able to influence the entire population. The rest of the customers are segmented according to specific characteristics (geographical location, interests, price sensitivity, etc.) but they do not interact between each other.

Thus the demand function of customer i in cluster $k < K$ depends only on p_k, p_K and α_K:

$$f_i^k(X_i, p_k, p_K) = \frac{e^{\beta_k.X_i - \beta_{r,k}p_k + \beta_{r,k}\gamma_k\sigma(\alpha_K)(p_k - p_K)}}{1 + e^{\beta_k.X_i - \beta_{r,k}p_k + \beta_{r,k}\gamma_k\sigma(\alpha_K)(p_k - p_K)}} \tag{4.4}$$

Note that the different lower clusters $1, \ldots, K-1$ still have different parameters (β, γ) but they all are influenced by the VIP cluster K.

In summary, the nested and VIP models represent two possible approaches to model social influence. In the nested model, K clusters are ordered hierarchically and each cluster influences the cluster below. The VIP model selects a group of highly influential customers that are able to influence the whole population. The rest of the customers are "at the same level" and segmented according to common characteristics. Our goal is to solve the pricing problem for the these two structures and compare the results in terms of pricing policies (reward levels) and total revenue.

4.2.3 Optimization Formulation

We want to assign a price p_k to each cluster k in order to maximize the total revenue taking into account the social interactions. In a general setting, let us denote $f_i^k(X_i, p_k, p_{-k})$ the demand of customer i in cluster k. p_{-k} represent the prices offered to the other clusters. Note that in the nested model f depends only on (p_k, p_{k+1}) and in the VIP model it depends only on (p_k, p_K).

The revenue generated by customer i in cluster k is $p_k f_i^k(X_i, p_k, p_{-k})$ and thus the revenue maximization problem can be written:

$$\begin{aligned} \max_{p_1,\ldots,p_K} \quad & \sum_{k=1}^{K} p_k \sum_{i \in \mathcal{C}_k} f_i^k(X_i, p_k, p_{-k}) \\ \text{s. t.} \quad & p^{min} \leq p_k \leq p^{max}, \quad k = 1, \ldots K \\ & \text{additional business rules for } p \end{aligned} \tag{4.5}$$

This problem can incorporate additional business constraints on p, for example the retailer may want to bound the difference in prices between certain clusters, or bound the sum of discounts offered. Any business rule that can be formulated with linear constraints can be added.

4.3 Dynamic Programming approach

We propose an approach to solve the optimization problem formulated in (4.5) based on Dynamic Programming. We detail our approach for the nested model. We start by simplifying the notations and aggregate the demand for every cluster. We define:

$$f_k(p_k, p_{k+1}) = \sum_{i \in \mathcal{C}_k} f_i^k(X_i, p_k, p_{k+1})$$

f_k is the total demand from cluster k.

Assume that p_1 can take continuous values while $p_2, \ldots, p_K$ take values in a discrete set $r_1, \ldots, r_J$. Imagine that the retailer has initially in mind a price p_0 for his product. He

wants to assign a base price p_1 that will be reported on his website for new members or "non-influencers". p_1 can take continuous values and can be lower or greater than p_0. For clusters 2 to K, the retailer wants to assign prices from a predefined grid $r_1, \ldots, r_J$ that represent for example a fraction of p_0. This situation is common in practice because prices often need to end by .99 or .49.

We can then use a dynamic programming approach to solve this problem. Dynamic programming is a standard approach that enables to decompose a complex optimization problem in simpler subproblems. It is based on recursive relationships called Bellman equations. They express the problem for one cluster as a function of the previous cluster in a recursive way that preserves optimality.

Here, the Bellman equations are:

$$\begin{cases} V(2, p_2) & = \max\limits_{p_1} p_1 f_1(p_1, p_2) \\ V(k+1, p_{k+1}) & = \max\limits_{p_k} p_k f_k(p_k, p_{k+1}) + V(k, p_k) \\ V(K+1) & = \max\limits_{p_K} p_K f_K(p_K) + V(K, p_K) \end{cases} \tag{4.6}$$

where V is the value function and $V(K+1)$ gives the solution to 4.5.

These equations translate the following recursive approach:

1. For $p_2 \in \{r_1, \ldots r_J\}$, fix the value of p_2 and solve the pricing problem for cluster 1:

$$\max_{p_1} p_1 f_1(p_1, p_2)$$

   Denote its optimal value $V(2, p_2)$ and store the associated optimal price $p_1(p_2)$.

k. For $k+1 \in \{3, \ldots, K\}$, fix a value of $p_{k+1} \in \{r_1, \ldots, r_J\}$ and solve the pricing problem for clusters 1 until k:

$$\max_{p_1, \ldots, p_k} \sum_{i=1}^{k} p_i f_i(p_i, p_{i+1}) = \max_{p_k} p_k f_k(p_k, p_{k+1}) + V(k, p_k)$$

where $V(k, p_k)$ denotes the optimal revenue from clusters 1 to $k-1$ given the value of p_k.

K+1. Finally, the optimal revenue for K clusters can be obtained by

$$\max_{p_K} p_K f_K(p_K) + V(K, p_K)$$

Note that for $k \geq 2$ the values of p_k are in a discrete set, thus the maximization problems $V(k, .)$ can be solved by simple enumeration.

The approach used in dynamic programming is represented graphically on Figure 4-4.

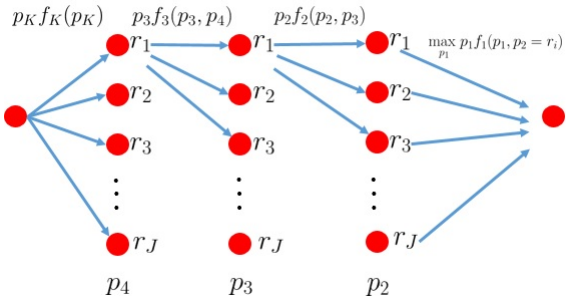


Figure 4-4: Sketch of Dynamic Programming approach

Complexity

Dynamic Programming allows to significantly reduce the complexity of the problem. Instead of considering all the possible values of $(p_2, \ldots, p_K) \in \{r_1, \ldots, r_J\}^{K-1}$, only a subset (that scales linearly with K) is considered. This makes the problem tractable even for a large number of clusters K.

Assume that $\max_{p_1} p_1 f_1(p_1, p_2)$ for p_2 fixed can be solved efficiently. This is a univariate optimization problem that can be solved fast if the objective function is concave or unimodal for example. Denote $|\mathcal{A}|$ the complexity of this problem. Then the overall complexity of the dynamic programming approach is $\mathcal{O}(J|\mathcal{A}| + J(K-1))$ where J is the cardinal of the discrete set $\{r_1, \ldots, r_J\}$. Note that this is linear in K and thus the dynamic programming approach can be solved "fast" even with a large number of clusters. In summary, if one can solve the (univariate) pricing problem for a single cluster, our Dynamic Programming approach can

be used to solve efficiently the pricing problem for K clusters.

Note that the same Dynamic Programming approach can be used for the VIP model. In this case, the Bellman equations have an easier form (as the cluster of VIP influences directly every other cluster). We first have to fix p_K and solve the pricing problem for clusters $\{1, \ldots, K-1\}$ separately and then choose the best value of $p_K \in \{r_1, \ldots, r_J\}$. The complexity of the problem becomes $\mathcal{O}((K-1)J|\mathcal{A}|)$.

To conclude, the myopic property of our demand model (where clusters are influenced only by the price offered to the cluster above) allows to decompose the optimization into simpler sub-problems and to solve it efficiently using dynamic programming if we assume that the prices offered to the clusters 2 to K lie in a discrete set. This approach can be used for any shape of demand function and different types of business constraints (order of prices, bounds on the feasible prices ...) can be incorporated.

4.4 Insights in symmetric case

After proposing an efficient way to solve the revenue optimization problem under a general demand model, we want to characterize the optimal solutions. We want to answer the questions:

- How does p_k depends on k?
- What is the role of the cross-cluster coefficient γ?
- How do the nested and VIP compare in terms of optimal policy and optimal revenue?
- How does the optimal solution depend on the distribution of customers across clusters (α)?

To answer these questions, we consider a simple setting where the customers are symmetric: every cluster has the same demand function. We start our analysis with an additional simplifying hypothesis: we assume a linear function for the demand f_k. This allows us to solve problem (4.5) in closed form, study the relationship between the solution and the problem parameters and present meaningful insights on the behavior of the solution. We will

then show using computations that the same results extend to different common demand functions (including the logistic model).

4.4.1 Symmetric Linear Model

Let us assume that the demand function is symmetric (every cluster has the same parameters and same number of customers) and linear. For K clusters we then have:

$$\begin{cases} f_k(p_k, p_{k+1}) & = \bar{d} - \beta p_k + \beta\gamma(p_k - p_{k+1}) \text{ for } k < K \\ f_K(p_K) & = \bar{d} - \beta p_K \end{cases} \tag{4.7}$$

In a linear demand model, $\bar{d}$ represents the market share (or the demand in the case where the product is offered for free) and β is the price sensitivity. The symmetry assumption means that both clusters have the same market share and price sensitivity, the only difference is the social influence. Note that the term $\sigma(\alpha)$, introduced in the initial definition of influence is dropped here. α is a constant in this case, it represents the fraction of customers in each clusters: $\alpha = \frac{1}{K}$ as the clusters are equally populated. To simplify the notations, we write $\gamma \leftarrow \sigma(\alpha)\gamma$ in the symmetric case.

Proposition 3.1. *Under a linear demand model, the nested and VIP optimization problems are quadratic problems and can be solved in closed form.*

The details of the formulation and the proof of Proposition 3.1 can be found in Appendix C.1.

4.4.2 Comparing optimal solutions for the Nested and VIP models

In this paragraph, we compare the solutions for the nested and VIP models. For $K = 2$ the two models are equivalent, we will start our analysis with $K = 3$.

Optimal prices for VIP For the VIP model, the assumption of symmetric demand function simplifies the problem. Clusters 1 to $K - 1$ have the same linear demand coefficients

$(\bar{d}, \beta, \gamma)$ and they all are influenced by the cluster of VIP (K). By symmetry of the problem, clusters 1 to $K-1$ are offered the same price.

Proposition 3.2. *For a symmetric VIP model with K clusters and linear demand the optimal prices are*

$$p_K = \frac{\bar{d}\,(\gamma - 2)}{\beta\,((K-1)\,\gamma^2 + 4\gamma - 4)}$$

$$p_1 = \ldots = p_{K-1} = \frac{\bar{d}\,(K\gamma + \gamma - 2)}{\beta\,((K-1)\,\gamma^2 + 4\gamma - 4)}$$

where $p_K < p_{K-1} = \ldots = p_1$

The cluster of VIP customers (K) receives a lower price than the other clusters. Note that the ration between p_1 and p_K depends only on γ and K. When the social influence is stronger (large γ) then the ratio increases, this translates a stronger price discrimination.

Optimal prices for the nested model For the nested model, we can solve the pricing problem in closed form for any value of K by solving the linear system presented in equation C.3 in Appendix C. Nevertheless, the symmetry of the problem in this case is not sufficient to give a closed form expression that can be applied for any value of K. For this reason, we focus here on the case where $K = 3$.

Proposition 3.3. *For a symmetric nested model with $K = 3$ clusters the optimal prices are*

$$p_3 = \frac{\bar{d}}{\beta}\frac{-3\gamma + 2}{\gamma^3 + 2\gamma^2 - 8\gamma + 4} \tag{4.8}$$

$$p_2 = \frac{\bar{d}}{\beta}\frac{\gamma^2 - 4\gamma + 2}{\gamma^3 + 2\gamma^2 - 8\gamma + 4} \tag{4.9}$$

$$p_1 = \frac{\bar{d}}{\beta}\frac{3\gamma^2 - 5\gamma + 2}{\gamma^3 + 2\gamma^2 - 8\gamma + 4} \tag{4.10}$$

and, for $\gamma < 0.5$, we have $p_3 < p_2 < p_1$.

The proof of Proposition 3.3 is straightforward and is derived from inverting the linear system of equations (C.3). Proposition 3.3 shows that the optimal price allocation gives a lower price to most influential customers. This result is intuitive and confirms the validity of our modeling assumptions. The condition $\gamma < 0.5$ required for the correct ordering of

prices, is derived from analyzing the closed form expressions for p_1, p_2, p_3 and guarantees the diagonal dominance of the matrix M presented in Appendix C.1. This is a mild assumption from a modeling perspective, because we assume that the cross-cluster influence is smaller than the price sensitivity.

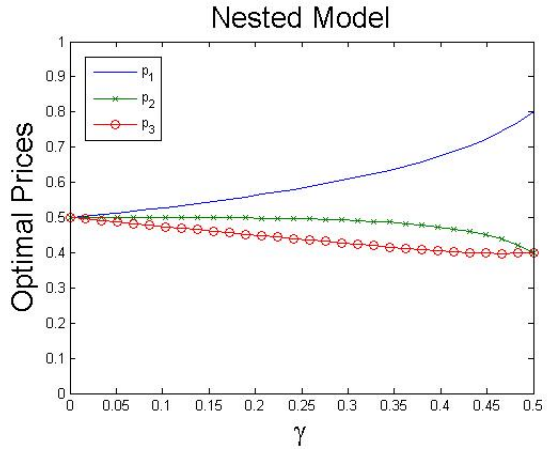


Figure 4-5: Optimal prices for the nested model with 3 clusters as a function of γ for $\frac{\bar{d}}{\beta} = 1$

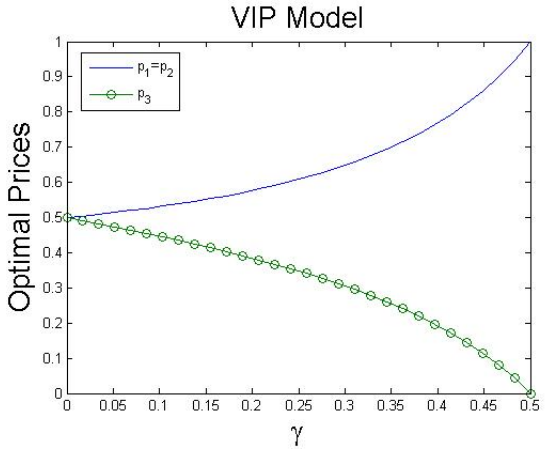


Figure 4-6: Optimal prices for the VIP model with 3 clusters as a function of γ for $\frac{\bar{d}}{\beta} = 1$

Comparing nested and VIP model We first compare the optimal pricing policies for the two models. Note that, for both models, the closed form expression for the prices can be decomposed as $\frac{\bar{d}}{\beta}$ times a function of γ. Thus, $\bar{d}$ and β influence in the same way all the clusters while γ drives the pricing difference among the clusters. Figures 4-5 and 4-6 show the optimal prices for 3 clusters and for a ratio $\frac{\bar{d}}{\beta} = 1$. Note that in both cases, when $\gamma = 0$ the prices are identical. In fact, $\gamma = 0$ corresponds to a situation without cross-cluster influence. The two clusters are then identical and they receive the same price. Also note that when γ increases the price discrimination (difference between the prices) increases. This is also intuitive, as with a high cross-cluster influence the retailer should offer lower prices to the influencers to increase the overall consumption and he can afford to offer higher prices to the lower clusters. Note also that the price discrimination is larger in the VIP model than in the nested model (for $\gamma = .45$, in the nested model $p_3 = 40, p_2 = 0.45, p_1 = 0.72$, in the VIP model $p_3 = 0.11, p_2 = p_1 = 0.86$). In general, for $K = 3$ and $\gamma < 0.5$ we have $p_{3VIP} \leq p_{3nest} \leq p_{2nest} \leq p_{1nest} \leq p_{2VIP} = p_{1VIP}$.

We can also compare the two models in terms of optimal revenue generated. We consider again the case $K = 3$ and we use the expressions for the optimal prices computed above to compute the optimal revenue in the two cases.

Proposition 3.4. *For linear symmetric demand function and $K = 3$ the ratio of the optimal revenues generated by the nested and VIP models is given by:*

$$\frac{\Pi_{VIP}}{\Pi_{nested}} = 1.5\frac{(1-\gamma)\left(\gamma^3 + 2\gamma^2 - 8\gamma + 4\right)}{\left(-\gamma^2 - \gamma + 2\right)\left(2\gamma^2 - 6\gamma + 3\right)} \geq 1$$

Thus, if the two models have the same parameters $(\bar{d}, \beta, \gamma)$, then the VIP model always generates more revenue than the nested.

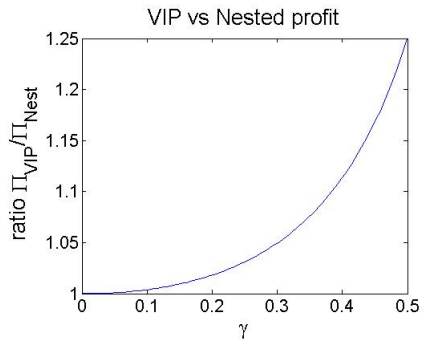


Figure 4-7: Ratio of revenues generated by the VIP and nested model for $K = 3$

Note that the ratio of revenues does not depend on $\bar{d}$ and β. It only depends on γ. Figure 4-7 represents the ratio of revenues as a function of γ. We can see that it is an increasing function of γ, thus, the greatest the social influence, the greatest is the additional revenue generated by the VIP model compared to the nested. For $\gamma = 0.5$ the VIP model generates 25% more revenue than the nested (assuming that the two models have the same parameters).

To summarize, we compared the VIP and nested models for a symmetric and linear demand model in terms of optimal pricing strategies and optimal reward generated. For $K = 3$ clusters, we have seen that both models assign lower prices to more influential customers and that the difference among prices increases with the cross-cluster influence parameter γ. The VIP model has a more "aggressive" strategy and gives an important discount to the VIP

cluster and the same and higher price to the other clusters. The nested model has a more gradual approach with less price discrimination. Assuming the same demand parameters $(\bar{d}, \gamma, \beta)$, the VIP model always generate more revenue than the nested and the the ratio of revenues is an increasing function of γ. This makes sense intuitively, when the population has a set of VIP that can influence the entire population, the retailer can extract a significant revenue by giving an important discount to the VIP and letting them influence the entire population. If there is a hierarchical structure as in the nested model, the retailer has to give discounts to a significant fraction of the population in order to have a cascading cross-cluster influence, this decreases the potential revenue.

We have detailed our analysis under a symmetric and linear model to keep the analysis tractable for the revenues and prices. The same analysis can be done with more general symmetric demand functions. The optimal prices and revenues can be derived by solving the dynamic programming approach presented in Section 4.3. We found extremely similar behaviors for other types of demand functions. We consider an exponential demand function:

$$d_{exp}(p_k, p_{k+1}) = \bar{d} e^{-\beta p_k + \beta\gamma(p_k - p_{k-1})}$$

and the logistic demand function introduced in equation (4.1).

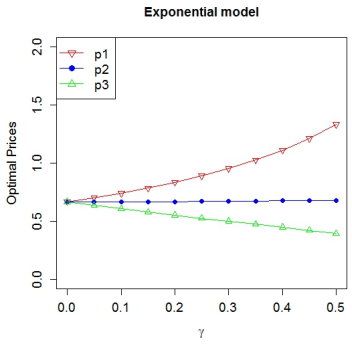


Figure 4-8: Exponential demand and nested model

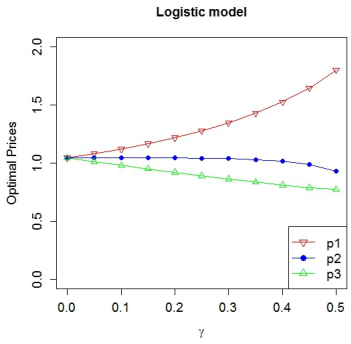


Figure 4-9: Logistic demand and nested model

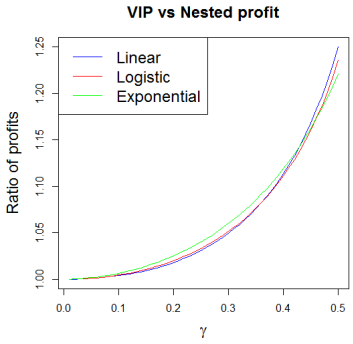


Figure 4-10: Ratio of revenue for different demand models

Figures 4-8 and 4-9 show the behavior of the optimal solutions for the nested model under exponential and logistic demand functions. The behavior of the solutions as a function of γ

is really similar to the linear case. Even more striking is Figure 4-10 where the ratio between VIP and nested revenues are extremely similar for the three possible demand shapes. We can conclude that the qualitative insights developed on the nested and VIP models under linear demand extend to other demand functions (and especially to a logistic demand) under a symmetry assumption.

Allocation of customers among clusters

In the previous paragraphs, we have considered a totally symmetric case where every cluster has the same number of customers. Let us now study the impact of different number of customers in the clusters. In this paragraph we only consider the case where there are two clusters (illustrated in Figure 4-1). Let us assume that $\alpha\%$ of the population is in cluster 2. The demand function follows the assumptions given in 3. Then, under the symmetry assumption, for a logistic demand model, we have:

$$d_1(p_1, p_2) = (1-\alpha)\frac{e^{\beta_0 - \beta_r p_1 + \beta_r \gamma \sigma(\alpha)(p_1 - p_2)}}{1 + e^{\beta_0 - \beta_r p_1 + \beta_r \gamma \sigma(\alpha)(p_1 - p_2)}} \tag{4.11}$$

$$d_2(p_2) = \alpha\frac{e^{\beta_0 - \beta_r p_2}}{1 + e^{\beta_0 - \beta_r p_2}} \tag{4.12}$$

We multiply the original demand functions presented in equation (4.3) by the number of customers in the clusters. Note also that here the parameter of interest is α thus we reintroduce the term $\sigma(\alpha)$ in the cross-cluster influence. We assumed that σ is a concave and increasing function with $\sigma(0) = 0$ and $\sigma(1) = 1$ and we consider the family of functions $\sigma(\alpha) = \alpha^i$ with $i \in (0, 1)$.

Figure 4-11 represents the optimal revenue as a function of α for $\sigma(\alpha) = \sqrt{\alpha}$. We obtain a concave and non-symmetric "bell-shaped" function that reaches its maximum for $\alpha < .5$. Note that with $\alpha = 0$ and $\alpha = 0$ there is effectively only one cluster and this generates the same level of revenue. For $\gamma \in]0, 1[$, the model builds two distinct clusters and thus the revenue is larger. We prove that the optimal revenue function is concave, under the assumption that the revenue given p is concave in α, in Proposition 3.5. This result intuitively makes sense. For a concave influence function σ it is optimal to give a

discount to a "small" fraction of customers that will be able to significantly increase the overall consumption. The value of α that maximizes the revenue depends on the shape of the influence function σ. In Figure 4-12, we represent the optimal value of α for $\sigma(x) = x^i$ for different values of $i \in (0, 1)$. The result again is intuitive. Small values of i represent a very "steep" influence function, thus it is sufficient to consider a smaller fraction of the population as influencers in order to maximizes the total revenue. Notice that for every value of $i < 1$ the optimal ratio α_{max} is lower than 50%.

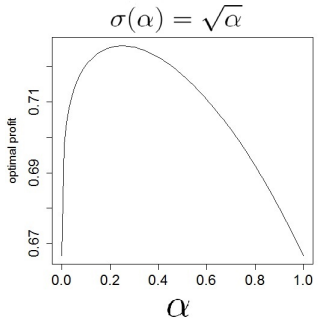


Figure 4-11: Optimal revenue as a function of α for $\sigma(\alpha) = \sqrt{\alpha}$

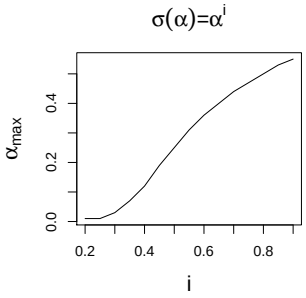


Figure 4-12: Optimal value of α for different influence functions σ

Proposition 3.5. *Let us consider two clusters and let $R(\alpha, p)$ be the revenue generated by price vector p if a proportion α of the population is in cluster 2. Let*

$$\Pi^* : \alpha \to \max_p R(\alpha, p)$$

Assume that R is a concave function of α. Then Π^* is a concave function of α.

Proof. We have that, $\Pi^*(\alpha) = \max_p R(\alpha, p)$. By assumption, for α fixed, $R(\alpha, .)$ is a concave function thus it has an unique maximum. Let us denote $p^*(\alpha) = \operatorname{argmax}_p R(\alpha, p)$. We then have that $\Pi^*(\alpha) = R(\alpha, p^*(\alpha))$. The first order conditions give $\frac{\partial}{\partial p} R(\alpha, p^*(\alpha)) = 0$. Let us

compute the second derivatives of Π^*:

$$\frac{\partial}{\partial\alpha}\Pi^*(\alpha) = \frac{\partial}{\partial\alpha}R(\alpha,p^*(\alpha)) + \underbrace{\frac{\partial}{\partial p}R(\alpha,p^*(\alpha))}_{=0}(p^*)'(\alpha) \tag{4.13}$$

$$\frac{\partial^2}{\partial^2\alpha}\Pi^*(\alpha) = \frac{\partial^2}{\partial^2\alpha}R(\alpha,p^*(\alpha)) + \underbrace{\frac{\partial^2}{\partial\alpha\partial p}R(\alpha,p^*(\alpha))}_{=0}(p^*)'(\alpha) \tag{4.14}$$

$$= \frac{\partial^2}{\partial^2\alpha}R(\alpha,p^*(\alpha)) \leq 0 \tag{4.15}$$

because, using the first order conditions, $\frac{\partial}{\partial p}R(\alpha,p^*(\alpha)) = 0$ for all α and $\frac{\partial^2}{\partial\alpha\partial p}R(\alpha,p^*(\alpha)) = \frac{\partial}{\partial\alpha}\frac{\partial}{\partial p}R(\alpha,p^*(\alpha))$. □

Conclusion on the symmetric case

In this Section,we have studied the symmetric case where every cluster has the same parameters (β,γ). We have first started our analysis considering a linear demand model. We have shown that the VIP model leads to an aggressive strategy where the price discrimination between VIP and the other clusters is important. The nested model leads to a more gradual pricing strategy with smaller price discrimination between clusters. Assuming the same demand parameters, we also show that the VIP model generates more revenue than the the nested and that the ratio of revenues depends only on the cross-cluster influence factor γ. We prove these results under the linear demand assumption and, using simulations, we show that the qualitative results extend to more general demand function (including the logistic model). We finally consider the problem of allocating customers to clusters and we show that for a strictly concave influence function $\sigma(x) = x^i$ with $i \in (0,1)$ it is always optimal to consider less than 50% of the population as influencers.

Chapter 5

Conclusion and Further Research

This research considers the problem of building a social loyalty program for an online retailer using social media data. Our approach has three main components.

In Chapter 2, we consider the problem of joint clustering and logistic regression. our goal is to cluster customers into categories according to their social media and purchase behavior and jointly build a distinct demand function for each category. We choose a logistic (binary or multinomial) function to model the customers' purchase decision process. We start by showing that the problem of joint clustering and logistic regression can be formulated as a mixed-integer optimization problem with a concave objective function (in the continuous variables) and linear constraints. We show with computations that this problem can be solved efficiently by commercial solvers even for a large number of customers. We then apply our results to SHOP.CA data and highlight two main results. First of all, incorporating social media data to a demand estimation model significantly improves the forecasting accuracy (by 13% in our application). The number of reviews written, the frequency of website log ins and the number of referrals sent are efficient predictors of the future demand. Social media data can be extremely useful for online retailers to have a more accurate understanding of customers' behavior and correctly identify segments in the population. Secondly, in a setting with heterogeneous customers, adding a clustering step in the demand estimation process can be extremely valuable. The logistic clustering model outperforms by 6% in terms of prediction accuracy an aggregated model that uses the same set of features in the demand model but does not incorporate clusters. The logistic clustering method seems an

efficient and robust approach to demand estimation with heterogeneous customers. This has to be confirmed with extensive simulations on different types of data sets coming from different application settings. Furthermore, we restricted our analysis to the "threshold implementation" for the clustering step. A study of the performances of other types of linear constraints has to be done. Finally, we carried our analysis with a set of 500 customers, with a larger amount of data, an unsupervised pre-clustering step (as in [5] for example) can be considered to reduce the computation time.

In Chapter 3, we focus on demand estimation with missing data. In Chapter 2, we estimated customers' demand as a function of past social and transaction activity: characteristics that are observable for every customer. We want to incorporate in the demand function a price sensitivity component but we have missing data. For customers that make a purchase in the future we observe the price they paid, for customers that decide not to make a purchase we do not observe the price they were offered. We want to jointly estimate the distribution of rewards among customers and to incorporate rewards in the demand function in this setting of missing data. We first introduce a classical parametric approach: the EM algorithm. Then, we introduce a novel non parametric approach, denoted NPM algorithm, that can be used without any parametric assumption on the shape of the distribution of rewards. The non parametric assumptions allow for more flexibility, a key aspect when modeling human behavior. We show that the NPM algorithm is a consistent estimator in the "no features" setting and validate our approach in the general case with extensive simulations. With synthetic data, we show that the NPM algorithm is a robust and efficient estimator that significantly outperforms the EM algorithm in terms of computation time- in our simulations, the NPM algorithm converges faster than the EM in 96% of the instances.

Finally, Chapter 4 focuses on the pricing problem. We assume that the customers are clustered into categories and that we have estimated each category's demand function. We first incorporate a social influence component in the demand and define two special structures: the nested and VIP models. We then analyze and compare these models in terms of optimal pricing and revenue. We analyze the symmetric and linear demand setting and build insights on the behavior of the optimal solutions. We show that the VIP model extracts more profit than the nested model and that, in a general setting, it is optimal to give

higher discounts to the most influential customers. We then show numerically that these results extend to nonlinear demand models as the logistic. Finally we analyze the impact of the proportion of customers considered as influencers on the total profit. The last step of this research would be to test these optimal pricing strategies (with the demand models estimated in Chapters 2 and 3).

Appendix A

Data and estimation

A.1 Data

A.1.1 Transaction Data

Transaction data	
Number of transactions	200k
Number of unique users	100k
Time range	January 2013-February 2014

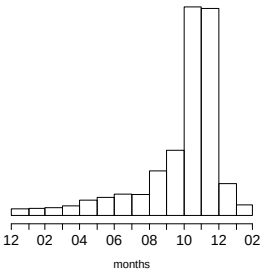


Figure A-1: Histogram of monthly transaction from January 2013 to February 2014

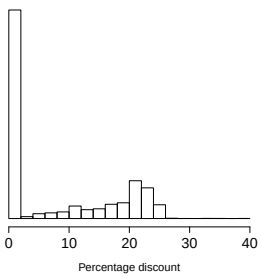


Figure A-2: Histogram of percentage of discount received

A.1.2 Social interactions

The list of possible actions is reported in the following table:

Action	Attributes	Customer Action Description
Log In		Logs in into his account
Update Profile	picture, description	Updates his profile: add or update a picture, add or update personal description
Message	Recipient ID	Sends a private message to other registered member (the recipient)
Review	ITEM_ID, Rating	Writes a review about item ITEM_ID and gives a rating
Recommendation	ITEM_ID, USER_ID	Private recommendation to another registered member about a specific item
Referral	Success	Email invitation to register to the website sent to a person that is not already a registered member. If the attribute Success=1, the recipient accepts the invitation and creates an account
Social Network Information	(Action, Social Network)	Interaction with a social network. Customers can link their account with their profile on a social network, they can post something on the social network through the online retailer website, or send messages. Possible social networks: Facebook, LinkedIn, Twitter
Friends	USER_ID	Friend added in the internal network

Table A.1: Possible Social Actions

Category	**Feature**	Description
Transactions	Number of Purchases	
	Past Purchases	Total amount spent ($)
	Number of promo codes used	
	Average rewards used	
	Days since last purchase	
Social Activity computed as of 11/1/13	Days since last log in	
	Days since last social activity	
	Active days in the last 4 months	An "active day" is a day where the customer logs in on his account
	Successful Referrer	Referrals sent with Success=1
	Referrals received	
	Reviews	Number of reviews written
	Messages	Number of messages sent and received
	Sharer	Number of item recommendations sent
	Share referree	Number of item recommendations received
	Social Networks	Total number of activities on a social network
	Friends	Number of Friends (on the website platform)
	Friends Past Purchases	Amount ($) spent on the website but the customer's friends

Table A.2: Customers Features built from Past Period

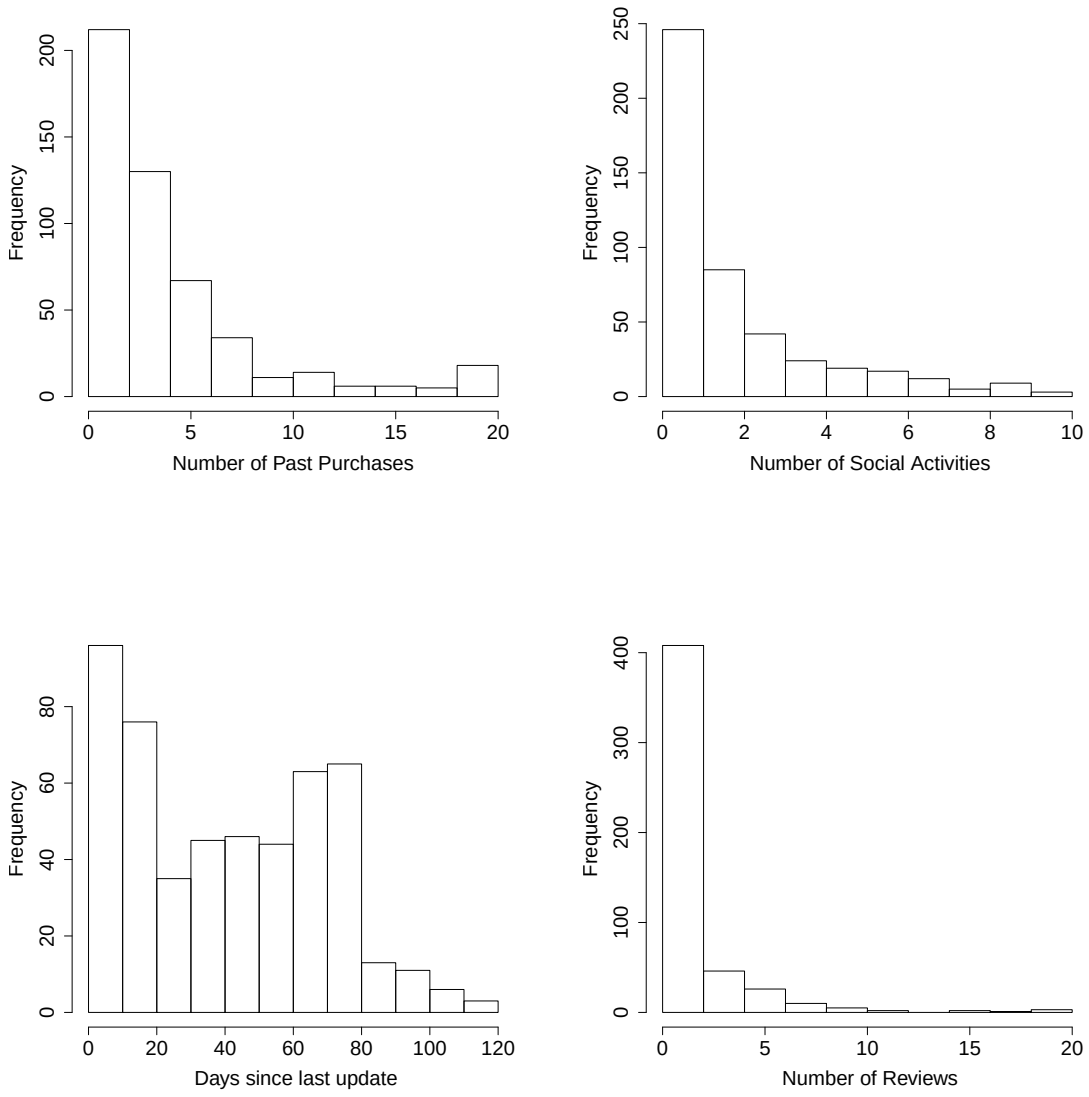


Figure A-3: Features of interested customers

A.2 Proof of Proposition 1.5

Proof. The approach is similar to the one used before, we need to define the variables δ_i and v_i^j for $j = 2 \dots J$.

As before, let us focus on a customer i and on k such that $a_{i,k} = 1$. Then we have to distinguish two cases:

- If $y_{i,0} = 1$ then the objective function can be rewritten as

$$-\ln(1 + \sum_{j\geq 1} e^{\beta_k^j.\mathbf{X}_i})$$

and the maximization can be reformulated:

$$\begin{aligned} \max \quad & -\ln(1 + e^{\delta_i} + \sum_{l\geq 2} e^{v_i^l}) \\ \text{s. t.} \quad & \delta_i \geq \beta_k^1.\mathbf{X}_i \\ & v_i^j \geq \beta_k^j.\mathbf{X}_i,\ j \geq 2 \end{aligned} \tag{A.1}$$

The objective is decreasing in every variable, thus the equality constraints can be transformed into inequalities.

- If $y_{i,0} = 0$ then the objective function can be rewritten as

$$\sum_{j\geq 1} y_{i,j}\beta_k^j.\mathbf{X}_i - \ln(1 + \sum_{j\geq 1} e^{\beta_k^j.\mathbf{X}_i})$$

and the maximization can be reformulated:

$$\begin{aligned} \max \quad & \delta_i - \ln(1 + e^{\delta_i} + \sum_{j\geq 2} e^{v_i^j}) \\ \text{s. t.} \quad & \delta_i \leq \sum_{j\geq 1} y_{i,j}\beta_k^j.\mathbf{X}_i \\ & v_i^j \geq (1 - y_{i,j})\beta_k^j.\mathbf{X}_i + y_{i,j}\beta_k^1.\mathbf{X}_i\ j \geq 2 \end{aligned} \tag{A.2}$$

as the objective is increasing in δ and decreasing in v.

We can put these two cases together using linear constraints and get our result.

□

A.3 Evaluating the performance of a classifier

Definition of confusion matrix and accuracy measures. Consider a data set with P positive labels ($y_i = 1$) and N negative labels. A classification algorithm predicts a value for each data points. Some are correctly classified some are not. To assess the predictive power of the classifier we count the number of true positives TP (the label is positive and the algorithm predicts positive), true negatives TN, false positives FP (the label is negative but the algorithm predicts positive) and false negatives FN. The algorithm predicts correctly for the true positive and the true negatives. The four possible outcomes are presented in Table A.3. Accuracy, Specificity, Sensitivity and Precision are four ways of evaluating the performance of a binary classifier in a context of an unbalanced data set (where one of the labels is present significantly more often than the other).

		Prediction	
		0	1
y_i	0	TN	FP
	1	FN	TP

Proportion of non-buyers	$\frac{N}{N+P}$
Accuracy	$\frac{TP+TN}{P+N}$
Specificity	$\frac{TN}{N}$
Sensitivity	$\frac{TP}{P}$
Precision	$\frac{TP}{TP+FP}$

Table A.3: Definition of Confusion Matrix and accuracy measures

A.4 Benchmarks

In Section 2.7.2, we introduced three alternative models to benchmark the performance of the Logistic Clustering. In what follows, we present here the details of these approaches.

Baseline The baseline is the simplest model we can think of. For every cluster, it predicts for every customer the most frequent outcome. Here, for our two clusters, the most frequent outcome is $y_i = 0$ thus the Baseline predicts that none of the customers makes a purchase in the future. This model seems rather simple but it demonstrates good accuracy in situations of unbalanced data (as in this case) where one of the two outcomes is significantly more frequent than the other. With unbalanced data, it is already hard to build a predictive model that outperforms the baseline.

Benchmark We introduce the benchmark to analyze the value of using social features in the demand estimation. In traditional revenue management, retailers use only transaction data to predict future demand. We want to show that incorporating social information about customers can turn out to be extremely valuable to estimate their future consumption. For clusters High and Low, we run a logistic regression model using only the Transaction features presented in Table A.2. In the same way as for Logistic Clustering, we use a greedy backward selection approach to eliminate non-significant features. The only significant feature is Past Purchases. The regression coefficients for Cluster Low and High are reported in Tables A.1 and A.2.

	Estimate	Std. Error	z value	$\mathbb{P}(>\|z\|)$
Intercept	-1.2761	0.2169	-5.88	$< 10^{-8}$***
Past Purchases	0.0007	0.0008	0.89	0.3723
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05				

Table A.1: Regression Results for Benchmark for cluster Low

	Estimate	Std. Error	z value	$\mathbb{P}(>\|z\|)$
Intercept	-0.9377	0.1961	-4.78	$< 10^{-5}$***
Past Purchases	0.0021	0.0004	4.69	$< 10^{-5}$***
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05				

Table A.2: Regression Results for Benchmark for cluster High

Aggregated model We introduce the aggregated model to investigate the impact of the clustering step in Logistic Clustering. The Aggregated model builds a single logistic regression for the entire interested customers data set (without splitting it into clusters Low and High). All the social and transaction features presented in Table A.2 can be used. Again, we first run a model with the entire set of features and then use a greedy backward selection approach to remove non-significant features. The regression coefficients estimated from the train set presented in 2.7.2 are reported in Table A.3.

	Estimate	Std. Error	z value	$\mathbb{P}(>\|z\|)$
Past Purchases	0.0017	0.0005	3.80	0.0001***
successful referrer	0.7441	0.2604	2.86	0.0043**
days since last log in	-0.0304	0.0038	-7.96	$< 10^{-14}$***
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05				

Table A.3: Regression Coefficients for Aggregated model (train set)

Appendix B

NPM algorithm

B.1 NPM algorithm under a general set of possible rewards $\mathcal{R}$

Let us consider that the possible rewards are in a discrete and finite set $\mathcal{R} = \{r_1, \ldots, r_K\}$. In this case, the NPM algorithm can be applied in a very similar way as in the binary case described in 3.5.1.

NP step

In the NP step we want to build an estimate of $f_x(r_k)$ for $k = 1, \ldots, K$. The approach is the same as in the binary case, but instead of inverting a ratio of linear functions, we have to solve a system of linear equations.

For every k we write Bayes' rule and, assuming that we know β, we transform it into:

$$\mathbb{P}(R_i = r_k | X_i = x, y_i = 1) = \frac{\mathbb{P}(y_i = 1 | R_i = r_k, X_i)\mathbb{P}(R_i = r_k | X_i = x)}{\mathbb{P}(y_i = 1 | X_i = x)} \tag{B.1}$$

$$\hat{\mathbb{P}}(R_i = r_k | X_i = x, y_i = 1) = \frac{\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r r_k}}{1+e^{\beta X_i \beta_r r_k}}\right) f_x(r_k)}{\sum_{q=1}^{K} f_x(r_q)\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i + \beta_r r_q}}{1+e^{\beta X_i \beta_r r_q}}\right)} \tag{B.2}$$

For every value of k, the left hand side can be estimated from the available data. Thus by multiplying equation (B.2) by the denominator of the right-hand side, it becomes a linear

equation on $f_x(r_q)$ for $q \in [1, K]$. We have a different equation for every value of k. Thus solving this system of linear equations we recover the values of f_x. Note that for every value of k the denominator in the right hand side is the same. Thus this system of equation can be solved easily (without having to invert a matrix) and we get:

$$f_x(r) = C \frac{\hat{\mathbb{P}}(R_i = r | X_i = x, y_i = 1)}{\hat{\mathbb{E}}_{X_i \simeq x} \left(\frac{e^{\beta X_i + \beta_r r}}{1 + e^{\beta X_i \beta_r r}} \right)}$$

where $\frac{1}{C} = \sum_{r \in \mathcal{R}} \frac{\hat{\mathbb{P}}(R_i = r | X_i = x, y_i = 1)}{\hat{\mathbb{E}}_{X_i \simeq x} \left(\frac{e^{\beta X_i + \beta_r r}}{1 + e^{\beta X_i \beta_r r}} \right)}$.

M step

The M step is identical to the one considered in the binary rewards case. The incomplete data likelihood given in equation (3.2) is written for a general set $\mathcal{R}$.

B.2 Proof of Proposition 2.7

The proof of Proposition 2.7 follows the same outline as the proof in the no feature case. To simplify the notations let us assume that X_i takes only a finite and discrete set of values χ. We have seen that this assumption is not necessary, we only need that the subset of X that influences f takes a finite number of values. To avoid to insert a new notation we will assume that f depends on X_i and that X_i takes a finite and discrete number of values. To mimic the notations introduced in the no features case, let us denote $\alpha_x = f_x(1)$.

Convergence

Proposition 3.6. *Starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ is a strictly increasing sequence and $\alpha_x(k)$ is strictly decreasing for every value of $x \in \chi$.*

Proof. Recall that in the no feature case we defined RB as the empirical expected reward among buyers. Here, let us define RB_x the empirical expected reward among buyers that have $X_i = x$: $RB_x = \hat{\mathbb{E}}(R | y = 1, X = x)$.

We, then, describe the proof in four steps:

1. First, note that in the NP step $\alpha_x^{(k)}$ is defined in section 3.5.1:

$$\alpha_x^{(k)} = -\frac{\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right) RB_x}{\left[\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right) - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)\right] RB_x - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right)}$$

   The right hand side is a strictly decreasing function of $\beta_r^{(k)}$ thus, if $\beta_r^{(k)}$ increases, $\alpha_x^{(k)}$ strictly decreases.

2. Secondly, we can show that $\beta_r^{(1)} > \beta_r^{(0)} = 0$.

   In fact, using the result of Proposition 2.3, we know that, for a fixed value of α, $\mathcal{L}_i(\mathcal{D}, \beta, \alpha)$ is a concave function of β. Furthermore, we know that $\alpha_x^{(0)} = RB_x > f_x(1)$. Finally,

$$\begin{aligned}
\frac{\partial \mathcal{L}_i}{\partial \beta_r}(\mathcal{D}, \beta, \beta_r = 0, \alpha^{(0)}) &= \sum_i y_i R_i \frac{1}{1+e^{\beta.X_i}} - (1-y_i)\alpha_{X_i}^{(0)} \frac{e^{\beta.X_i}}{1+e^{\beta.X_i}} \\
&= \sum_{x \in \chi} \sum_{i|X_i=x} y_i R_i \frac{1}{1+e^{\beta.x}} - (1-y_i)\alpha_x^{(0)} \frac{e^{\beta.x}}{1+e^{\beta.x}} \\
&= \sum_{x \in \chi} \frac{1}{1+e^{\beta.x}} \sum_{i|X_i=x} y_i R_i \left[1 - \frac{\sum_{i|X_i=x} 1-y_i}{\sum_{i|X_i=x} y_i} e^{\beta.x}\right] > 0
\end{aligned}$$

   because, for every value of x, $\alpha_x^{(0)} = RB_x = \frac{\sum_{i|X_i=x} y_i R_i}{\sum_{i|X_i=x} y_i}$ and $\beta_r > 0$ thus:

$$\frac{\sum_{i|X_i=x}(1-y_i)}{\sum_{i|X_i=x}} e^{\beta.x} \simeq \frac{\mathbb{P}(y_i=0|X_i=x)}{\mathbb{P}(y_i=1|X_i=x)} e^{\beta.x} = \frac{\alpha_x \frac{1}{1+e^{\beta.x+\beta_r}} + (1-\alpha_x)\frac{1}{1+e^{\beta.x}}}{\alpha_x \frac{e^{\beta.x+\beta_r}}{1+e^{\beta.x+\beta_r}} + (1-\alpha_x)\frac{e^{\beta.x}}{1+e^{\beta.x}}} \cancel{e^{\beta.x}} < 1$$

   Thus, $\mathcal{L}_i(\mathcal{D}, \beta_r, \alpha^{(0)})$ is a concave function of β_r and its derivative at 0 is positive, thus its maximum is strictly positive and $\beta_r^{(1)} > 0$.

3. If, for a given k, $\alpha_x^{(k-1)} < \alpha_x^{(k)}$ for all values of x then $\beta_r^{(k)} > \beta_r^{(k+1)}$.

   This step can be derived using the first order conditions in the M step of the NPM

algorithm.

$$
\begin{aligned}
& \beta_r^{(k+1)} = \text{argmax}_\beta \mathcal{L}_i(\mathcal{D}, \beta, \alpha^{(k)}) \\
\implies \quad & \frac{\partial \mathcal{L}_i}{\partial \beta_r}(\mathcal{D}, \beta_r^{(k+1)}, \alpha^{(k)}) = 0 \\
\implies \quad & \sum_{x \in \chi} \sum_{i | X_i = x} \underbrace{y_i R_i \frac{1}{1 + e^{\beta . x + \beta_r^{(k+1)}}} - (1 - y_i) \frac{\alpha_x^{(k)} \frac{e^{\beta . x + \beta_r^{(k+1)}}}{\left(1 + e^{\beta . x + \beta_r^{(k+1)}}\right)^2}}{\alpha_x^{(k)} \frac{1}{1 + e^{\beta . x + \beta_r^{(k+1)}}} + (1 - \alpha_x^{(k)}) \frac{1}{1 + e^{\beta . x}}}}_{\text{increasing in } \alpha \text{ and decreasing in } \beta_r} = 0
\end{aligned}
$$

The pairs $(\beta_r^{(k+1)}, \alpha^{(k)})$ are the solutions to the fixed point equation in ((B.3)). We can thus conclude that if $\alpha_x^{(k-1)} < \alpha_x^{(k)}$ for all x then $\beta_r^{(k)} > \beta_r^{(k+1)}$.

4. Finally, combining the two previous results, we can get the result by induction:

   - Initialization: $\beta_r^{(1)} > \beta_r^{(0)}$ using step 2.
   - Assume that, for a given k, $\beta_r^{(k)} > \beta_r^{(k-1)}$. Then, using step 1 we have that $\alpha_x^{(k-1)} < \alpha_x^{(k)}$ for all x and, using step 3, we get that $\beta_r^{(k+1)} > \beta_r^{(k)}$.
   - We can then conclude that $\beta_r^{(k)}$ is an increasing sequence and $\alpha^{(k)}$ is a decreasing sequence.

□

Proposition 3.7. *Starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ and $\alpha^{(k)}$ converge.*

Proof. $\beta_r^{(k)}$ and $\alpha^{(k)}$ are monotonic sequences, thus they converge. □

Consistency From the previous paragraph we have that, starting from $\beta_r^{(0)} = 0$ and assuming $\beta_{r0} > 0$, $\beta_r^{(k)}$ and $\alpha^{(k)}$ are converging sequences. Let us denote $\beta_r^{(\infty)}$ and $\alpha^{(\infty)}$ their limit points. We will show here that these limit points are β_{r0} and α_0.

Proposition 3.8. *If β is assumed to be known and X_i take discrete values, the NPM algorithm is a consistent estimator of $(\beta_{r0}, \alpha_0(x))$.*

Proof. The iterations of the NPM algorithm are defined by

$$\alpha_x^{(k)} = -\frac{\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right) RB_x}{\left[\hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right) - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i}}{1+e^{\beta X_i}}\right)\right] RB_x - \hat{\mathbb{E}}_{X_i \simeq x}\left(\frac{e^{\beta X_i+\beta_r}}{1+e^{\beta X_i+\beta_r}}\right)}$$

and

$$\beta_r^{(k+1)} = \text{argmax}_{\beta_r} \mathcal{L}_i(\mathcal{D}, \beta, \beta_r, \alpha^{(k)})$$

First of all, let us notice that, by consistency of the maximum likelihood estimator, (β_{r0}, α_0) maximize the expected incomplete data likelihood:

$$(\beta_{r0}, \alpha_0) = \text{argmax}_{(\beta_r, \alpha)} \mathbb{E}\left[\mathcal{L}_i(\mathcal{D}, \beta, \beta_r, \alpha)\right]$$

$\mathcal{L}_i$ is a differentiable function, thus

$$\begin{cases} \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta, \beta_{r0}, \alpha_0) = 0 \\ \frac{\partial}{\partial \alpha} \mathbb{E}\left[\mathcal{L}_i\right](\beta, \beta_{r0}, \alpha_0) = 0 \end{cases} \tag{B.3}$$

Secondly, by definition of α we have that:

$$\alpha_x = -\frac{\frac{e^{\beta x}}{1+e^{\beta x}} RB_x}{\left[\frac{e^{\beta x+\beta_{r0}}}{1+e^{\beta x+\beta_{r0}}} - \frac{e^{\beta x}}{1+e^{\beta x}}\right] RB_x - \frac{e^{\beta x+\beta_{r0}}}{1+e^{\beta x+\beta_{r0}}}}$$

With an abuse of notation, let us denote

$$\alpha_x(\beta_r) = -\frac{\frac{e^{\beta x}}{1+e^{\beta x}} RB_x}{\left[\frac{e^{\beta x+\beta_{r0}}}{1+e^{\beta x+\beta_r}} - \frac{e^{\beta x}}{1+e^{\beta x}}\right] RB_x - \frac{e^{\beta x+\beta_r}}{1+e^{\beta x+\beta_r}}}$$

We then have $\alpha(\beta_{r0}) = \alpha_0$ and $\alpha(\beta_r^{(k)}) = \alpha^{(k)}$.

Using this notation, the limit points $(\beta_r^{(\infty)}, \alpha^{(\infty)})$ are a fixed point for the system of equations:

$$(\beta_r, \alpha) \text{ such that } \begin{cases} \alpha(\beta_r) = \alpha \\ \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta_r, \alpha) = 0 \end{cases} \tag{B.4}$$

We already know that (β_{r0}, α_0) verifies this system of equations. Let us show that it is the only fixed point.

Let us consider the function that associates to β_r the derivative of the expected likelihood with respect to the first variable, evaluated in $(\beta_r, \alpha(\beta_r))$.

$$h : \beta_r \longrightarrow \frac{\partial}{\partial \beta_r} \mathbb{E}\left[\mathcal{L}_i\right](\beta, \beta_r, \alpha(\beta_r))$$

Then:

- $h(\beta_{r0}) = 0$
- h is a decreasing function

Thus β_{r0} is the only root of h thus (β_{r0}, α_0) is the only fixed point of the system of equations (B.4), $(\beta_{r0}, \alpha_0) = (\beta_r^{(\infty)}, \alpha^{(\infty)})$ and the NPM algorithm is consistent.

□

B.3 Simulation results

Parameters		NPM		EM		DM
α	$\beta = (\beta_0, \beta_d, \beta_r)$	$\hat{\beta}$	# iterations	$\hat{\beta}$	# iterations	$\hat{\beta}$
0.7 0.36 0.85	0.86 -0.07 2.76	0.84 -0.07 2.8	7	0.83 -0.07 2.9	10	0.83 -0.07 2.91
0.18 0.41 0.21	-0.63 -0.03 3.69	-0.64 -0.03 3.67	25	-0.65 -0.03 3.63	83	-0.64 -0.03 3.72
0.64 0.17 0.54	0.22 -0.01 2.27	0.23 -0.01 2.1	10	0.21 -0.01 2.16	61	0.19 -0.01 2.25
0.3 0.44 0.87	0.45 -0.08 2.65	0.41 -0.08 2.64	7	0.41 -0.08 2.63	30	0.41 -0.08 2.68
0.65 0.34 0.36	0.82 -0.04 3.12	0.82 -0.04 3.14	12	0.82 -0.04 3.1	38	0.83 -0.04 3.13
0.8 0.26 0.73	-0.06 -0.09 3.72	-0.03 -0.08 3.61	7	-0.03 -0.09 3.68	14	-0.05 -0.09 3.81
0.67 0.79 0.83	0.26 -0.08 2.15	0.3 -0.08 1.95	13	0.13 -0.08 2.35	57	0.23 -0.08 2.06
0.82 0.12 0.84	-0.07 -0.06 3.62	-0.06 -0.06 3.54	7	-0.06 -0.06 3.5	14	-0.06 -0.06 3.54
0.16 0.81 0.48	0.31 -0.04 3.24	0.31 -0.04 3.23	8	0.31 -0.04 3.19	32	0.32 -0.04 3.22
0.74 0.56 0.22	-0.98 -0.02 3.35	-0.99 -0.02 3.14	12	-0.99 -0.02 3.09	62	-0.99 -0.02 3.19
0.73 0.62 0.68	-0.75 -0.04 2.61	-0.36 -0.03 1.66	16	-0.79 -0.04 2.71	11	-0.78 -0.04 2.69
0.45 0.35 0.49	0.44 0 3	1.1 0 0.14	3	0.55 0 1.77	151	0.51 0 1.94
0.35 0.26 0.66	0.76 -0.01 2.92	0.78 -0.01 2.38	7	0.76 -0.01 2.66	86	0.77 -0.01 2.68
0.77 0.16 0.62	-0.97 -0.02 2.45	-0.96 -0.02 2.45	13	-0.95 -0.02 2.42	48	-0.96 -0.02 2.47
0.8 0.11 0.68	0.12 -0.05 3.55	0.15 -0.05 3.4	7	0.13 -0.05 3.31	26	0.14 -0.05 3.4
0.62 0.39 0.26	-0.08 -0.01 2.91	-0.06 -0.01 2.77	15	-0.06 -0.01 2.87	108	-0.09 -0.01 2.94
0.4 0.84 0.76	-0.68 -0.04 3.95	-0.63 -0.04 3.82	15	-0.64 -0.04 3.82	26	-0.64 -0.04 3.94
0.63 0.39 0.56	-0.87 0 2.92	-0.85 0 2.61	37	-0.84 0 2.52	131	-0.9 0 2.88
0.79 0.87 0.8	0.05 -0.07 3.03	0.27 -0.06 2.48	10	-0.06 -0.07 3.27	38	0 -0.07 3.08
0.6 0.72 0.59	-0.8 -0.06 3.38	-0.76 -0.06 3.13	25	-0.78 -0.06 3.28	11	-0.79 -0.06 3.35

Table B.1: Comparing performances of NPM, EM and DM

Appendix C

Optimal Prices in the symmetric and linear case

C.1 Closed form solution under a linear demand function

We prove Proposition 3.1 and detail the derivation of the closed form solution under a linear demand function.

C.1.1 Nested Model

Under the nested model, the demand is

$$\begin{cases} f(p_k, p_{k+1}) & = \bar{d} - \beta p_k + \beta\gamma(p_k - p_{k+1}) \text{ for } k < K \\ f_K(p_K) & = \bar{d} - \beta p_K \end{cases}$$

This can be written in a matrix way. Let $d(p) \in \mathbb{R}^K$ be the vector of demands for the different clusters. Then $d(p) = \bar{d} - Mp$ where

$$M = \begin{bmatrix} \beta(1-\gamma) & \beta\gamma & 0 & \dots & 0 \\ 0 & \beta(1-\gamma) & \beta\gamma & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & & & \beta(1-\gamma) & \beta\gamma \\ 0 & \dots & \dots & 0 & \beta \end{bmatrix} \tag{C.1}$$

C.1.2 VIP model

Under the VIP model, the demand is

$$\begin{cases} f(p_k, p_K) & = \bar{d} - \beta p_k + \beta\gamma(p_k - p_K) \text{ for } k < K \\ f_K(p_K) & = \bar{d} - \beta p_K \end{cases}$$

As for the nested model, let $d(p) \in \mathbb{R}^K$ be the vector of demands for the different clusters. Then $d(p) = \bar{d} - Mp$ where

$$M = \begin{bmatrix} \beta(1-\gamma) & 0 & \dots 0 & & \beta\gamma \\ 0 & \beta(1-\gamma) & \ddots & 0 & \beta\gamma \\ \vdots & \ddots & \ddots & \ddots & \beta\gamma \\ 0 & & & \beta(1-\gamma) & \beta\gamma \\ 0 & \dots & \dots & 0 & \beta \end{bmatrix} \tag{C.2}$$

For both models, the revenue maximization problem can be rewritten

$$\max_{p_1,\dots,p_K} p.d(p) = \max_{p} p.(\bar{d} - Mp)$$

This is a quadratic program. Without any constraints on p derived from business rules, it can be solved in closed form using the first order conditions:

$$\nabla_p \left[p.\left(\bar{d} - Mp\right) \right] = 0 \Leftrightarrow \bar{d} = (M + M^T)p \Leftrightarrow p = (M + M^T)^{-1}\bar{d} \tag{C.3}$$

Bibliography

[1] http://www.practicalecommerce.com/articles/3960-6-Uses-of-Big-Data-for-Online-Retailers

[2] *Big data: The next frontier for innovation, competition, and productivity*, McKinsey Global Institute, http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation

[3] L. Breiman, J. Friedman, R. Olshen, C. Stone, *Classification and Regression Trees*, Wadsworth International, Belmont, CA, 1984.

[4] Jerome Friedman, *Multivariate adaptive regression splines*, The Annals of Statistics, Vol. 19 n 1, pp 1–67, 1991.

[5] Dimitris Berstimas and Romy Shioda, *Classification and Regression via Integer Optimization*, Operations Research, Vol.55 n 2 ,pp 252-271, 2007.

[6] Philipp W. Keller, Retsef Levi and Georgia Perakis, *Efficient formulations for pricing under attraction demand models*, Mathematical Programming,Vol.145 n 1-2, pp 223-261, 2014.

[7] Kenneth Train, *Discrete Choice Models With Simulations*, Cambridge University Press, Second Edition, 2009.

[8] A. P. Dempster, N. M. Laird and D. B. Rubin, *Maximum Likelihood from Incomplete Data Via the EM algorithm*, Journal of the Royal Statistical Society, Series B, Vol.39, pp 1-38, 1977.

[9] G. Vulcano, G.J. van Ryzin, and W. Chaar, *Choice-Based Revenue Management: An Empirical Study of Estimation and Optimization*, M& SOM Vol. 12 n 3, pp 71-392, 2010.

[10] Robert Phillips,A. Serdar Simsek and Garrett van Ryzin, *Estimating Buyer Willingness-to-Pay and Seller Reserve Prices from Negotiation Data and the Implications for Pricing*, (working paper).

[11] Srikanth Jagabathula and Paat Rusmevichientong, *A Two-Stage Model of Consideration Set and Choice: Learning, Revenue Prediction, and Applications*, Revenue Prediction, and Applications, 2013.

[12] Tulin Erdem, Michael Keane and Baohong Sun, *Missing price and coupon availability data in scanner panels: Correcting for the self-selection bias in choice model parameters*, Journal of Econometrics, Vol 89, pp 177-196, 1998.

[13] Jeffrey P. Newman, Mark E. Ferguson, Laurie A. Garrow and Timothy L. Jacobs, *Estimation of Choice-Based Models Using Sales Data from a Single Firm*, Manufacturing & Service Operations Management, Vol 16 n 2, pp 183-197, 2014.

[14] Vivek F. Farias, Srikanth Jagabathula, and Devavrat Shah, *A Nonparametric Approach to Modeling Choice with Limited Data*, Management Science Vol 59 n 2 , pp 305-322, 2013.

[15] C. F. Jeff Wu, *On the Convergence Properties of the EM Algorithm*, The Annals of Statistics Vol 11 n 1, pp 95-103, 1983.

[16] Alexandre Tsybakov, *Introduction to Nonparametric Estimation*, Springer Series in Statistics, 2009.

[17] Ozan Candogan, Kostas Bimpikis and Asuman E. Ozdaglar, *Optimal Pricing in Networks with Externalities*, Operations Research, Vol 60 n 4, pp 883-905, 2012.

[18] Maxime Cohen and Pavithra Harsha, *Designing Price Incentives On a Network with Social Interactions*, (working paper).

[19] Francis Bloch and Nicolas Quérou, *Pricing in Social Networks*, Games and Economic Behavior, Vol 80, pp 243-261, 2013.

[20] Antonio Paez, Darren M Scott and Erik Volz, *A Discrete Choice Model Approach to Modeling Social Influence on Individual Decision Making*, Environment and Planning B: Planning and Design, Vol 35 n 6, pp 1055-1069, 2008.

[21] David Kempe, Jon Kleinberg and Eva Tardos, *Maximizing the spread of influence through a social network*, KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp 137-146, 2003.

[22] Jason Hartline, Vahab S Mirrokni and Mukund Sundararajan, *Optimal Marketing Strategies over Social Networks*, World Wide Web Conference 2008, Refereed Track: Internet Monetization-Recommendation & Security.

[23] Elchanan Mossel and Sebastien Roch, *Submodularity of Influence in Social Networks: From Local to Global*, SIAM Journal on Computing, Vol 39 n 6, pp 2176-2188, 2010.