UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

IMPROVEMENT OF RECOMMENDATION SYSTEM FOR A WHOLESALE STORE CHAIN USING ADVANCED DATA MINING TECHNIQUES.

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN GESTIÓN DE OPERACIONES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

IVAN FERNANDO VIDELA CAVIERES

PROFESOR GUÍA:
SEBASTIÁN A. RÍOS PÉREZ

MIEMBROS DE LA COMISIÓN:
ANDRÉS MUSALEM SAID
DANIEL SCHWARTZ PERLROTH

SANTIAGO DE CHILE
MARZO 2015

Resumen

En las empresas de *Retail*, las áreas de *Customer Intelligence* tienen muchas oportunidades de mejorar sus decisiones estratégicas a partir de la información que podrían obtener de los registros de interacciones con sus clientes. Sin embargo se ha convertido en un desafío poder procesar estos grandes volúmenes de datos.

Uno de los problemas que se enfrentan día a día es segmentar o agrupar clientes. La mayoría de las empresas generan agrupaciones según nivel de gasto, no por similitud en sus canastas de compra, como propone la literatura. Otro desafío de estas empresas es aumentar las ventas en cada visita del cliente y fidelizar. Una de las técnicas utilizadas para lograrlo es usar sistemas de recomendación.

En este trabajo se procesó alrededor de medio billón de registros transaccionales de una cadena de supermercados mayorista. Al aplicar las técnicas tradicionales de *Clustering* y *Market Basket Analysis* los resultados son de baja calidad, haciendo muy difícil la interpretación, además no se logra identificar grupos que permitan clasificar a un cliente de acuerdo a sus compras históricas.

Entendiendo que la presencia simultánea de dos productos en una misma boleta implica una relación entre ellos, se usó un método de graph mining basado en redes sociales que permitió obtener grupos de productos identificables que denominamos comunidades, a las que puede pertenecer un cliente. La robustez del modelo se comprueba por la estabilidad de los grupos generados en distintos periodos de tiempo.

Bajo las mismas restricciones que la empresa exige, se generan recomendaciones basadas en las compras históricas y en la pertenencia de los clientes a los distintos grupos de productos. De esta manera, los clientes reciben recomendaciones mucho más pertinentes y no solo son basadas en los que otros clientes también compraron.

La novedosa forma de resolver el problema de segmentar clientes ayuda a mejorar en un 140% el actual método de recomendaciones que utiliza la cadena Chilena de supermercados mayoristas. Esto se traduce en un aumento de más de 430% de los ingresos posibles.

A Mi Familia
Por Su Eterno Apoyo
e Infinito Amor.
Sin Ellos Nada
de Esto Sería Posible.

Abstract

In Retail companies, Customer Intelligence teams have big room for improving their strategic decision making based on information that could be drawn from the registered interactions with their customers. However, it has become a big challenge to process big volumes data.

One of the problems they face every day is segmenting or creating groups of clients. Most companies make groups by level of expenditure, not considering similarities in their market baskets as proposed in the literature. Another difficulty they have is to increase sales in every visit to the store and also increase fidelity. One of the techniques used to accomplish that is to use recommendation systems.

In this project, almost 500 billion of transactional registers from a wholesale store chain where processed. When applying traditional techniques like Clustering and Market Basket Analysis, the results are of low quality, making it very difficult to interpret and identify groups that would allow classifying a client based on their historical purchases. Understanding the simultaneous presence of two products in a single ticket implies a relationship between them, a graph mining method based on social networking was applied and it allowed obtaining product groups that can be labeled, called communities, to which a customer can belong. The quality of the proposed model is proven, given that comparing the groups of products generated using different time periods, show stable results.

Under the same constraints required by the company, recommendations are generated based on previous purchases of customers and their belonging degree to different groups of products. This way, clients receive more relevant recommendations, not only based on what other customers also bought.

This novel technique of solving the problem of segmentation helps improving the current recommendation method used by the Chilean wholesale store company on 140%. This translates on 430% of possible income.

Contents

List of Figures

List of Tables

Part I

Content

Chapter 1

Introduction

"Nothing is impossible, the word itself says 'I'm possible'!"

Audrey Hepburn

A general overview of the purpose of this thesis project will be presented in this chapter. Initially, the problem to be solved is introduced, then general aspects about retail industry are mentioned, followed by the general and specific objectives of this work. Finally, the methodology used and the overall structure of this thesis is depicted.

Since 2010 and thereafter, Chilean economy has been growing around 5% per year. Retail industry[1], has been growing at an average rate of 10% per year since 2008. Showing incomes above US$ 48,000 million in year 2013 [39]. About US$ 16,500 million corresponds, only, to Chilean supermarkets [37], this represents near 5 % of the Gross Domestic Product of Chile (US$ 277,000 million) [94]. This growth brings an increase both the revenue and the challenges, such as better understanding of the customer and its preferences. This thesis aims to resolve a common problem for many retailers but in a novel way. In the next section the problem will be described.

1.1 The Problem

The problem this thesis aims to solve is finding a way to generate personalized recommendations based on a novel customer characterization using graph mining

[1]considering supermarkets, department stores and home improvement

techniques to discover overlapped groups of products. We have information from a Chilean wholesale store that makes recommendations to each registered customer offering coupons with discount, in selected products, this approach has very poor results in terms of the coupon redeemed. In fact, less than $0,32\%$ of the printed coupons are redeemed.

One of the first problems we faced was the profile generation, in order to obtain groups of customer with similar purchases. Nowadays, customer are grouped based only in the amount of money spent. This is described next.

We observe, in figure 1.1, the money spent by three different customers in the same period of time. Customers depicted spent relatively the same amount of money, just distributed differently in the range of products available. It is clear, that customers 2 (blue line) and 3 (light blue line) have a similar purchase pattern, which is different than customer 1 (red line).

Tipical retail analysis, would have classified these three customers in the same group, because they spent relatively the same amount of money. We would like to classify customer 1 in a group and customers 2 and 3 in another, because the products purchased by 2 and 3 and the amount of money spent in each product is very similar. In the rest of the chapters we will show our approach to this problem.

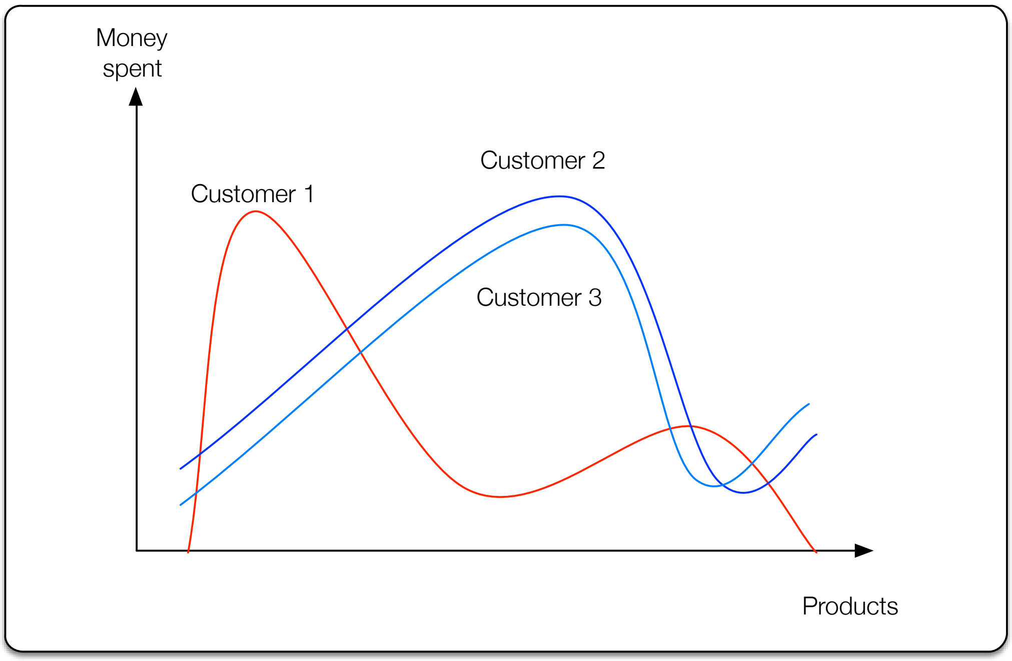


Figure 1.1: Money spent per customer in different products available.

1.2 Motivation

Generating a strong bond between customers and marketers requires a deep understanding of the consumer. From the large volumes of costumer data that companies today can store, great knowledge can be obtained. Processing this data in different ways may lead to improve different aspects of customer relationship; from marketing and advertising to customer service and experience.

It is important for a company knowing its group of customers, because optimized promotions and recommendation strategies can be built [3, 110]. Configuring layout of products at the store [32], placing related products nearby or making them accessible to customers, could be some of these actions.

Most Internet users have come across a recommender system in one way or another. Imagine, for instance, that a friend of yours recommends you to read a new book then you visit your favorite online bookstore and search. After typing in the title of the book, it appears as just one of the results listed amongst others. In one section of the web page usually called "Customers Who Bought This Item Also Bought", there is a list of other books that can be interesting to you. The software system that determines which books should be shown to a particular visitor is called a *recommendation system*.

Companies have developed a new way to manage their relationship with customers. This new way has been widely studied and it's commonly called *Customer Relationship Management* (CRM).

Typically, marketers had been trying to acquire customers, either new ones or the competitors. Today, they are trying to retain them and increase ticket size instead.

Reichheld [109] studied the dramatic increase in profits after small increases in customer retention rates. He showed an example where an increase in retention of 5% had an impact as high as 95% on the net present value delivered by customers. The fact exposed by Reichheld is complemented by Winer [128] who explains that long–term relationships with customers can deliver long–term profits. This facts have changed the way marketers see the world.

Customer relationship has evolved –in last years– from a marketing oriented approach to a customer oriented one [61]. Marketing approach is basically oriented to the quality of the product or service delivered; to maximize profits in short term, i.e., seeks to attract a large number of consumers to achieve the greatest possible number of transactions. Instead, the consumer-centric approach aims to deliver value and satisfaction to the consumer, rather than maximizing the number of transactions; it seeks to attract and retain customers in order to establish long-term customers relationships.

This approach has pushed companies to apply a number of different techniques so they can respond to customer's requests and provide them a highly and customized experience. For example, a website can personalize their content depending on the visitor characteristics as shown in [40, 90, 103]. Verhoef [123] explains that customer loyalty programs that reward the customer for affective commitment to the business have effects in retention rates. On the other side sending mass emails has no effect in customer loyalty, however extending payment date does show some benefit.

In the first part of this motivation we have shown the importance of customer relationship management. Now we will explain, that working with big amounts of data is a motivation itself, because implies big challenges extracting useful knowledge based on the ability of mine data. Indeed, goverments, scienfic communities and other industries already use advance data mining techniques for processing big volumes of data [34].

In this case we will work with data from Chilean retailers that have big databases full of customer transactions. As most of retailers, each customer is identified thanks to loyalty programs [119, 122]. These huge databases provide the necessary information to apply a methodology such as Knowledge Discovery in Databases (KDD) [43] or Cross Industry Standard Process for Data Mining (CRISP-DM) [129].

The company is currently generating 48 products recommendations for each registered customer using *association rules*. Their recommendations are retrieved as coupons by customers and less than $0,32\%$ are been redeemed.

Following this approach, data quality becomes more relevant even than data volume. Inaccurate, incomplete or simply absent data would produce very poor results. Despite the process with data mining techniques, the information obtained would be of low quality. The value of high quality data for companies is evident and depicted by facts such as companies are willing to give gifts and/or prizes to its customers when they update their information or give their RUT [2] in each purchase. In this line, Wong [130] shows that when data is missing, obsolete or has scale differences (i.e. Fahrenheit to Celsius degrees) the process of finding patterns is affected and the results can be incorrect.

In order to obtain valuable information retailers have been using several approaches to determine customer profiles including methods such as K-means [60], Self Organized Maps [73], Neural Networks [42, 68, 121], Decision Trees [106, 115] just to name a few. In the case of retail, these algorithms are applied on transactional data for discovering customer behavior that is not trivially observed. Transactions are segmented into groups (or clusters) with alike characteristics between them and different between groups. Each group found has characteristics that let us describe its members. As retailers have transactional information for each customer, once clusters are determined, is easy to find the groups where the customer belongs.

One aspect is determining customer profiles, on the other hand, retailers worldwide have commonly used Market Basket Analysis techniques to discover frequent item sets, i.e. products that are commonly present on purchases. The main family of algorithms are known as *Association Rules* [6, 7]. This kind of algorithms deliver results that enable analysts to describe and interpret relationships between products. These relationships

[2]RUT: Rol Único Tributario in spanish. National identification number.

means that when product is purchased –with high probability– the other product will be too [72].

After customer profiles and relationships between products are obtained, we can go one step forward analyzing new techniques in order to enhance the results obtained until now. In this line, an interesting approach is representing products purchased, in a retail, as a social network as is introduced in [33,107]. Social Network Analysis (SNA) [126] is an area of graph mining, which uses graphs to represent and model people, products, organizations, etc. and its relations or interactions. Usually these graphs are called *networks*. Networks can describe various types of complex systems and permit social networks analysis [112] for World wide web [41], epidemiology [92], scientific collaboration [84] just to mention some of the fields where graph theory is applied.

Algorithms from SNA are applied for discovering product communities [33]. Other studies propose a utility measure for those communities [107], and others replicate this SNA approach to the purchased problem [70]. An interesting approach for community discovering is Overlapping Community Detection. A process for finding communities in a graph. This process is similar to classical community discovering process, the most important difference is that a node can belong to more than one community. It follows logic, because a person can belongs to several communities in a social network, or a product may belong to several communities. For example, one kind of juice may belong to a community of soft drinks and to a community of soft beverages.

As we can see, exists several ways to address this problem, we will explore throughout this work these new techniques and will show our novel approach.

1.3 Significance of the study

This study presents a new approach for generating personalized recommendations of products. Beginning from detection of communities of products we made a customer characterization based on the belonging degree to each community. This methodology allows improving customer segmentation, better profiling and enhancing recommender systems amongst other.

For generating accurate groups of products it is necessary to process very large databases with million of records. Techniques used in this project makes groups easily visualized, enhancing the analysis and interpretation process.

It is important to note that communities can be very useful for other strategic decision makers, such as segmented marketing strategies, for instance sending a particular email according to the customers interests. An operational decision can be, for example, products that should be in display simultaneously for certain groups of customers.

1.4 Objectives

In this section are presented the objectives of this thesis project. Both, main objectives and specific objectives.

Main Objective

The main objective of this thesis project is to enhance the recommendation system used in a wholesale supermarket chain applying graph mining techniques over million of transactional entries.

Specific Objectives

1. Research the state–of–the–art in clustering, market basket analysis, overlapping community detection and recommendation systems.
2. Benchmark the results when applying current clustering and market basket analysis techniques over big data.
3. Apply graph mining techniques over transactional data and compare the results with those obtained previously.
4. Generate customer profiles characterizing and completing existing customer data.
5. Generate personalized recommendations and measure the quality comparing with previous company's results.

1.5 Expected Results / Contributions

1. Generate a report (chapter) with state–of–the–art techniques of clustering, market basket analysis, graph mining and recommendation systems.

2. Use an efficient software implementation of the state–of–the–art techniques on large and real transactional data (big data).

3. Use an existing software or develop one for applying graph mining techniques on transactional data.

4. Measure the quality of the results, obtained with our novel method, and compare against those obtained previously.

5. Generate a set of discount coupons per each customer and measure its quality in term of the incomes potentially obtained and the number of potential hits.

1.6 Thesis Structure

In order to carry out this thesis project, an understanding of market basket analysis, community detection and recommendations systems is needed. Main literature about data sources, customer profiles, market basket analysis and the state–of–the–art in Social Network Analysis and recommender systems are presented in chapter 2. In addition, several definitions, to fully understand this thesis project, are given. The algorithms involved to accomplish this thesis project will be presented.

Other chapters of the thesis are described next, where the methodology explains every stage completed through the application of the novel method on real data and finally conclusions and future work are presented.

Methodological Process

The methodology of this thesis is based on Cross Industry Standard Process for Data Mining (CRISP-DM) combined with Graph Mining. For achieving the main objective –personalized recommendations– frequent item sets from transactional data must be built first. This goal is reached going through various stages of data processing: Network Construction, Overlapping Community Detection, Community Characterization, Customer Profiles. Finally, personalized recommendations are generated, chapter 3 exhibits the details.

As a summary, the process can be depicted in five general stages as we show in figure 1.2.

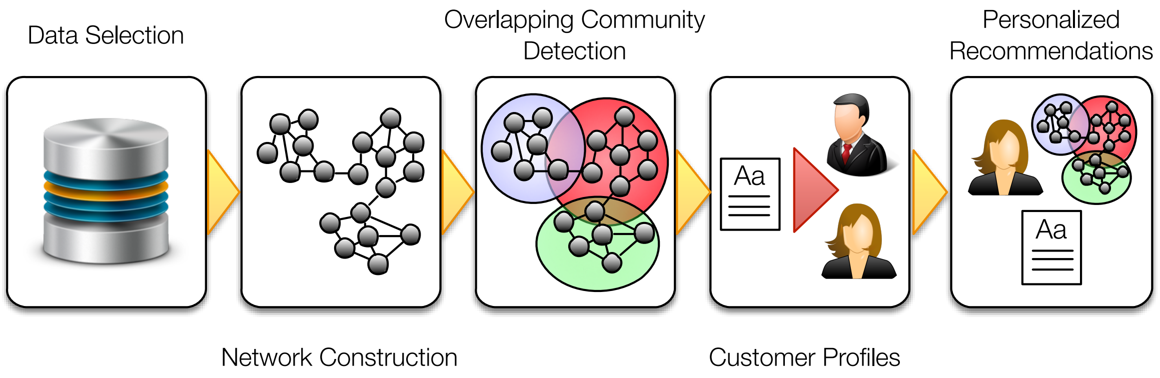


Figure 1.2: Personalized Recommendation Process.

Transactional data is selected according different time windows criteria. Then, networks of products –a graph representation using products as nodes and presence in a buying opportunity as edges– are built.

For discovering communities of products, overlapping community algorithms are applied. Communities obtained are characterized according to the products that it contains. Following, customer profiles are generated analyzing previous purchases and then personalized recommendations are calculated.

Application on Real Data

Our methodology will be tested with real data from two Chilean retail companies. The results obtained will be presented in chapter 4. Thresholds and filters developed to reduce spurious edges are explained. In order to handle millions of records, it was necessary parallel processing.

Later, the results obtained from the application of the described methodology are compared to classical approach on both analytical and expert criteria point of view. In the same chapter the system developed for handling communities and customer knowledge is presented. Its construction, interfaces and purpose are explained. Then, the results after applying the proposed methodology to generate recommendations are presented.

Conclusions and Future Work

Finally, in chapter 5, main conclusions are described, including relevant findings and contributions from this thesis project. Also, future work and following lines of research. In one aspect –the community characterization– experts found results very promising. Our method provides much better results than classical techniques for clustering and market basket analysis. On the other hand, results obtained from new recommendations are truly remarkable generating potential incomes of 434 % of the current sales generated by current recommendations.

Chapter 2

State of the Art

"Do not fear mistakes. You will know failure. Continue to reach out."

Benjamin Franklin

In this chapter, the literature will be reviewed. It is intended to give readers a brief introduction into the methods and algorithms required to understand this thesis. In first place, it is provided a description about big data, sources of data, market basket analysis and the main algorithms used to discover frequent item sets. Afterwards, it is presented a general review about graph mining and social network analysis. Finally, the current approaches to mix market basket analysis and graph mining are described.

2.1 Big Data

It is common to hear that *"We live in the information age"*. However, we believe that we live in the data age. To illustrate this, here are some facts: every day are created 2.5 quintillion bytes of data [62]. In fact, 90% of the data in the world today has been created in the last two years alone. This data came from different sources such as, sensors, climate information, posts to social media sites, digital pictures and videos, purchase transaction records, and cell phone GPS signals to name a few.

In [64] these facts are shown in numbers: Every minute in the world, 48 hours of video are uploaded to Youtube, email users send over 200 million messages, Google receives over 2 million search queries, Facebook share more than 600 thousand pieces of content, consumers spend almost US$ 300 thousand on web shopping, Apple receives about 47 thousand app downloads, Flickr users add 3,125 new photos, Foursquare users perform around 2 thousand *check-ins*[1]. These numbers show the importance of not only store this information, but being able to process it. We will described the sources of information and how can be processed in order to obtain valuable information.

2.2 Data Sources

The customer information file is the building block for a customers–focused company. Because, every decision should be based on the analysis of customer data.

Customer information may include everything about customers from their names, address, phone numbers, transaction histories and every information from customer contacts. It is a good idea, to prioritize different data elements to be included in the customer information file because exists a trade–off between costs and benefits of collecting data. There is a cost from store and process that information. The benefit is getting detailed information that leads to better decisions.

There are no standard ways to classify the types of data elements included in customer information file. In fact, the types of data elements are different according to the industry or company where they belong. In [19] it is presented a classification in six different groups: (1) customer identification data, (2) demographic data, (3) psychographic or lifestyle data, (4) transaction data, (5) marketing action data and (6) other types of data. Each type is described in the following subsections:

2.2.1 Customer Identification Data

It is the most basic customer information, covering various classification data, the customer's contact addresses and other useful identification data. Typically the information from a customer is name, a unique id, home and business addresses, home, business and mobile telephone numbers, email addresses, date of birth. In the case of business customer data, can also include names of contacts, departments, fax numbers, etc.

Customer identification information is rarely used in building response models partially because it is nominally scaled. However, they are critical in maintaining the customer relationship with the company, because they provide the means to contact the customer. It is impossible to contact a customer if the email address or home address is incorrect.

[1]Check-in, is the process whereby a person announces their arrival at a hotel, airport, etc. in several mobile applications.

Is important to mention one customer identification field, the customer ID, that is uniquely assigned to each customer upon her first contact with the company. This customer ID is used as a key field to link to other databases. This ID is used in every interaction of the customer with the company, for example this ID helps to keep a track of every purchase of the customer.

2.2.2 Demographic Data

Demographic data is data obtained, typically, from census realized every 10 years in average by the government. They include the age of head of household, family income, family size, occupation of head of household, marital status, presence of children, length of residence, education level, own or rent, type of dwelling, car ownership and its types, gender, race, and so on.

Demographic information is particularly useful for targeting prospects. Marketing researchers have found that the best predictors for customer's future purchase behaviors are their historical purchase/transaction information, Nevertheless, transactional data is only available for current customers. To be able to realize a targeting of prospects without any transactions. The demographic (or psychological) characteristics are used. For example, one could use a predictive model to identify what types of demographic characteristics a high–value customer has. Then we target prospects whose demographic profiles are similar to those of current high–value customers. Then, once prospects become customer, transactional data is collected and used to adjust the model.

2.2.3 Psychographic or Lifestyle Data

Lifestyle is a way of life or style of living that reflects the attitudes and values of a consumer while psychographics are psychological characteristics of consumers such as attitudes, values, lifestyles and opinions. According to [104], to measure lifestyles are specific questions that consist of three groups: *activities* on work, shopping, entertainment, hobbies, etc.; *interests* in fashion, recreation, job, family, food, media, etc.; *opinions* on business, politics, economics, future, products, etc.

This kind of information could be useful for a company to infer customers' attitude and behavior from their product usage. For example a technological company would like to know if their customers tend to hiking, because they might be able to buy a GPS[2] device.

[2]GPS: Global Positioning System

2.2.4 Transaction Data

In general, each record in a transactional database captures a transaction, such as a customer's purchase, a flight booking, or a user's clicks on a web page. A transaction typically includes a unique transaction identity number (*transaction_ID*) and a list of the items making up the transaction, such as the items purchased in the transaction, purchase date, salesperson ID, discount, sales tax, return code, sizes, quantity, prices, purchase amount, among others. A transactional database, may contain additional tables, with information related to the transactions, such as item description, information about the salesperson or the branch.

An example of the records inside a transactional database are depicted in 2.1:

Transaction ID	Date	SKU	Customer ID	Quantity	Price	Total Price
925	05-07-2009	P1	10021	1	350	350
925	05-07-2009	P2	10021	3	500	1500
925	05-07-2009	P4	10021	2	500	1000
926	05-07-2009	P3	-1	4	600	2400
926	05-07-2009	P4	-1	9	500	4500
927	05-07-2009	P1	1308	4	350	1400
927	05-07-2009	P3	1308	7	600	4200

Table 2.1: Example of a transaction set.

These kind of data are the most powerful data for predicting future customer purchase behavior. In fact, scanners data researchers for the last two decades have developed various models predicting consumer purchase behavior of packaged goods based on transaction histories [31, 55, 91].

Because of data storage and maintenance costs, some companies only save part of transaction data or its summary. For example, a telecommunication company has million customers and each customer makes several calls a day also their mobile devices are constantly doing *ping*[3] to the cell tower. The size of calling and logs data easily becomes terabytes in weeks. Instead of saving all calling data and logs, the transaction data is recorded in summarized quantities (only successful calls made each day, the most frequently called numbers, etc.).

2.2.5 Marketing Action Data

Transaction data and marketing action data may be the most important types of data for efficient marketing strategies and tactics. Marketing action information covers all marketing efforts directing to customers and prospects whereas transaction information is their responses to the marketing effects. For example, if we send out a catalog and the

[3]Ping: Utility used to test the reachability of a host in a network.

customer responds, the mailing of a catalog is a marketing action data and the response or no response is transaction data.

2.2.6 Other Types of Data

Besides data from different systems of a company, exist other types of data. For instance, financial data are especially important for financial institutions and service providers who require regular payments over time. They use external data providers, that supply a customer–level financial data, such as a credit score[4], credit card history, etc. As an example, in Chile, exists several credit scores from different companies, but only one *Boletín Comercial*, which is, a centralized credit history for an individual.

In this thesis project, we will focus mainly on transactional data and how useful information is extracted. In the following sections 2.4 and 2.6, we will describe the principals methods to apply over transactional data.

2.3 Customer Profile

In this section it will be explain what is understood by customer profile for this thesis project. Customer profile, according to [5] is has two parts: factual and behavioral. The factual part is based on information like, name, gender, date of birth, etc. Factual part can contain information obtained from transactional data, such as "the favorite cookie of the customer is 'fantastic cookies' ". On the other hand, behavioral part models the customer's actions and is usually derived from transactional data. For example, "when beer is purchased, the customer usually bought diapers".

First of all, we will explain the process follow to obtain a customer profile. The process begin when customers purchase a set of products in a retail store generating transactional data. This data is store in a database. Then the information pass through a clustering process, in order to obtain products that are related between them. In other words, products that are usually purchased together. This process is depicted in figure 2.1.

[4]Credit Score: Is the probability that credit users will pay their bills.

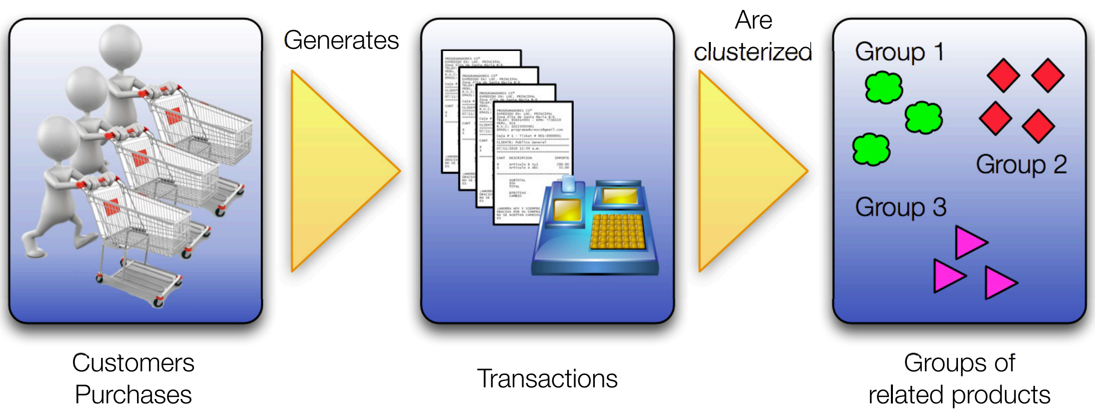


Figure 2.1: Step 1 of the customer profile generation.

Next, with the groups of related products ready. It is time to characterize groups by an analyst of the retail. Then the transactional data of a particular customer will be process to obtain the belonging grade to each of the groups found in step 1. In figure 2.2 is depicted the situation described, also are represented the belonging factors a, b and c.

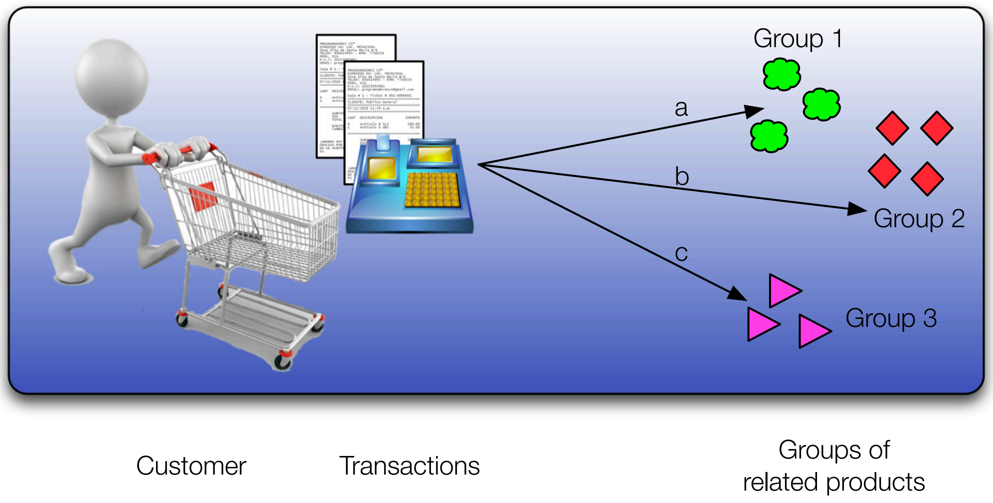


Figure 2.2: Step 2 of the customer profile generation.

Customer will belong to each of the different groups in a factor that can vary between $[0, 1]$. These factor not necessary have to sum 1. i.e., if we call a_j the belonging factor to group j we have that:

$$\sum_{j \in groups} a_j \neq 1. \tag{2.1}$$

With all this information we will obtain a profile composed by factual information as the one shown in 2.2.1, and a behavioral information as shown previously. Figure 2.3 shows the final customer profile obtained.

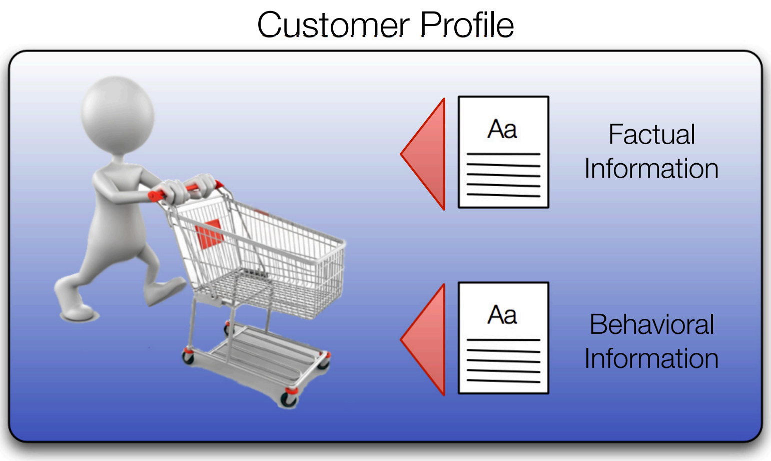


Figure 2.3: Customer profile obtained after step 1 and 2.

An important fact to note, is that we are obtaining a set of belonging factor to the groups. This is independent of the method used to obtain these groups. This fact allow us to present the groups to an analyst that will choose the best method in terms of the quality and usefulness that represent for him. After we have obtained good groups of related products we will proceed with the step 2 introduced previously.

In the following section 2.4, we will present the most popular methods to do a cluster analysis in order to obtain the groups of products related.

2.4 Clustering Analysis

Is one of the vast group of data mining techniques. The main idea is, to segment a customer database so that customers within segments are similar, and collectively different from customers in other segments. Similarity is in terms of the "clustering variables", which may be psychographics, demographics, or transaction measures such as recency, frequency or monetary value.

The clusters offer rich interpretations with strong implications for which customers should be targeted with a particular offer or marketed to in a certain way.

Unlike in classification process, where exists a class label for each customer and based on this label, elements are classified. In this case, the class label of each customer is unknown. So, through clustering analysis these groupings are discovered. Clustering is the process of partitioning a set of data objects into subsets. Each subset is a cluster, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters. In this context, different clustering methods may generate different clusterings on the same data set. The partitioning is not performed by humans, but by the clustering algorithm. Consequently, clustering is useful in that it can lead to the discovery of previously unknown groups within the data.

Cluster analysis has been widely used in many applications such as image pattern recognition [9], business intelligence [2, 125], information retrieval [87], biology [17] and security [105]. In business intelligence, clustering can be used to organize a large number of customers into groups, where customers within a group share strong similar characteristics.

In image recognition, clustering can be used to discover clusters in handwriting character recognition systems. Clustering has also found many applications in web search. For example, a keyword search may return a large number of pages relevant to the search. Clustering can be used to organize the search results into groups and present the results in a concise and easily accessible way. Besides, clustering techniques have been developed to cluster documents into topics, which are commonly used in information retrieval.

Because a cluster is a collection of data objects that are similar to one another within the cluster and dissimilar to objects in other clusters, a cluster of data objects can be treated as an implicit class. In this sense, clustering is sometimes called automatic classification. Also, one important use of clustering is outlier detection, where *outliers*[5] may be more interesting than common cases, for example an application that detects credit card fraud [29, 30].

2.5 Clustering Process

To conduct a cluster analysis, are necessary several steps:

- Select variables on which to cluster.
- Select a similarity measure and scale the variables.
- Select a clustering method.
- Determine the number of clusters.

[5]outliers : values that are far away from any cluster.

- Conduct the cluster analysis, interpret the results, and apply them.

Exists a wide variety of methods that can be used for each of these steps, especially involving similarity measures and clustering methods. Choice of these methods is often subjective.

2.5.1 Selecting Clustering Variables

One of the first questions that face an analyst is what variables should form the basis for the clustering. That is, on which set of variables do we want to form similar groups of customers? In typical applications, there are several variables available. These includes:

- *Phychographics*: Attitudes and behaviors relevant to a particular product category. For example, if the product category is consumer electronics, customer self-report as innovator, opinion leader, etc. are psychographics characteristics.
- *Demographics*: Characteristics like age, income, region, etc. are included in these variables.
- *Geo-demographics*: These includes characteristics inferred by where the customer lives. For example, from census, a company can infer the average income of a customer.
- *Behavior*: These includes recency, frequency and monetary value (RFM) behaviors measured from company's customer file [23]. Recency refers to how recently did the customer purchase. Frequency refers to how often do they purchase. Finally, Monetary value, refers to how much do they spend on each purchase.
- *Competitive measures*: These include *share–of–wallet* –which is the percentage ("share") of a customer's expenses ("of wallet") for a product that goes to the firm selling the product– [85]. Includes also, competitor preferences, etc.
- *Customer Value*: These may include responsiveness to marketing, lifetime value, customer lifetime duration, etc.

The choice of variables will depend on the application. For instance, for a cross–selling application, clustering on channel usage or RFM for various departments might be useful.

2.5.2 Similarity Measures

The main idea of clustering, is to group customers into several homogeneous clusters. Customers in the same cluster are supposed to be similar while subjects across different clusters are dissimilar. The process of clustering begins with selecting a similarity measure and a set of variables regarding which the similarity is calculated.

Exists two kinds of similarity measures: distance type and matching type. Which are describe next.

2.5.2.1 Distance–type Similarity Measures

Distance types of similarity measures are more appropriate when variables are measured on a common metric, so that the similarity between two customers can be measured on a common metric, so that the similarity between two customers can be measured as their distance in a metric space.

For example, three customers are plotted in figure 2.4. Each customer is represented as a point in two–dimensional space. These attributes could represent, for example, income and age. Here, we are implicitly using the distance between points or subjects as the similarity measure.

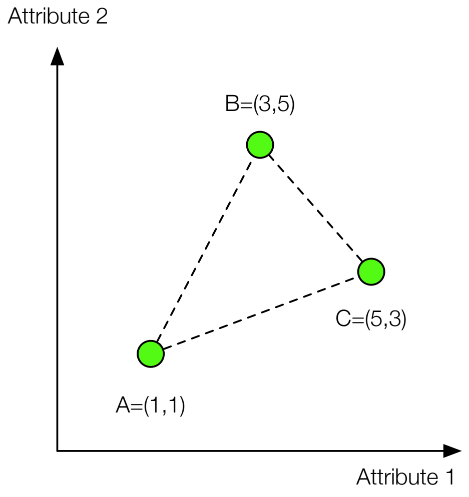


Figure 2.4: Customers and their attributes.

The most popular distance similarity measure is the Euclidean distances between two points [58]. The Euclidean distance between two subjects of dimension p , ie. $X = [x_1, x_2, \ldots, x_p]$ and $Y = [y_1, y_2, \ldots, y_p]$ is defined as

$$d(x, y) = [(x_1 - y_1)^2 + \cdots + (x_p - y_p)^2]^{\frac{1}{2}} = [(x - y)'(x - y)]^{\frac{1}{2}} \tag{2.2}$$

Each subject is represented in two–dimensional space ($p = 2$) in figure 2.4. For example, the Euclidean distance between customers A and B is $(1-3)^2+(1-5)^2]^{\frac{1}{2}} = \sqrt{20}$

and the distance between B and C is $\sqrt{8}$. The distance between B and C is the shortest, so A and B are the most similar.

The Minkowski distance is a metric that generalizes the concept of the Euclidean distance. The Minkowski distance between two p-dimensional subjects is given by

$$d(x,y) = \left(\sum_{i=1}^{p} |x_i - y_i|^m \right)^{\frac{1}{m}} \tag{2.3}$$

It is clear, that the Minkowski distances become the Euclidean distance when $m = 2$.

2.5.2.2 Matching-Type Similarity Measure

In the case of categorical variables, geometric distance is not a meaningful one. Instead, we can use the degree of matching to measure the similarity between customers when they are represented by a set of categorical characteristics. For example, two customers are treated as similar if both of them are students.

We measure the similarity between two customers as the degree of matching, specifically, the ratio of the number of matching attributes to the total number of attributes considered. For example, suppose that two customers are represented by the presence or absence (coded as 1 or 0 respectively) of five attributes. If customer A's attributes can be represented as $x = [0, 0, 1, 1, 1]$ and customer B's as $y = [1, 0, 1, 1, 0]$, then there are three matches out of five attribute comparisons and, so the similarity between customers A and B is 0.6.

The are some variants of matching–type similarity measures such as assigning differential weighting on matching from mismatching cases. For example, when are a large number of zeros in the data, we assign a large weight on the matches of ones.

2.5.3 Scaling and weighting

Metric variables are frequently measured in different units, and this can distort the cluster analysis results. For instance, if we multiply one variable by a large number, then the similarity measure will be dominated by the value of the variable. The idea is, to rescale the variables such that a percentage change of one variable is not more significant than the same percentage change of another variable in similarity calculation.

Exists several approaches to rescale the variables, one is to rescale all the variables to range from zero to one. If X_i is the original variable and X_i^* is the scaled variable, $X_i^* = (X_i - X_{min})/(X_{max} - X_{min})$ where X_{min} and X_{max} are the minimum and the maximum observed values for the original variable respectively. Another one is, rescale variable $Z_i = (X_i - \bar{X})/\sigma$, where $\bar{X}$ and σ represents the mean and the standard deviation of the original variable respectively.

Another way to deal with the scaling issue, is to consider weighting each of the clustering variables. For example, if we believe that the income variable is much more important than the age, it is reasonable to assign a large weight to income variable by multiplying it by a large number (at least larger than one).

There are no correct answer to the scaling and weighting problem, so it is necessary to try several methods in order to find the best result.

2.5.4 Clustering methods

The goal of clustering is to group subjects into an arbitrary number of segments such that customers in the same segment are similar in their characteristics while customers across different segments are dissimilar. However, it is not a simple task to find the optimal clustering solution. For instance, there exists millions of ways to cluster only 20 customers. While there is only one way to form a cluster with 20 customers, there exist $524,287$ ways to group 20 customers into two clusters. In fact the number of ways of sorting n subjects into k nonempty groups can be computed by a Stirling number of the second kind that is given by [52]:

$$\left\{ {n \atop k} \right\} = \frac{1}{k!} \sum_{j=0}^{k} (-1)^{k-j} \binom{j}{k} j^n \tag{2.4}$$

There exist several algorithm available for clustering. They are broadly classified in four groups: partitioning, hierarchical, density–based and grid–based methods [19, 59].

Partitioning Methods The simplest version of cluster analysis is partitioning, which organizes the objects of a set into several exclusive groups or clusters. We assume that the number of clusters is given. This parameter is the starting point for partitioning methods.

Formally, given a data set, D, of n objects, and k, the number of clusters to form, a partitioning algorithm organizes the objects into k partitions ($k \leq n$), where each partition represents a cluster. The clusters are formed to optimize an objective partitioning criterion, such as a similarity function based on distance, so that the objects within a cluster are *similar* to one another and *dissimilar* to objects in other clusters in terms of the data set attributes.

Hierarchical Methods These methods are based in the creation of a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either *agglomerative* or *divisive*, based on how the hierarchical decomposition is formed.

The *agglomerative* approach, starts with each object forming a separate group. It successively merges the objects or groups close to one another, until all the groups are merged into one, or a termination condition is reached. The *divisive* approach, starts with all the objects in the same cluster. In each successive iteration, a cluster is split into smaller clusters, until eventually each object is in one cluster, or a termination condition holds.

Density–based Methods The majority of the methods cluster objects based on the distance between objects. These methods can find only spherical–shaped clusters and encounter difficulty in discovering clusters of arbitrary shapes. Other clustering methods have been developed based on the *density* notion. The main idea is to continue growing a given cluster as long as the density –understood as the number of objects– in the vicinity exceeds some threshold. For instance, for each data point within a given cluster, the vicinity of a given radius has to contain at least a minimum number of points. These kind of methods can be used to filter out noise or outliers and discovery clusters of arbitrary shape.

Grid–based Methods These methods quantize the object space into a finite number of cells that form a grid structure. All the clustering operations are performed on the grid structure. The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantize space.

2.5.5 Algorithms for clustering

Exists a wide range of algorithms for clustering a set of elements. As we depicted previously are four families of methods to clusterize. In this section we will explain two algorithms that are widely used, K–means and Self Organizing Feature Maps.

2.5.5.1 K–Means

The K–means [60] may be the most popular clustering method among data miners. This algorithm is a centroid–based technique and belongs to the family of partitioning methods. Let's explain how this algorithm works:

Suppose a data set, D, contains n objects in Euclidean space. Partitioning methods distribute the objects in D into k clusters, $C_1, \ldots, C_k$, that is, $C_i \subset D$ and $C_i \cap C_j = \varnothing$ for $(1 \leq i, j \leq k)$. An objective function is used to assess the partitioning quality so that objects within a cluster are similar to one another but dissimilar to objects in other clusters.

A centroid–based partitioning technique uses the *centroid* of a cluster, C_i, to represent that cluster. Conceptually, the centroid of a cluster is its center point. The centroid can be defined in various ways such as by the mean or medoid of the objects (or points) assigned to the cluster. The difference between an object $p \in C_i$ and c_i, the representative of the cluster, is measured by $dist(p, c_i)$, where $dist(x, y)$ is the Euclidean distance between two points x and y. The quality of the cluster C_i can be measured by the within–cluster variation, which is the sum of squared error between all objects in C_i and the centroid c_i defined as:

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, c_i)^2 \tag{2.5}$$

where E is the sum of the squared error for all objects in the data set; p is the point in space representing a given object; and c_i is the centroid of cluster C_i. This objective function tries to make the resulting k clusters as compact and as separate as possible.

Optimizing the within–cluster variation is computationally challenging. In the worst case, we would have to enumerate a number of possibles partitionings that are exponential to the number of clusters, and check the within–cluster variation values. It has been shown that the problem is NP–hard in general Euclidean space even for two clusters. Moreover, the problem is NP–hard for a general number of clusters k even in the 2–D Euclidean space. If the number of clusters k and the dimensionality of the space d are fixed, the problem can be solved in time $O(n^{dk+1}\ log\ n)$ where n is the number of objects. Greedy approaches are used in practice.

This algorithm defines the centroid of a cluster as the mean value of the points within the cluster. It proceeds as follows. First, it randomly selects k of the objects in D, each of which initially represents a cluster mean or center. For each of the remaining objects, an object is assigned to the cluster to which it is most similar, based on the Euclidean distance between the object and the cluster mean. The k–means algorithm then iteratively improves the within–cluster variation. For each cluster, it computes the new mean using the objects assigned to the cluster in the previous iteration. All the objects are then reassigned using the updated means as the new cluster centers. The iterations continue until the assignment is stable, that is, the clusters formed in the current round are the same size as those formed in the previous round. The k–means algorithm is described in algorithm 1.

In figure 2.5 is depicted the process with three different types of objects. The algorithm start creating three randomly centroids or cluster centers that we called it seeds. Then the seeds begin to the center of the groups by computing the mean among the data vectors.

We will try to cluster products from transactional data as described in figure 2.5 then we will determine a belonging factor to each cluster based on previous transactions as depicted in figure 2.6, where a, b, c are the belonging factors to each cluster, as shown in 2.3.

Algorithm 1: The k–means algorithm.

```
Input:
    $k$: the number of clusters
    $D$: a data set containing $n$ objects
Output:
    A set of $k$ clusters.
Method:
1   Arbitrary choose $k$ objects from $D$ as the initial cluster centers
2   repeat
3       Assign each object to the cluster to which the object is the most
        similar, based on the mean value of the objects in the cluster
4       Update the cluster means, that is, calculate the mean value of the
        objects for each cluster
5   until no change
6   return
```

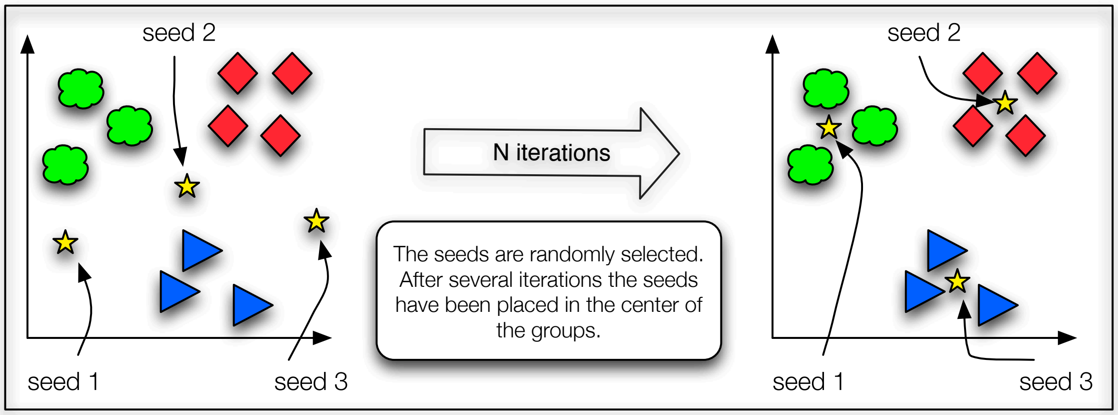

seed 2
seed 1
seed 3
N iterations
The seeds are randomly selected. After several iterations the seeds have been placed in the center of the groups.
seed 2
seed 1
seed 3

Figure 2.5: k–means algorithm.

2.5.5.2 Self–Organizing Features Maps (SOM)

There is a type of neural network model called the self–organizing map (SOM) that can be employed for a clustering task. Proposed by Kohonen [73], the SOM was originally used for image and sound, and recently applied to clustering people. Like other neural network models, the SOM has an input layer and an output layer. Each unit (or cluster) in the output layer is connected to units (or attributes) in the input layer. The strength of this connection is measured by a weight.

The SOM is fundamentally different from other neural network models in that its goal

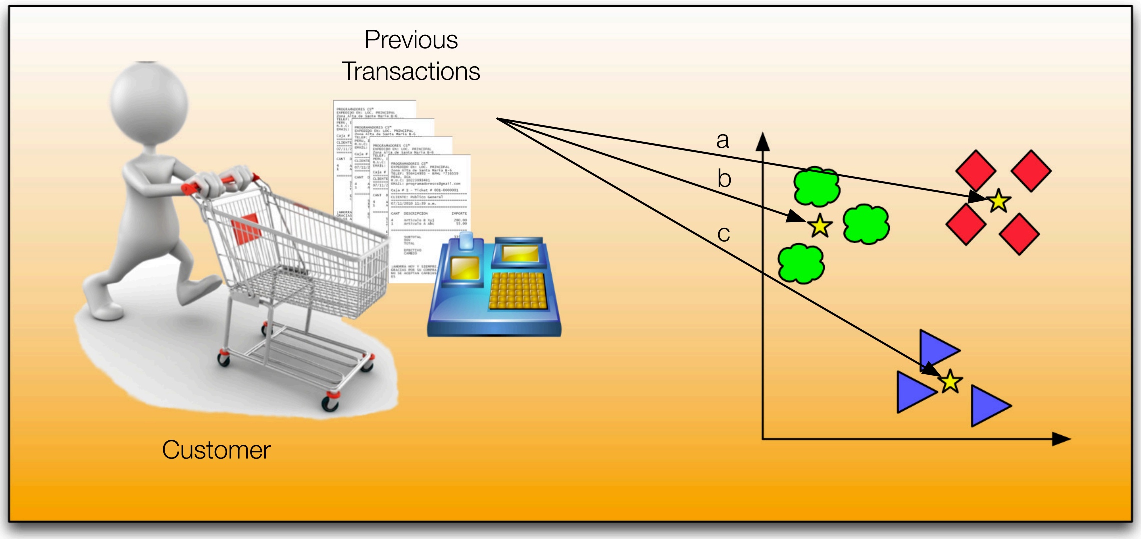


Figure 2.6: Customer belonging to k–means clusters.

is to identify no prespecified "dependent variable" for the output layer. The SOM is looking for unknown patterns in the data. Using the terminology of machine learning, the SOM is developed for unsupervised learning.

SOM represents the result of a vector quantization process as describe [111]. The SOFM map the input space into a bi–dimensional array of nodes also called neurons. The array's lattice can be defined rectangular or hexagonal. Every neuron is an array of similar dimensionality than data vectors. Let's call $m_i \in \Re^n$ the neuron i from a SOM then the components of every neuron are called the synaptic weights. Figure 2.7 depict how these neurons and weights are structured.

To begin with the SOFM algorithm all neurons must be initialized. This process is performed creating random-synaptic weights for every neuron. Afterwards, we commence the learning or training phase of the SOM. Let $x_i \in \Re^n$ be an input data vector, we present x_i to the network and using a metric, we determine the most similar neuron (center of excitation, winner neuron, best matching unit (BMU) or centroid). This process is performed for every input example x. Once all data vectors have been presented to the network, we say an *epoch* has finished. Next, we must begin another *epoch*, this is done by presenting all data vectors to the network again. Finally, after many epochs, we obtain convergence on the SOM and we can finish the training phase.

The best matching unit (BMU) is obtained from equation 2.6, if the SOFM has N neurons, then $m_c(t)$ is the winner neuron defined as the BMU in the whole network for the example x_i in the epoch

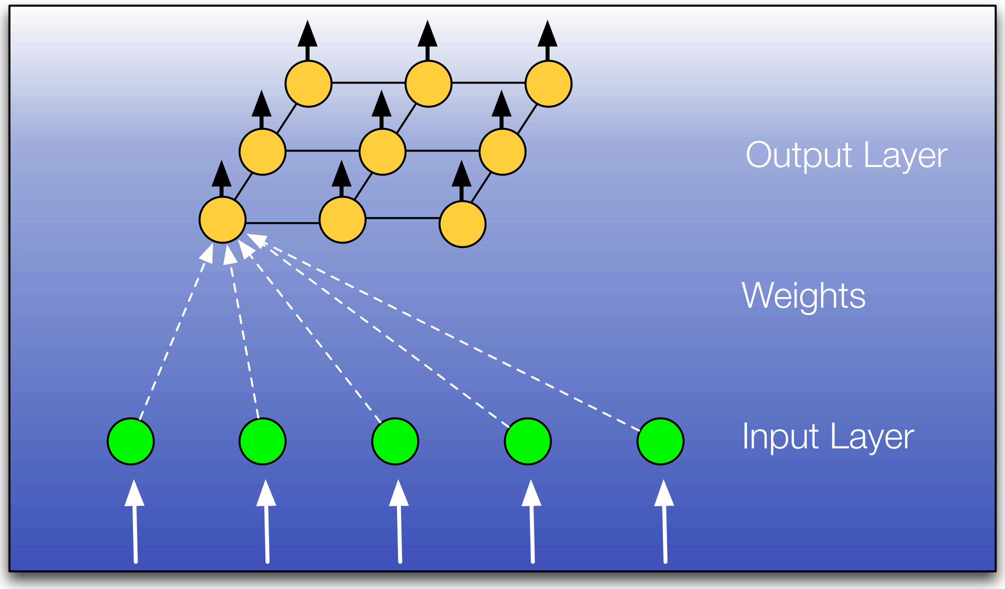


Figure 2.7: Architecture of a SOM.

$$\|x_i - m_c(t)\| = min\|x_i - m_j(t)\| \text{ or } c = argmin\|x_i - m_j(t)\| \quad \forall j = 1, \ldots, N \tag{2.6}$$

Once the winner neuron (BMU) has been identified, we activate it to actually learn the example vector. To do so, we use a learning functions as shown in equation 2.7. An important factor is that the learning function alters the synaptic weights of the BMU but, it also alters the weights of the surrounding neurons in a lesser degree. This way, the BMU is moved towards the data example and the surrounding neurons also are moved but lesser than the BMU. This effect depends on the distance from the BMU (or centroid) as shown in equation 2.7 and is transmitted to all neurons in the network.

$$m_j(t+1) = m_j(t) + h_{cj}(t) * (x_i(t) - m_j(t)) \quad \forall j = 1, \ldots, N \tag{2.7}$$

The equation 2.7 show the basic weight modification algorithm, where t is an integer and represent the iteration, and the h_{cj} is called "neighborhood kernel". It is a function defined over the lattice points and usually $h_{cj} = h(\|rc - rj\|, t)$, where $r_c, r_j \in \Re$ are the radius between the BMU and another neuron in the array. The expression h_{cj} is so that when $\|r_c - r_j\|$ is increased, $h_{ci} \to 0$. Depending on the points chosen by hci it is possible to define different notions of neighborhood. Therefore, we are able to define diverse topologies of the SOM (see Fig. 2.7), such as:

- Open topology: The idea is to maintain the bi–dimensional space geometry when producing the learning. The edges of the map are not connected to other neurons, producing an abrupt end of the learning. This effect is more notorious when the BMU is closer to the edges of the map.

- Tape/cilindrical topology: Using this topology the idea is to connect two borders of the bi–dimensional map. Thus, the learning effect continues to the opposite side of the map, although, the other sides of the map are disconnected.
- Toroidal topology: In this topology we connect all borders of the bi-dimensional grid. This way we never finish the learning effect in either direction. We always transmit the effect to all neurons in the map in a smooth way. We say that this topology helps to maintain the continuity of the space.

The previous topologies presented are depicted in figure 2.8.

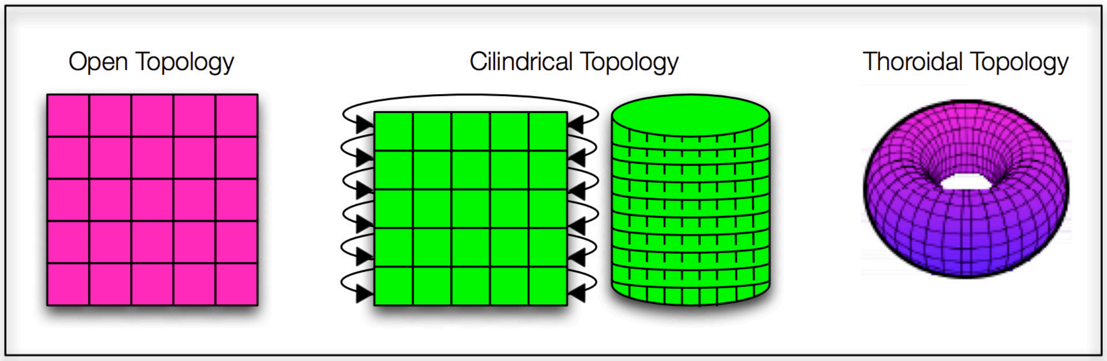


Figure 2.8: SOM possible topologies.

2.5.6 Number of Clusters

Determining the appropriate number of cluster is one of the most difficult problems in clustering. Usually, the criteria chosen tends to be subjective. For example, the relative sizes of the clusters should be large enough to be managerially meaningful. The clusters with few elements (customers) may be treated as outliers and ignored.

The methods for determining the number of clusters depend on the clustering algorithm being used and there are several criteria available. However, Milligan [89] showed that the procedure by Calinski [26] performed is the best among 30 different criteria. [26] suggested the following criterion to determine the number of clusters:

$$G(k) = \frac{(n-k)(T-W)}{(k-1)W} \tag{2.8}$$

where k is the number of clusters, n is the number of customers, W is the square sum of the distances of the customers to the center of its cluster, and T is the square sum of the differences of each customer to the average customer, essentially, the center of the full data. The optimal number of cluster can be determined by selecting k which returns the maximum value for $G(k)$, because in that case, W, or the distances between customers and the center of their clusters, is relatively small compared to T, the distances between customers and the center of the entire data.

2.5.7 Interpreting the results

The results of a cluster analysis are interpreted by examining the means for each cluster of the clustering variables, and also examining the means of any other variable, i.e., "discrimination variables", not included in the clustering routine. For instance, we may cluster based on benefits sought but have a host of other variables, for example demographics, that we use for interpretation.

Interpreting the clusters is a subjective but interesting task that often adds insight into the nature of the market. Also, the interpretations are subjective and make use of both the clustering and discrimination variables. Note also there are clear managerial implications in that companies with particular strengths could plausibly target one of these clusters, but probably not all of them (at least with the same product).

2.6 Market Basket Analysis

Market basket analysis (MBA) tries to examine the products customers usually buy together, and using this information to promote products trying to induce an up-selling or a cross-selling. The first one tries that customer purchase more expensive items, upgrades, or other add-ons in order to make a more profitable sale. The second one, try to sell an additional product or service to an existing customer. In this chapter we will discuss key concepts of *confidence*, *support*, and *lift* applied to market basket analysis.

Blattberg et al. [19] shows that marketing researchers have been interested in studying product affinity for a long time. Because the output of a market basket analysis is a series of rules. These rules are used to improve the efficiency of marketing strategies and tactics. Retail stores learn from the analysis which products are purchased at the same time or in a particular sequence.

Market basket provides valuable information for firms to develop various marketing strategies and tactics. First, association rules from a market basket analysis can be used for a supermarket to manage its space. It may stock the associated items close together such that customers would not forget to purchase both items. On the other hand, it may stock the associated items far apart such that consumers would have to spend more time browsing aisle by aisle.

Second, market basket can be used to design different promotional strategies. It will provide an idea to develop a coupon program where customers purchasing item A get the discount coupon for item B.

Third, market basket, can be very helpful to marketers for selecting cross–selling items. For instance, a customer purchase item A, and the salesperson offers a cross–sell product B.

2.6.1 Definition

Market basket analysis, according to [18] does not refer to a single technique; it refers to a set of business problems related to understanding point–of–sale transaction data. The most common technique is association rules, and will be described in section 2.6.3.

2.6.2 Market Basket Data

Market basket data belong to the family of transactional data (described in section 2.2.4). It is, basically, a description and a relationship between three entities:

Customers It refers, to the person that makes a purchase in a particular store or supermarket. This information is not always available, for example if the customer is not logged in the web site, or if the customer do not belong to a loyalty club.

Orders Are the item set purchased in a single purchase event by a customer. This might correspond to a customer ordering several products on a web site, or a customer purchasing a basket of groceries. This includes the total amount of the purchase, the total amount, additional shipping charges, payment type, etc.

Item It refers, to the description of a particular product, for example, SKU, product family, price, etc. Usually includes information that might prove valuable for analysis like product hierarchy.

Blattberg [19] describes that the input for a market basket analysis is customer–level transactions data, although it is not necessary that each customer be explicitly identified. For example, grocery stores record each customer's transaction data, with their scanning device even through they do not know the customer's name, address, phone number, e–mail address, etc. For each transaction they know the date, the cashier identifier, items purchased, prices of each item, coupons redeemed. Table 2.2 shows an example of the information contain in a groceries' store database. There are five transactions and each transaction consists of a set of items. The main focus of market basket analysis is the set of items purchased for each transaction. From this transaction data, market basket analysis provides a series of association rules where we infer which items are purchased together.

Transactions	**Items purchased**
1	Milk, orange juice, ice cream, beer, soap
2	Milk, ice cream, beer
3	Milk, orange juice, detergent
4	Milk, ice cream, pizza
5	Milk, orange juice, soap

Table 2.2: Market basket data from a grocery store.

Each association rule consists of an antecedent and a consequent. For instance, consider the association rule, "if a consumer purchases item A, also tends to purchase item B". In this example, A is the *antecedent* while B is the *consequent*.

2.6.3 Association Rules

Association rules have been widely studied, since the paper by Agrawal [6], within the field of knowledge discovery [7, 21, 86] and is usually called the market basket problem.

2.6.3.1 Definition

Let $I = \{I_1, I_2, \ldots, I_m\}$ be an item(product) set. Let D be a set of a database transactions where each transaction T is a nonempty item set such that $T \subset I$. Each transaction is associated with an identifier, called *transaction id* or TID. Let F be a set of items. A transaction T is said to contain F if $F \subset T$.

An *association rule* is an implication of the form $A \Rightarrow B$, where $A \subset I$, $B \subset I$, $A \neq \emptyset$, $B \neq \emptyset$, and $A \cap B = \emptyset$

2.6.4 Deriving Association Rules

We are going to intuitively obtain purchase patterns, from transactional data of table 2.2. If we look the data, we can check that every purchase include milk, also milk and orange juice are purchased together in three out of five transactions. From this observation we can say that there is a cross–selling possibility between milk and orange juice. Ice cream and beer are purchased together in two out of five transactions. If we continue doing this process, we can formulate an association rule between orange juice and soap, but we are interested in selecting only "interesting" rules, to do so we need some criteria or a metric to quantify the quality of the rule obtained [16]. Hence, researchers have proposed many different metrics. The three more popular criteria evaluating the quality of an association rule are, support, confidence and lift. Next, each one of these is describe:

Support is the percentage of transactions containing a particular combination of items relative to the total number of transactions in the data set. Support for an individual item A can be interpreted as the probability a transaction contains item A, or $P(A)$. However, we are interested in associations with multiple items, so the support for the combination A and B would be $P(AB)$. For example, considering the association rule "if milk then beer" from table 2.2. Support measure how often milk and beer are purchased together. They are purchased together two out of five transactions. So, the support for the association rule is 40%. Support for multiple items, can be interpreted as a joint probability.

Support has one critical disadvantage measuring the quality of an association rule. In table 2.2 is shown that association rule "if beer then milk" has support of 40%. However, as every customer buys milk, and itself has a high support, the support for any element with milk will be high. So, we need another measure to avoid this fact.

Confidence measures how much the consequent (products or items) is dependent on the antecedent (products or items). In other words, confidence is the conditional probability of the consequent given the antecedent, $P(A|B)$. For instance, the confidence for the association rule "if ice cream then beer" is 66% since three transactions contain the antecedent (ice cream) and two among the three transactions contain the consequent (beer). This means that given that the baskets containing ice cream is selected, there is 66% chance that the same basket also contains beer. Confidence, unlike the support, is asymmetric. For example, confidence of "if beer then ice cream" is 100% while the confidence of "if ice cream then beer" is 66%.

The law of conditional probability states that $P(B|A) = P(AB)/P(A)$. That is, confidence is equal to the support of the association rule divided by the probability or the support of the antecedent. For instance, the support of an association rule "if ice cream then beer" is 40% (two of five transactions) while the support of the probability of ice cream is 60% (three out of five). The confidence is $(40\%/60\%) = 66\%$.

Confidence just like support, has some problems. Consider a rule "if ice cream then orange juice". Its confidence or $P(B|A)$ is 33% so you may think it is an interesting rule, but there is 60% chance (for example $P(B) = 60\%$) that a randomly chosen transaction contains orange juice. So, ice cream is not a powerful antecedent for identifying an orange juice purchase. Its confidence is lower than a random chance of identifying an orange juice purchase.

Lift is a measure to surpass the problems with support and confidence. Let's suppose a general association rule "if A then B". The lift for the rule is defined as $P(B|A)/P(B)$ or $P(AB)/(P(A)P(B))$. Lift is symmetric in that the lift for "if A then B" is equal to "if B then A".

$P(B)$ is the probability that a randomly chosen transaction contains item B. It is an unconditional probability of purchasing item B regardless of other items purchased (also known as "expected confidence"). Lift is said to measure the difference between the confidence of a rule and the expected confidence. For instance, the lift of an association rule "if ice cream then beer" is 1.67 because the expected confidence is 40% and the confidence is 67%. This means that consumers who purchase ice cream are 1.67 times more likely to purchase beer than randomly chosen customers. A larger lift means more interesting rules.

Lift equal to 1 has a special meaning. We know that $P(AB) = P(A)P(B)$ if A and B are independent. Lift greater than 1 indicates that the item A is independent and the item B tend to occur together more often than by random chance. On the other hand, if lift is lower than 1 indicates that the item A and B are purchased together less likely than would be predicted by random chance.

In summary, these are the three most popular criteria for evaluating association rules in market basket analysis, defined as follows:

$$Support = P(A \cup B) \tag{2.9}$$
$$Confidence = P(B|A) \tag{2.10}$$
$$Lift = P(B|A)/P(B) \tag{2.11}$$

Each criteria has its advantages and disadvantages. In general, rules with high support, high confidence and high lift are preferred because they might indicate interesting rules. Rules with very low lift are interesting because implies that two product "repel" each other. For example, pepsi and coke, they usually tend not to be in the same basket. Knowing which products tend to "repel" each other gives and idea of products that should not be promoted together.

2.6.5 Frequent Itemset Mining Methods

In this subsection we will present the methods for mining the simplest form of frequent patterns such as those described in previously in section 2.6.3. We begin our description, with Apriori algorithm (2.6.5.1), the basic algorithm for finding frequent itemsets. Then, we show how to generate strong association rules from frequent itemsets (2.6.5.2).

2.6.5.1 Apriori Algorithm

Apriori is an algorithm proposed by Agrawal et al. [7] for mining frequent itemsets for boolean association rules. The name is based in the fact that the algorithm uses prior knowledge of frequent itemset properties. Apriori algorithm uses an iterative approach known as level–wise search, where k–itemsets are used to explore $(k+1)$–itemsets. First, the set of frequent 1–itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The result is named as L_1. Next, L_1 is used to find L_2, the set of frequent 2–itemsets, which is used to find L_3, and so on, until there is no more frequent k–itemsets to be discovered. Is necessary a full scan of the database to discover each set of L_k.

The apriori algorithm used a two step process, **join** and **prune**. *Join* step finds L_k from a set of candidate k–itemsets generated by joining L_{k-1} with itself, generating a set denoted by C_k, is important to note that C_k is at least equal or bigger than L_k. The *prune* step is necessary to remove members from C_k that may not be frequent. To do so, a database scan is perform to determine the count of each candidate of C_k that would result in the determination of L_k. As C_k can be huge, a property known as *Apriori property* is used. This property says: *All nonempty subsets of a frequent itemset must also be frequent.* Property is used as follows, any $(k-1)$–itemset that is not frequent cannot be a subset of a frequent k–itemset. Hence, if any $(k-1)$–subset of a candidate k–itemset is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k. This subset testing can be done quickly by maintaining a hash tree of all frequent itemsets.

In algorithm 2 is depicted the Apriori algorithm.

Algorithm 2: The Apriori algorithm.

```
Input:
    $D$: a database of transactions.
    $min\_sup$: the minimum support count threshold
Output:
    $L$: frequent itemsets in $D$.
Method:
1   $L_1 = find\_frequent\_1\text{–}itemsets(D)$
2   for $(k = 2; L_{k-1} \neq \phi; k = k + 1)$ do
3       $C_k = \text{apriori\_gen}(L_{k-1})$
4       foreach transaction $t \in D$ do
5           $C_t = \text{subset}(C_k, t)$
6           foreach candidate $c \in C_t$ do
7               $c.count = c.count + 1$
8           end
9       end
10      $L_k = \{c \in C_k \mid c.count \geq min\_sup\}$
11  end
12  return $L = \cup_k L_k$
```

Definitions:

- l_1 and l_2 are itemsets in L_{k-1}.
- $l_i[j]$, refers to the jth item in l_i (for example $l_1[k-2]$ refers to the second to the last item in l_1).

Algorithm 3: Procedures : apriori_gen.

```
apriori_gen($L_{k-1}$: frequent $(k-1)$–itemsets)
for itemset $l_1 \in L_{k-1}$ do
    for itemset $l_2 \in L_{k-1}$ do
        if $(l_1[1] = l_2[1])$ & ... & $(l_1[k-2] = l_2[k-2])$ & $(l_1[k-1] < l_2[k-1])$
        then
            $c = l_1 \bowtie l_2$        // Join step
            if has_infrequent_subset($c, L_{k-1}$) then
                delete $c$      // Prune step
            else
                add $c$ to $C_k$
    end
end
return $C_k$
```

Algorithm 4: Procedures : has_infrequent_subset.

```
has_infrequent_subset($c$: candidate $k$–itemsets
                      $L_{k-1}$: frequent $(k-1)$–itemsets.)
for $(k-1)$–subset $s$ of $C$ do
    if $s \notin L_{k-1}$ then
        return TRUE
end
return FALSE
```

2.6.5.2 Generating Association Rules from Frequent Itemsets

Now, we have the frequent itemsets from the transactions database D, it is time to generate strong association rules from them. We understand strong as rules that satisfy both minimum support and minimum confidence. These can be done using equation 2.10, for confidence. This equation says:

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support\_count(A \cup B)}{support\_count(A)} \tag{2.12}$$

The conditional probability is expressed in terms of itemset support count, where $support\_count(A \cup B)$ is the number of transactions containing the itemsets $A \cup B$, and $support\_count(A)$ is the number of transactions containing the itemset A. Based on this we can generate the association rules as follows:

- For each frequent itemset l, generate all nonempty subsets of l.
- For every nonempty subset s of l, output the rule "$s \Rightarrow (l - s)$" if $\frac{support\_count(l)}{support\_count(s)} \geq min\_conf$, where min_conf is the minimum confidence threshold.

With this process we can obtain the association rules that are important, meaningful and relevant for the retail. A relevant issue about association rules is that are hard to analyze because rules are not grouped and require an expert analysis to be fully understood.

2.7 Social Network Analysis

This section is based on previous work realized by [28, 36, 93].

The concept of social network analysis, were introduced in 1967. Stanley Milgram realiced an experiment trying to understand how people are connected to others. The experiment consists in asking people to forward a package to any of their acquaintances who they thought might be able to reach specific target individual [88]. Milgram found that most people were connected by six acquaintances. Backstrom et al. [12] made an experiment using the facebook network of active users discovering that most people were connected by four acquaintances.

Research on networks, typically, consists in analize observations about the structure and the models giving rise to such structures. This approach aims to discover structural properties or finding patterns [28], such as heavy–tailed degree distributions [22, 41], statistical properties [80], local clustering of edges [81], study of network community structure [45, 50]. Just to name a few.

2.7.1 Networks and Graph Theory

This section will develop basic ideas about graph theory and the study of network structure.

Graphs, nodes and edges According to [28], a graph is a way of specifying relationships among a collection of items, a set of nodes, pairs of which might be connected by edges. This definition applies to a wide array of disciplines. For example, computer networks consists of routers/computers (*nodes*) and the links (*edges*) between them. For example, the graph in Figure 2.9 consists of 5 nodes labelled $1, 2, 3, 4$ and 5, with 3 connected to each of the other four nodes by edges. We say that two nodes are neighbors if they are connected by an edge.

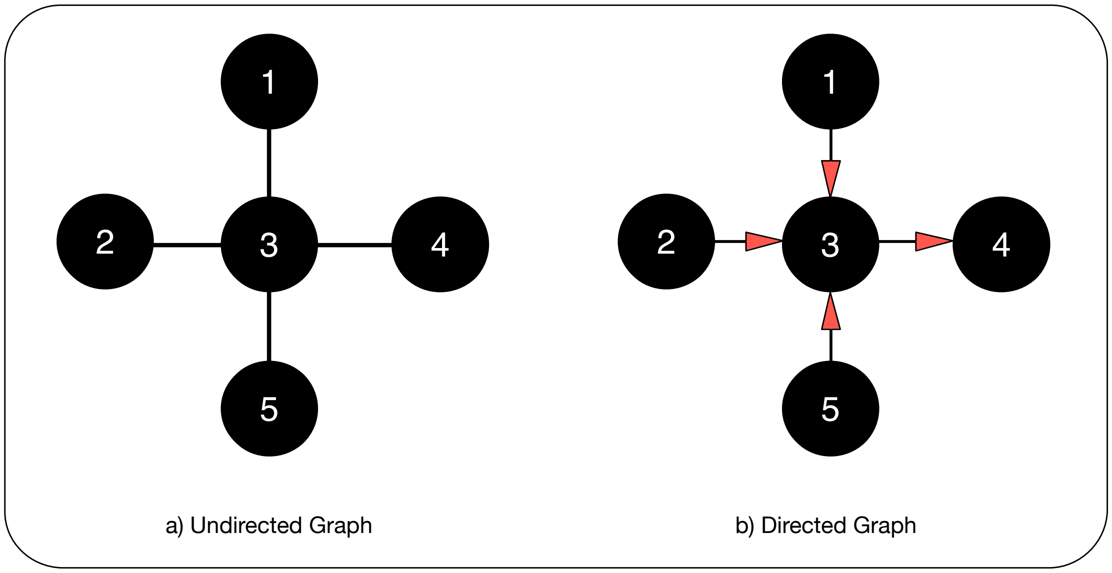


Figure 2.9: Example of graphs: directed and undirected.

In Figure 2.9 a), it is possible to think the relationship between two ends of an edge as being symmetric; the edge simply connects them to each other. In many settings, however, asymmetric relationships are useful -for example, that 2 points to 3 but not vice versa. For this purpose, a *directed graph* is defined as a set of nodes with a set of directed edges; each directed edge is a link from one node to another, in this case direction is important. Directed graphs are generally drawn as in Figure 2.9(b), with edges represented by arrows. If it wants to emphasize that a graph is not directed, can refer to it as an *undirected graph.*

Mathematically, we represent a network by a graph (V, g) which consists of a set of nodes $V = \{1, \ldots, n\}$ and a $n \times n$ matrix $g = [g_{ij}]_{i,j \in V}$ (referred to as an adjacency matrix), where $g_{ij} \in \{0, 1\}$ represents the availability of an edge from node i to node j. The edge weight $g_{ij} > 0$ can also take on non-binary values, representing the intensity of the interaction, in which case we refer to (V, g) as a weighted graph. We refer to a graph as a *directed graph* if $g_{ij} \neq g_{ji}$ and an *undirected graph* if $g_{ij} = g_{ji}\ \forall i, j \in V$.

Another representation of a graph is given by (V, E), where E is the set of edges in the network. For *directed graphs*, E is the set of directed edges, i.e., $(i, j) \in E$ and for *undirected graphs* E is the set of undirected edges, i.e., $\{i, j\} \in E$.

Also, two nodes are said to be *adjacent*, *neighbors*, or *connected* if there exist an edge between them. If all k nodes in the graph are adjacent, the graph is said to be *k-complete.* A graph is simple, i.e. loop-less and lacks multiple edges, if there is at most one edge between each pair of nodes and no node is neighbor with itself.

A *walk* is an ordered set of alternating nodes and edges that starts in one node i and ends in another node j. If the walk only transverses each node at most once, it is called a *path*. A k-*cycle* is a path where the first and last nodes are the same, and the path contains k edges. A graph is *connected*, if there exists a path between any given pair of nodes. The *shortest path* between two nodes is the *geodesic* and the longest geodesic is the diameter of the graph.

A graph without cycles is called a *tree* (or a forest if unconnected). A *subgraph*, G', of a graph G contains all edges that connect a subset of the node set, i.e. $V'(G) \subset V(G)$ such that $E'(G) \subset E(G)$ contains all edges connecting the nodes in $V'(G)$. One says that the edge set is spanned by the set of nodes. Two subgraphs are therefore disjoint and not connected. A k-*clique* is a k-complete sub-graph.

2.7.2 Metrics in social network analysis

Within graph theory and network analysis, there are various measures of the centrality of a vertex within a graph that determine the relative importance of a vertex within the graph. There are four measures of centrality that are widely used in network analysis: degree centrality, closeness, betweenness, and eigenvector centrality.

Degree centrality

The most intuitive measure of centrality of a vertex into a network is called degree centrality. Given a graph $G = (V, E)$ represented by means of its adjacency matrix A, in which a given entry $A_{ij} = 1$ if and only if i and j are connected by an edge, and $A_{ij} = 0$ otherwise, the *degree centrality* $C_D(v_i)$ of a vertex $v_i \in V$ is defined as:

$$C_D(v_i) = d(v_i) = \sum_j A_{ij} \tag{2.13}$$

The idea behind the degree centrality is that the importance of a vertex is determined by the number of vertices adjacent to it, i.e. the larger their degree, the more important the vertex is.

Even though, in real world networks only a small number of vertices have high degrees, the degree centrality is a rough measure but it is adopted very often because of the low computational cost required for its computation. There exists a *normalized* version of the degree centrality, defined as follows

$$C'_D(v_i) = \frac{d(v_i)}{n-1} \tag{2.14}$$

where n represents the number of the vertices in the network.

Closeness centrality

A more accurate measure of centrality of a vertex is represented by the *closeness centrality* [114]. The closeness centrality relies on the concept of *average distance*, defined as:

$$D_{avg}(v_i) = \frac{1}{n-1} \sum_{j \neq i}^{i} g(v_i, v_j) \tag{2.15}$$

where $g(v_i, v_j)$ represents the geodesic distance between vertices v_i and v_j.

The closeness centrality $C_C(v_i)$ of a vertex v_i is defined as

$$C_C(v_i) = \frac{1}{n-1} \sum_{j \neq i}^{i} g(v_i, v_j) \tag{2.16}$$

In practice, the closeness centrality calculates the importance of a vertex on how close the give vertex is to the other vertices. Central vertices, with respect to this measure, are important as they can reach the whole network more quickly than non-central vertices. Different generalizations of this measures for weighted and disconnected graphs have been proposed in [99].

Betweenness centrality

A more complex measure of centrality is the betweenness centrality [47,48]. It relies on the concept of shortest paths. In detail, in order to compute the betweenness centrality of a vertex, it is necessary to count the number of shortest paths that pass across the given vertex.

The betweenness centrality $C_B(v_i)$ of a vertex v_i is computed as

$$C_B(v_i) = \sum_{v_s \neq v_i \neq v_t \in V} \frac{\sigma_{st}(v_i)}{\sigma_{st}} \tag{2.17}$$

where σ_{st} is the number of shortest paths between vertices v_s and v_t and $\sigma_{st}(v_i)$ is the number of shortest paths between v_s and v_t that pass through v_i. Vertices with high values of betweenness centrality are important because maintain an efficient way of communication inside a network and foster the information diffusion.

Eigenvector centrality

Another way to assign the centrality to a vertex is based of the idea that if a vertex has many central neighbors, it should be central as well. This measure is called eigenvector centrality and establishes that the importance of a vertex is determined by the importance of its neighbors. The eigenvector centrality $C_E(v_i)$ of a given vertex v_i is

$$C_E(v_i) \propto \sum_{v_j \in N_i} A_{ij} C_E(v_j) \tag{2.18}$$

where N_i is the neighborhood of the vertex v_i, being $x \propto Ax$ that implies $Ax = \lambda x$. The centrality corresponds to the top eigenvector of the adjacency matrix A.

For simplicity, we will compute Degree, Betweenness and Closeness Centrality for all networks in this work. However, results for every node of these networks are not shown in this thesis. Results are available online[6]

2.8 Overlapping community detection

Detecting clusters or communities in real-world graphs such as large social networks, web graphs, and biological networks is a problem of considerable practical interest that has received a great deal of attention [33, 46, 50, 56, 67].

The first problem in graph clustering is to look for a quantitative definition of community. According to Fortunato [45] there is no universally accepted definition. As a matter of fact, the definition often depends on the specific system and/or application one has in mind. From intuition we get the notion that there must be more edges *inside* the community than edges linking vertices of the community with the rest of the graph [50]. This is the reference guideline at the basis of most community definitions. But many alternative recipes are compatible with it. Moreover, in most cases, communities are algorithmically defined, i.e. they are just the final product of the algorithm, without a precise a priori definition.

The problem can be mathematically defined as follows:

[6]http://users.dcc.uchile.cl/ ividela/resultados/calculos.zip

Given a network or graph $G = \{E, V\}$, where V is a set of n nodes and E is a set of m edges. For dense graphs $m = O(n^2)$, but for sparse networks $m = O(n)$. The network structure is determined by the $n \times n$ adjacency matrix A for unweighted networks and weight matrix W for weighted networks. Each element A_{ij} of A is equal to 1 if there is an edge connecting nodes i and j; and 0 otherwise. Each element w_{ij} of W takes a non–negative real value representing strength of connection between nodes i and j.

In the case of overlapping community detection, the set of clusters found is called a *cover* $C = \{c_1, c_2, \dots, c_k\}$ [77], in which a node may belong to more than one cluster. Each node i associates with a community according to a belonging factor (i.e., soft assignment or membership) $[a_{i1}, a_{i2}, \dots, a_{ik}]$ [95], in which a_{ic} is a measure of the strength of association between node i and cluster c. Without loss of generality, the following constraints are assumed to be satisfied

$$0 \leq a_{ic} \leq 1 \,\forall i \in V,\, \forall c \in C, \tag{2.19}$$

$$\sum_{k=1}^{|C|} a_{ic} = 1, \tag{2.20}$$

where $|C|$ is the number of clusters.

In general, algorithms produce results that are composed of one of two types of assignments, *crisp* (non-fuzzy) assignment or *fuzzy* assignment [54]. With *crisp* assignment, each node belongs to one or more communities with *equal* strength. The relationship between a node and a cluster is *binary*. That is, a node i either belongs to cluster c or does not. With fuzzy assignment, each node is associated with communities in proportion to a belonging factor. With a threshold, a *fuzzy* assignment can be easily converted to a *crisp* assignment. Typically, a detection algorithm outputs *crisp* community assignments.

2.8.1 Overlapping Community Detection Algorithms

Huge number of algorithms have been developed using a variety of methods; these vary in their effectiveness and time performance for different types of networks. In this section, algorithms for overlapping community detection are reviewed and categorized into five classes which reflect how communities are identified.

2.8.1.1 Clique Percolation

The clique percolation algorithm [101] (CPM) detects communities based on the assumption that a community or k-community is a set of nodes which can be reached through a series of adjacent k-cliques (a k-clique is set of k nodes, which are all connected to each other), where two k-cliques are adjacent if they share $(k - 1)$ nodes. The algorithms begins by identifying all cliques of size k in a network. Afterward, the method build k-communities from k-cliques found. Since a vertex can be in multiple k-cliques simultaneously, overlap between communities is possible. Empirically, $k = 3$ or 4 has been show to give the best results. CFinder[7] is an implementation of CPM, whose time complexity is polynomial in many applications. Despite conceptual simplicity, one may argue that CPM-like algorithms are more like pattern matching rather than finding communities since they aim to find specific, localized structures in a network.

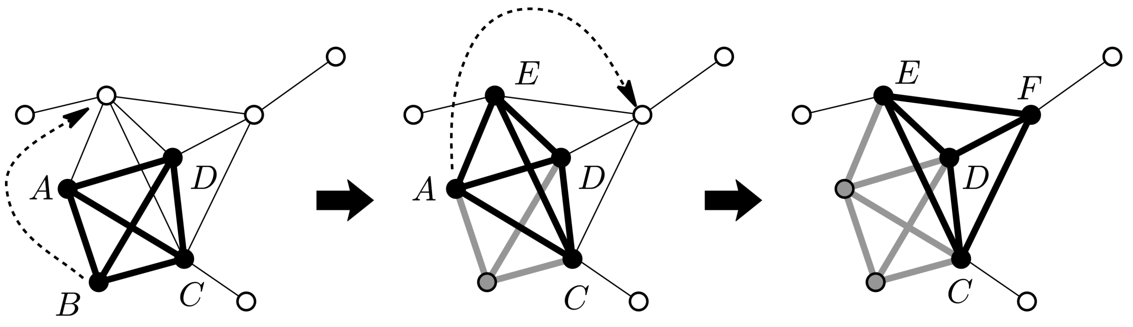


Figure 2.10: Clique Percolation [20]

Figure 2.10 shows the iterative procedure to detected overlapping communities using Clique Percolation methods.

2.8.1.2 Line Graph and Link Partitioning

The idea of partitioning links instead of nodes to discover community structure has also been explored. A node in the original graph is called overlapping if links connected to it are put in more than one cluster. In [8], links are partitioned via hierarchical clustering of edge similarity. Given a pair of links e_{ik} and e_{kj} incident on a node k, a similarity can be computed via the Jaccard Index defined in Equation 2.21.

$$S(e_{ik}, e_{kj}) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|} \tag{2.21}$$

Where N_i is the neighborhood of node i including i. Single-linkage hierarchical clustering is then used to build a link dendrogram. Cutting this dendrogram at some threshold yields to linked communities. The time complexity is $O(nk_{max}^2)$, where k_{max} is the maximum node degree in the network. Although, the link partitioning for overlapping detection seems conceptually natural, there is no guarantee that it provides higher quality detection than node based detection does [45] because these algorithms also rely on an ambiguous definition of community.

[7] *www.cfinder.org* [online: accessed 06-12-2013]

2.8.1.3 Local Expansion and Optimization

Algorithms utilizing local expansion and optimization are based on growing a natural community [77] or a partial community. Most of them rely on a local benefit function that characterizes the quality of a densely connected group of nodes. Baumes et al. [14, 15] proposed a two-step process. First, the algorithm *RankRemoval* is used to rank nodes according to some criterion. Then, the process iteratively removes highly ranked nodes until small, disjoint cluster cores are formed. These cores serve as seed communities for the second step of the process, *Iterative Scan* (IS), that expands the cores by adding or removing nodes until a local density function cannot be improved. The proposed density function can be formally given as

$$f(c) = \frac{w_{in}^c}{w_{in}^c + w_{out}^c} \tag{2.22}$$

where w_{in}^c and w_{out}^c are the total internal and external weight of the community c. The worst-case running time is $O(n^2)$. The quality of discovered communities depends on the quality of seeds.

OSLOM [8] [79] is based on the local optimization of a fitness function. This function expresses the statistical significance of clusters with regard to global null model (i.e., the random graph generated by the configuration model). OSLOM can be used alone or as a refinement procedure of partitions/covers delivered by other techniques.

OSLOM consists of three phases:

- First, it looks for significant clusters, until convergence is reached
- Second, it analyses the resulting set of clusters, trying to detect their internal structure or possible unions
- Third, it detects found clusters' hierarchical structure

iLCD [9] [27] is capable to detect communities taking network dynamics into account. Given a set of edges created at some time step, iLCD updates the existing communities by adding a new node if the following rules are satisfied:

- The node is able to access easily (in two step or lesser) to most of the community (at least as much as the other nodes of the community)

[8] *www.oslom.org* [online: accessed 06-12-2013]

[9] *http://cazabetremy.fr/Cazabet_remy/iLCD.html* [online: accessed 06-12-2013]

- The node has a *robust access* to other nodes; which means a node can be reached by at least two paths of length two or less.

The complexity of iLCD is $O(nk^2)$ in general, whose precise quantity depends on community structures and its parameters.

First, we need to define the input of the iLCD (intrinsic Longitudinal Community Detection) algorithm. Due to the longitudinal analysis, we will use a list of edges, ordered by their creation time, those edges could correspond to links creation among existing nodes or could also imply the creation of a new node. As some edges creations can be simultaneous (think about the publication of several articles in a given journal issue), we will use ordered sets of edges, where edges of a given set are created at the same time.

More formally, let's note $G = (V, E)$ the graph that is dynamically built and $C =< C_k >$ the set of communities that is dynamically built. Initially, G and C are empty. We then define E_{in} the set of edges in input as $E_{in} =< E_t >$ i.e. composed by ordered time-stamped sets of edges. (see Algorithm 5 for pseudo-code)

Algorithm 5: iLCD

```
for each time-stamped set $E_t$ do
    for each edge $(u, v)$ of the set $E_t$ do
        Add $(u, v)$ to $E$. If $u$ or $v$ is not in $V$, add it to $V$
        Determine the updates of existing communities. For each
        community $C_k$ to which $u$ (respectively $v$) belongs, try to integrate $v$
        (resp. $u$ ) to $C_k$
    end
    Update of previous communities
    If $u$ y $v$ do not already belong to the same community, Try to create a
    new community.
    Merge similar communities.
end
```

2.8.1.4 Fuzzy Detection

Fuzzy community detection algorithms quantify the strength of association between all pairs of nodes and communities. In these algorithms, a soft membership vector, or belonging factor [53], is calculated for each node. A drawback of such algorithms is the need to determine the dimensionality k of the membership vector. This value can be either provided as a parameter to the algorithm or calculated from the data.

2.8.1.5 Agent Based and Dynamical Algorithms

COPRA[10] [53], is an algorithm based on the label propagation technique of Raghavan, Albert and Kumara; which is able to detect communities that overlap. Like the original algorithm, vertices have labels that propagate between neighbouring vertices so that community members reach a consensus on their community membership, each node updates its belonging coefficients by averaging the coefficients from all its neighbors at each time step in a synchronous fashion.

COPRA algorithm(Community Overlap Propagation Algorithm) keeps two vectors of vertex labels: *old* y *new*; *old.x* (resp. *new.x*) denotes the previous (resp. updated) label for vertex x.

Each vertex label is a set of pairs (c, b), where c is a community identifier and b is the belonging coefficient. $N(x)$ is the set of neighbours of vertex x (see Algorithm 6 for

[10] *www.cs.bris.ac.uk/~steve/networks/software/copra.html* [online: accessed 06-12-2013]

pseudo-code).

Algorithm 6: COPRA

```
foreach vertex $x$ do
    old.$x$ ← $\{(x, 1)\}$
end
for each vertex $x$ do
    Propagate($x$, old, new).
end

if id(old)=id(new) then
    min ← mc(min, count(new))
. else
    min ← count(new).

if $min \neq oldmin$ then
    old ← new.
    oldmin ← min.
    Repeat from step 2.
foreach vertex $x$ do
    ids ← id(old.$x$).
    foreach $c$ in ids do
        if for some $g$, $(c, g)$ is in coms, $(c, i)$ in sub then
            coms ← coms $-\{(c, g)\} \cup \{(c, g \cup \{x\})\}$.
            sub ← sub $-\{(c, i)\} \cup \{(c, i \cap ids)\}$.
        else
            coms ← coms $\cup\{(c, \{x\})\}$.
            sub ← sub $\cup\{(c, ids)\}$.
    end
end
foreach $(c, i)$ in sub do
    if $i \neq \{\}$ then
        coms ← coms $-(c, g)$.
end
Split disconnected communities in coms.
```

SLPA[11] [133] is a general speaker-listener based information propagation process. It spreads labels between nodes according to pairwise interaction rules. Unlike others algorithms, where a node forgets knowledge gained in the previous iterations, SLPA provides each node with a memory to store received information (labels). The membership strength is interpreted as the probability of observing a label in a node's memory. One advantage of SLPA is that it does not require any knowledge about the number of communities. The time complexity is $O(tm)$, linear in the number of edges m, where t is a predefined maximum number of iterations.

SLPA is an extension of the Label Propagation Algorithm (LPA) proposed by Raghavan, Albet and Kumara. In LPA, each node holds only a single label that is iteratively updated by adopting the majority label in the neighborhood. Disjoint communities are discovered when the algorithm converges. One way to account for overlap

[11]now, SLPA has a new name, GANXiS. https://sites.google.com/site/communitydetectionslpa/ganxis

is to allow each node to possess multiple labels. SLPA follows this idea but applies different dynamics with more general features.

In the dynamic process, we need to determine 1) how to spread nodes' information to others; 2) how to process the information received from others. The critical issue related to both questions is how information should be maintained. A speaker-listener based information propagation process (SLPA) is proposed to mimic human communication behavior.

In SLPA, each node can be a listener or a speaker. The roles are switched depending on whether a node serves as an information provider or information consumer. Typically, a node can hold as many labels as it likes, depending on what it has experienced in the stochastic processes driven by the underlying network structure. A node accumulates knowledge of repeatedly observed labels instead of erasing all but one of them. Moreover, the more a node observes a label, the more likely it will spread this label to other nodes (mimicking people's preference of spreading most frequently discussed opinions).

2.8.2 Evaluation Criteria

Evaluating the quality of a detected partitioning or cover is nontrivial, and extending evaluation measures from disjoint to overlapping communities is rarely straightforward.

In this section we present two well-know quality measures *Normalized Mutual Information* and *Modularity*. In this thesis we will use *modularity* as clustering quality measure because the ground-truth is unknown.

2.8.2.1 Normalized Mutual Information

There are many evaluation criteria in the literature (see [47]), but most of them can be used to compare partitions: a *partition* is a union of subsets which are non-overlapping and which cover the whole set; a *cover* is just a collection of(overlapping) subsets.

Although there is currently no consensus on which is the best measure, information theoretic based measures have received increasing attention for their strong theoretical background. Let us first review some of the very fundamental concepts of information theory [35] and then see how those concepts might be used toward assessing clusterings agreement.

Definition 1. The information entropy of a discrete random variable X, that can take on possible values in its domain $\chi = \{x_1, x_2, \ldots, x_n\}$ is defined by:

$$H(X) = -\sum_{x \in \chi} p(x)\ log(p(x)) \tag{2.23}$$

Definition 2. The mutual information between two random variables X and Y with respective domains χ and Υ is defined by:

$$I(Y,X) = \sum_{x\in\chi}\sum_{y\in\Upsilon} p(y,x)\ log\frac{p(y,x)}{p(x)p(y)} \tag{2.24}$$

The mutual information (see Figure 2.11) is a symmetric measure that quantifies the mutual dependence between two random variables, or the information that X and Y share. It measures how much knowing one of these variables reduces our uncertainty about the other. This property suggests that the mutual information can be used to measure the information shared by two clusterings, and thus, assess their similarity. Lancichinetti et al. [78] has extended the notion of normalized mutual information to account for overlap between communities.

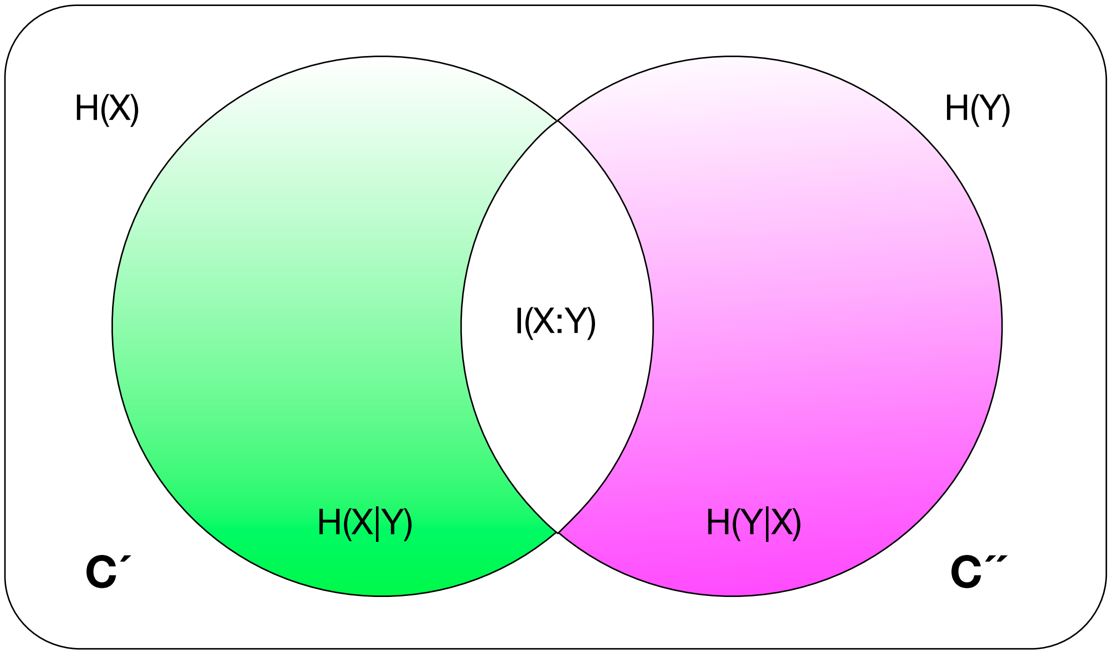


Figure 2.11: Mutual Information

The normalized mutual information is defined as:

$$NMI(X|Y) = \frac{H(X)+H(Y)-H(X,Y)}{(H(X)+H(Y))/2} \tag{2.25}$$

where $H(X)$ ($H(Y)$) is the entropy of the random variable X (Y) associated to the partition C' (C''), whereas $H(X,Y)$ is the joint entropy. This variable is in the range $[0,1]$ and equals 1 only when the two partitions C' and C'' are exactly coincident.

For each node i in cover C', its community membership can be expressed as a binary vector of length $|C'|$ (i.e., the number of clusters in C'). $(x_i)_k = 1$ if node i belongs to the k^{th} cluster C'_k; $(x_i)_k = 0$ otherwise. The k^{th} entry of this vector can be viewed as a random variable X_k, whose probability distribution is given by $P(X_k = 1) = n_k/n$, $P(X_k = 0) = 1 - P(X_k = 1)$, where $n_k = |C'_k|$ is the number of nodes in the cluster C'_k and n is the total number of nodes. The same holds for the random variable Y_l associated with the l^{th} cluster in cover C''. The joint probability distribution $P(X_k, Y_l)$ is defined as:

$$\begin{aligned}
P(X_k = 1, Y_l = 1) &= \frac{|C'_k \cap C''_l|}{n} \\
P(X_k = 1, Y_l = 0) &= \frac{|C'_k| - |C'_k \cap C''_l|}{n} \\
P(X_k = 0, Y_l = 1) &= \frac{|C''_l| - |C'_k \cap C''_l|}{n} \\
P(X_k = 0, Y_l = 0) &= \frac{n - |C'_k \cup C''_l|}{n}
\end{aligned}$$

The conditional entropy of a cluster X_k given Y_l is defined as $H(X_k|Y_l) = H(X_k, Y_l) - H(Y_l)$. The entropy of X_k with respect to the entire vector Y is based on the best matching between X_k and any component of Y given by

$$H(X_k|Y) = min_{l \in \{1,2,\ldots,|C''|\}} H(X_k|Y_l)$$

The normalized conditional entropy of a cover X with respect to Y is:

$$H(X|Y) = \frac{1}{|C'|} \sum_k \frac{H(X_k|Y)}{H(X_k)}$$

In the same way, one can define $H(X|Y)$. Finally the NMI for two covers C' and C'' is given by:

$$NMI(X|Y) = 1 - [H(X|Y) + H(Y|X)]/2$$

The extended NMI is between 0 and 1, with 1 corresponding to a perfect matching.

2.8.2.2 Modularity

To develop a method for community identification, one needs an evaluation criteria to judge the quality of the detected community structure. One of such measures was proposed by

Newman and Girvan in [96] and is based on the intuitive idea that random networks do not exhibit (strong) community structure. To measure the quality of a cover produced by overlapping detection algorithms on real-world social networks where the ground truth is usually unknown, most measures extend the framework of modularity Q for a disjoint partition, which is given as:

$$Q = \frac{1}{2m} \sum_{c} \sum_{i,j \in c} \left[A_{ij} - \frac{k_i k_j}{2m} \right],$$

where c is a community, A_{ij} is the element of the adjacency matrix for nodes i and j, $m = \frac{1}{2} \sum_{ij} A_{ij}$ is the total number of edges, and k_i is the degree of node i.

The idea behind modularity of Newman is simple: a subgraph is a community if the number of links among nodes in the subgraph is higher than what would be expected if links were randomly placed. This is exactly what happens in real-world communities, where the number and density of links among people belonging to groups (families, clubs, user groups etc) is higher than expected in a random graph of the same size [97,102,120]. This definition of modularity implies the choice of a so-called *null model* [96]; i.e. graph model to which any other graph can be compared in order to assert the existence of any degree of modularity. When testing for modularity of a complex network, the null model used has so far been a random graph with the same number of nodes, the same number of edges and the same degree distribution as in the original graph, but with links among nodes randomly placed.

2.8.2.3 Link Based Modularity

Nicosia et al. [98] proposed an extension to modularity measure based on the belonging coefficients of links to account for overlap between communities. Since each node has a belonging coefficient for each community, it is possible to define this coefficient for incoming or outgoing edges from a node. We can intuitively suppose that the community belonging coefficient c of an edge $l = (i, j)$ which starts at node i and ends at node j can be represented by a certain function of the corresponding belonging coefficients of i and j to community c, in the following equation:

$$\beta_{l(i,j),c} = F(a_{ic}, a_{jc}) \tag{2.26}$$

The expected belonging coefficient of any possible link $l(i, j)$ from node i to a node j in community c can be defined as $\beta^{out}_{l(i,j),c} = \frac{1}{|V|} \sum_{j \in V} F(a_{ic}, a_{jc})$. Accordingly, the expected belonging coefficient of any link $l(i, j)$ pointing to node j in community c is defined as

$\beta^{in}_{l(i,j),c} = \frac{1}{|V|}\sum_{i \in V} F(a_{ic}, a_{jc})$. Latter coefficients are used as weights for the probability of an observed link and the probability of a link starting from i to j in the null model. These are used in the new modularity defined as:

$$Q^{Ni}_{ov} = \frac{1}{m}\sum_{c}\sum_{i,j \in V}\left[\beta_{l(i,j),c}A_{ij} - \beta^{out}_{l(i,j),c}\beta^{in}_{l(i,j),c}\frac{k^{out}_i k^{in}_j}{m}\right] \tag{2.27}$$

As we said at the beginning of this section, the evaluation criteria chosen will be *modularity*.

2.9 Recommendation Systems

The steps needed to generate a customer profile turned up at the preceding sections (2.3, 2.4, 2.5). Now, once the profiles are obtained it is time to generate personalized recommendations.

Recommender systems are widely used on many web sites to suggest products to their customers. The products can be recommended based on the top overall sellers on a site, based on the demographics of the customer, or based on an analysis of the past buying behavior of the customer as a prediction for future buying behavior.

On general level, the recommendation task consists of predicting for each user (customer) score for each (available) item, describing the relevance or interest. Usually there are two types of information sources for achieving this: content information and transactions.

More formally, the recommendation problem can be formulated as follows:

Let C be the set of all users and let S be the set of all possible items that can be recommended, such as books, movies, or restaurants. The space S of possible items can be very large, ranging in hundreds of thousands or even millions of items in some applications, such as recommending books or CDs.

Similarly, the user space can also be very large millions in some cases. Let u be a utility function that measures usefulness of item s to user c, i.e., $u : C \times S \rightarrow R$, where R is a totally ordered set (for example, non-negative integers or real numbers within a certain range). Then for each user $c \in C$, we want to choose such item $s \in S$ that maximizes the user's utility. More formally:

$$\forall c \in C, \quad s'_c = \arg\max_{s \in S} u(c, s) \tag{2.28}$$

Depending on the application, utility u can either be specified by the user, as is often done for the user–defined ratings, or is computed by the application, as can be the case for a profit-based utility function.

Each element of the user space C can be defined with a *profile* that includes various user characteristics, such as age, gender, income, marital status, etc. In the simplest case, the profile can contain only a single (unique) element, such as User ID. Similarly, each element of the item space S is defined with a set of characteristics. For example, in a movie recommendation application, where S is a collection of movies, each movie can be represented not only by its ID, but also by its title, genre, director, year of release, leading actors, etc.

According to [13] recommender systems are usually classified into the following categories, based on how recommendations are made :

- *Collaborative recommendation:* the user gets recommended items that people with similar tastes and preferences liked in the past.
- *Content–based recommendation:* the user gets recommended items similar to the ones the user preferred in the past.
- *Knowledge–based recommendation:* the user gets recommended items using additional information about both the customer and the items.

2.9.1 Collaborative Recommendation

The main idea of collaborative recommendation approaches is to exploit information about the past behavior or the opinions of an existing user community for predicting which items the current user of the system will most probably like or be interested in. These types of systems are in widespread industrial use today, in particular as a tool in online retail sites to customize the content to the needs of a particular customer and to thereby promote additional items and increase sales.

From a research perspective, these types of systems have been explored for many years, and their advantages, their performance, and their limitations are nowadays well understood. Over the years, various algorithms and techniques have been proposed and successfully evaluated on real-world and artificial test data.

Pure collaborative approaches take a matrix of given user-item ratings as the only input and typically produce the following types of output: (a) a (numerical) prediction indicating to what degree the current user will like or dislike a certain item and (b) a list of n recommended items. Such a top-N list could contain items that the current user has already bought, but usually has not.

More formally, the utility $u(c, s)$ of item s for user c is estimated based on the utilities $u(c_j, s)$ assigned to item s by those users $c_j \in C$ who are *similar* to user c. For example, in a movie recommendation application, in order to recommend movies to user c, the collaborative recommender system tries to find the *peers* of user c, i.e., other users that have similar tastes in movies (rate the same movies similarly). Then, only the movies that are most liked by the *peers* of user c would get recommended.

Exists approaches to develop collaborative recommendation such as (according to [65]):

- *User–based nearest neighbor recommendation:* the main idea is that given a ratings database and the ID of the current (active) user as an input, the system identifies other users that had similar preferences to those of the active user in the past.
- *Item-based nearest neighbor recommendation:* The main idea of item-based algorithms is to compute predictions using the similarity between items and not the similarity between users.

Others approaches mentioned in [65] includes preprocessing techniques, matrix factorization/latent factor models [75], latent semantic factors [38], principal component analysis [51], association rule mining [82] among others.

2.9.2 Content–based Recommendation

We see that for applying collaborative filtering techniques, except for the user ratings, nothing has to be known about the items to be recommended. The main advantage of this is, of course, that the costly task of providing detailed and up-to-date item descriptions to the system is avoided. The other side of the coin, however, is that with a pure collaborative filtering approach, a very intuitive way of selecting recommendable products based on their characteristics and the specific preferences of a user is not possible.

In content-based recommendation methods, the utility $u(c, s)$ of item s for user c is estimated based on the utilities $u(c, s_i)$ assigned by user c to items $s_i \in S$ that are *similar* to item s. For example, in a movie recommendation system, in order to recommend movies to user c, the content–based recommender system tries to understand the common points among the movies user c has rated highly in the past (specific actors, directors, genres, subject matter, etc.). Then, only the movies that have a high degree of similarity to a user's preferences would get recommended.

According to [4] the system can be enhanced using techniques from information retrieval generating *item profiles* and *user profiles*. The improvement over the traditional information retrieval approaches comes from the use of user profiles that contain information about users' tastes, preferences and needs. The profiling information can be elicited from users explicitly, for example, through questionnaires, or implicitly learned from their transactional behavior over time. This kind of methods will be used in this thesis project. Generating recommendations based on user profiles obtained from transactional data.

Recommendations in content based methods make use of descriptions of events, items and users. Collaborative filtering (CF) methods use only information about users and items on identity level. The CF recommendations are based on information about user's previously purchased items and users who have purchased items in common. Burke [25] gives a more detailed description to CF and hybrid models based recommendation algorithms.

Recommendation algorithms have been applied in various domains before. Two of the most widely studied domains are item recommendations based on item similarities, for example [83] and predicting item ratings based on user ratings on other items. The latter has provided wide range of scientific research, probably because of availability of large datasets. For recommendation approaches in predicting user ratings, see for example [74] or [1].

2.9.3 Knowledge–based Recommendation

In other domains, such as consumer electronics industry, large number of one–time buyers are registered. This means that we cannot rely on the existence of a purchase history, a prerequisite for collaborative and content-based filtering approaches. However, more detailed and structured content may be available, including technical and quality features.

Take, for instance, a recommender system for digital cameras that should help the end user find a camera model that fits his or her particular requirements. Typical customers buy a new camera only once every few years, so the recommender system cannot construct a user profile or propose cameras that others liked, which –as a side note– would result in proposing only top–selling items.

Knowledge–based recommender systems help us address the above challenges. The advantage of these systems is that no rating data are needed for the calculation of recommendations. Recommendations are calculated independently of individual user ratings: either in the form of similarities between customer requirements and items or on the basis of explicit recommendation rules.

The recommendation process of knowledge-based recommender applications is highly interactive, a foundational property that is a reason for their characterization as conversational systems [24]. This interactivity aspect triggered a slight shift from the interpretation as a filtering system toward a wider interpretation where recommenders are defined as systems that "guide a user in a personalized way to interesting or useful objects in a large space of possible options or that produce such objects as output" [24]. Recommenders that rely on knowledge sources not exploited by collaborative and content–based approaches are by default defined as knowledge-based recommenders by [24] and [44].

Chapter 3

Methodology

"We do not remember days, we remember moments."

Cesare Pavese

The methodology of this thesis is based on Cross Industry Standard Process for Data Mining (CRISP-DM) [129] which was developed by IBM[1] and is based on the process of *Knowledge Discovery in Databases* (KDD) [43]. KDD is a linear process composed by five continuous stages (Selection, Pre-Processing, Transform, Application of Data Mining Techniques and Interpretation of the Results).

CRISP-DM is an spiral methodology. One of its main characteristics is that is independent of both the industry sector and technology used. Is intended to be a way, to develop large data mining projects, less costly, more reliable, more repeatable, more manageable, and faster according to [129].

This methodology contains the phases, their tasks, and their outputs. Is based on nine stages, as shown in figure 3.1, that are described as an overview of the life cycle of a data mining project. The sequence describe in figure 3.1 is not strict, and arrows indicate only the most important and frequent dependencies between phases.

The steps of the methodology and their description are based on [129]. We will add some steps required by the Market Basket Analysis based on Social Network Analysis that we propose.

[1]Acronym for International Business Machines Corporation

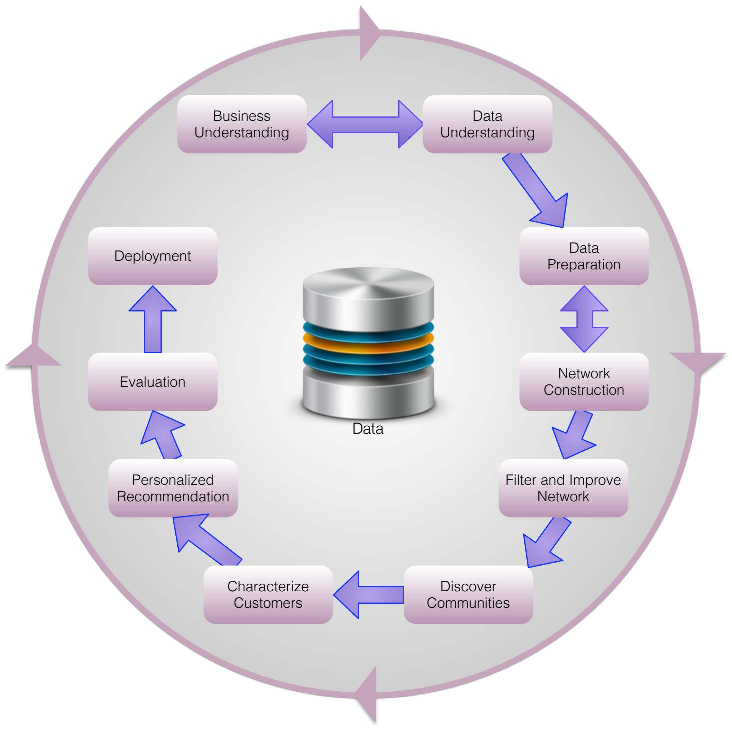


Figure 3.1: CRISP-DM Methodology.

Business Understanding

In order to understand the business problem is need to first pass through a sub-step of research of available literature both in Market Basket Analysis and Social Network Analysis. To understand the business is necessary to participate in several meetings with retail analysts.

Data Understanding

After is obtained a comprehension of the business it is time to understand how real retail's transactional data is structured. The meaning of each table in the database. Is necessary to understand if the data available allow to apply the different algorithms. Verify data quality , identifying quality problems. This is a first approach to the available data.

Data Preparation

Once data is fully understood. It is necessary to prepare data to be able to process it. This stage contains all activities related to data preparation, including data cleaning, removing incorrect values, unknown values or missing values. Data is aggregated according needs of the different models and algorithms that will be applied.

Modeling

At this stage takes place a process of discovery relationships, patterns and trends, using various techniques such as market basket analysis and graph mining.

Evaluation

Before presenting results to retail's analysts is necessary to validate the results, to check that are robust solutions according to the objectives proposed in section 1.4. It is necessary to compare different models in terms of quality and better fitting to the problem. If exists a problem both in models and considering the whole aspect of the problem it will be necessary to return into an early stage, reformulating and regenerating the execution and evaluation.

After models are validated in terms of quality, it is time to present them into retail analysts. They will interpret the results and will give a meaning to the solution found. This is an iterative process, because if more information is needed it will be necessary to rebuild the solution according to their requirements.

Deployment

After the models are obtained and validated by retail's analysts. It must be encapsulated in a system so that retail analysts can take full advantage of this information.

3.1 Basic Notation

Data from retailers is obtained within a period of time K. Time will be partitioned equally, for example daily, weekly, monthly, etc., obtaining a set of time periods, $M = \{period_1, period_2, , \dots, period_{|M|}\}$, where $|M|$ is the total number of periods available.

We have a set of products and transactions. Products are defined formally as $P = \{p_1, p_2, \dots, p_n\}$ where each p_i represents a specific SKU available. Indeed $|P| =$ *number of distinct SKUs*. Let D_m be a set of transactions occurred during a time period m. A transaction T is defined according to [7] as a set of items (products in this case) purchased in the same buying opportunity, such that $T \subseteq P$. Associated with each transaction exists a unique identifier, called its TID.

Nowadays, each coupon generated by the retailer can be separated into two smaller parts: frequent and associated products. Frequent are items that are usually purchased by the customer (simply ordering the products most purchased). Associated products are obtained after applying associations rules. Specifically, are the *consequent* item of the *rule* where the *antecedent* is a frequent item.

3.2 Data Selection

Data from retailers is usually stored in a relational database management system, with a particular structure. Basic data to be stored, in each purchase, is a ticket with certain data, such as: date of purchase, number of products, price, promotions or discount and also, customer identification number if exists.

This thesis projects aims to generate personalized recommendations. To accomplish this objective is necessary to complete and to characterize customer profiles using graph mining techniques. To meet this objective, is necessary to recognize –at least– a subset of customers and their particular purchases. Once we have, the products purchased we can obtain a customer characterization and the degree of membership to each discovered community.

3.3 Preprocessing Data

Databases are highly susceptible to noisy, missing and inconsistent data. Low-quality data will generate low-quality results. In this section, we will briefly describe the principal steps necessary to preprocess data.

3.3.1 Missing Values

Is important to check that we have all data required to understand the results generated by the community detection process. This implies that products have all its information, such as, name of the product, SKU, family, line, sub-line, etc. We have to review that tickets are complete, this means that there are no missing values, and if exists missing values, we have to define a policy, such as, ignore the tuple, fill in the missing value manually, use a global constant to fill in the missing value, use a measure (e.g., mean or median) to fill the missing value, use the most probably value to fill in the missing value.

3.3.2 Noisy Data

Noise is a random error or variance in a measured variable. For example, price equal to 100 where the price is usually 10. Common techniques to find noisy data are binning, regression and outlier analysis.

3.3.3 Tuple Duplication

We have to used normalized tables in order to avoid replications or discrepancies for instance, if a purchase order database contains attributes for the purchaser's name and address instead of a key to this information in a purchaser database, the same purchaser's name can appear with different addresses within the purchase order database.

3.4 Data Reduction

Retail data from a transactional database, where all the purchases are stored, can be a huge source of data. Develop a complex data analysis and mining on huge amounts of data will be hard to process and can take a long time, making such analysis impractical or infeasible.

Data reduction strategies includes dimensionality reduction, numerosity reduction and data compression.

Dimensionality reduction Is the process of reducing the number of random variables or attributes under construction. Dimensionality reduction methods include wavelet transforms (e.g., [10,69]) and principal components analysis (PCA) (e.g., [66]) which transform or project the original data onto a smaller space. Attribute subset selection is a method of dimensionality reduction in which irrelevant, weakly relevant, or redundant attributes or dimensions are detected and removed (e.g., [57, 127, 134]).

Numerosity reduction These techniques replace the original data volume by alternative, smaller forms of data representation. These can be parametric, where a model is used to estimate the data, so only the data parameters are stored, instead of the actual data. Regression and log–linear models are examples (e.g., [49, 131]). Nonparametric methods for storing reduced representations of the data include histograms, clustering, sampling and data cube aggregation.

Data compression In these methods, transformations are applied to the original data to obtain a reduced or compressed representation. If the original data can be reconstructed from the compressed data without any information loss, the data reduction is called lossless. Instead, if we can only reconstruct an approximation of the original data, then the data reduction is called lossy.

A dimensionality reduction that can be used is, to consider a transaction as a set of product families or lines. With the aim, to obtain a small representation that, can be manageable and also allows to apply data mining techniques.

Exists many other ways of organizing methods of data reduction. The computational time spent on data reduction should not outweigh the time saved by mining on a reduced data set size.

3.5 Network Configuration

To build the social network graph, tickets must be taken into consideration. We will assume that products purchased in the same buying opportunity are related to each other. Following this assumption, the network will be configured as follows: *nodes* will represents the products present in a specific ticket and *edges* will represent a relationship between them.

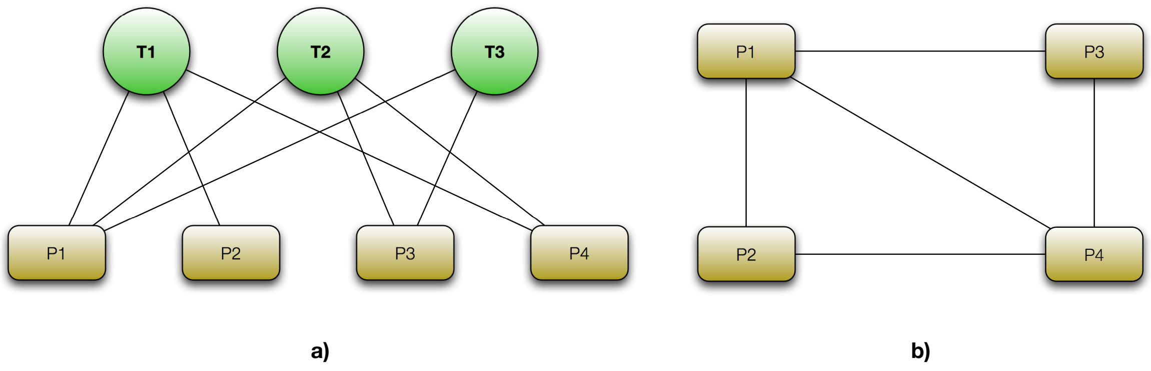


Figure 3.2: Bipartite transaction products network (a) and co–purchased product network (b).

In the literature exists two approaches to generate a network, one is introduced by [70, 116–118], where a bipartite customer product network is built. This network links transactions with products, as depicted in 4.3 (a). The second approach is introduced by [107], based only on transactions where each product is linked to others because they appear in the same ticket from the same buyer opportunity, this kind of network is named *co–purchased product network* and is depicted in figure 4.3 (b). In this thesis project the *co–purchased product network* will be used.

3.6 Network Construction

The objective of this section is to explain how are construct the network of products. The network will be build according the description gave in section 3.5. Considering all the transactions T occurred during a certain period of time k. We will review each transaction t, that transaction will be composed by a subset of products P_{subset}. For each pair of products $(p_i, p_j) \in P_{subset}$ an arc a_{ij} will be created. After this process is done, the resulting graph $G(N, E)$ can be filter according process described in section 3.7. This way, we can reduced the size of the graph reducing the spurious edges, that do not represent powerful relationships between products.

Procedure 7: Procedure to generate the co–purchased product network.

```
Input:
        $T_k$: set of transactions occurred over a period of time $k$.
Output:
        $G(N, E)$: co–purchased product network.
Method:
        $G(N, E)$ = empty graph.
        foreach Transaction $t \in T_k$ do
            foreach Pair of product $(p_i, p_j) \in t$ do
                $addNode(G(N, E), (p_i, p_j))$
                $addEdge(G(N, E), (p_i, p_j))$
            end
        end
        return $G(N, E)$
```

3.7 Network Filtering

The idea is to generate filters over the product network generated in 3.5. Then we will apply a threshold θ that each *edge* have to be, at least, bigger or equal to that threshold in order to remain in the product network and not to be removed. The idea behind this process, is to remove edges that represent spurious and not frequent relationship between products. Relationships that are not lasting over time.

The process recently described implies to iterate over the complete set of edges looking for those that do not meet the threshold θ.

The process is described in procedure 8

Procedure 8: Procedure to generate a filtered network.

```
Input:
    $G(N,E)$: a co–purchased product network.
    $\theta$: the minimum threshold required.
Output:
    $G'(N',E')$: co–purchased product network with spurious edges
    removed.
Method:
    $G'(N',E')$ = empty graph.
    foreach $edge(u,v) \in E$ do
        if $weight(u,v) \geq \theta$ then
            $addEdge(G'(N',E'),(u,v))$
    end
    return $G'(N',E')$
```

3.8 Community Detection

Before we explain the overlap community detection process, we have to explain what is understood by *Community Detection* in graphs. It is the process of trying to find a group of strongly connected nodes. According to [45] the first problem is to look for a quantitative definition of community, because usually, the definition depends on the specific system or application developed. Basically the main idea is that there should be more edges *inside* the community than edges linking vertices of the community with the rest of the graph. Moreover, most of the cases, communities are algorithmically defined without a precise a priori definition. Mathematically the problem of finding communities inside a graph is describe as follow:

Given a graph (or network) $G = \{V, E\}$, where V is a set of n nodes and E is a set of m edges, a series of disjoint subgraphs $K = \{K_1, \ldots, K_j\}$ are generated. The number j of subgraphs to find is not known previously and is determined by the algorithm, based on the maximization of a function $f(K)$. So typically, this function is *modularity* [96]. As we mentioned previously, this function measure the quality of the community in terms of the community structure. Is important that the communities have structure of communities, but its main contribution has to do with the underlying relationships between products.

3.8.1 Overlapping Community Detection

The overlapping community detection problem can be solved using the algorithms exposed in section 2.8.1.5. These algorithms uses a topology–based approach. We will use specifically two algorithms: COPRA [53] and SLPA [133], we based this election in the results obtained by [93, 132]. They show that these two algorithms have the best performance both in time and quality of the results.

3.9 Customer Characterization

Once the communities are found, it is time to characterize the customers based on their previous purchases. The idea is to find the belonging degree to each community. In order to do this, as we have the community with their respective products associated, we will calculate the belonging degree G_i^k, for each customer i to the community k as:

$$G_i^k = \frac{Q_i^k}{\sum\limits_{k \in K} Q_i^k} \quad , \quad \forall i, k \tag{3.1}$$

where Q_i^k represents the quantity of purchases realized by customer i, over the period of time, where at least one product belongs to community k. It is clear from equation 3.1 that the degree will be a value between $[0, 1]$. The belonging degree can be recognized as a *community support* because represents the fraction of transactions that belongs to a particular community over the number of transactions realized that belongs to the rest of communities in that period of time.

3.10 Personalized Recommendations

Once the customer is characterized it is time to generate a personalized recommendation. To do so, we will use the customer characterization, specifically, the belonging degree to each community discovered in 3.9. We choose the most relevant products to each customer mixing their previous purchases, from a certain time window, and the belonging degree to the communities with the set of available products to be recommended. Figure 3.3 explains this process.

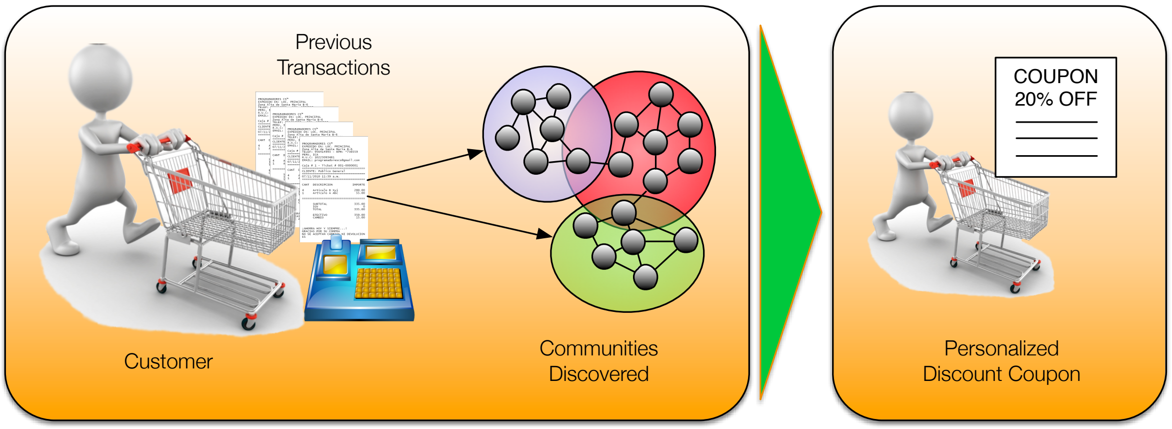


Figure 3.3: Recommendation Process

In detail, the process consists of the following stages:

Available Products

The retailer provides us a set of products, $A = \{p_1, p_2, \ldots, p_m\}$, that can be selected to be included into the discount coupon. This set is a subset of P (described in 3.1), i.e. $A \subseteq P$.

Customer Characterization

For each customer we take a set of the available transactions to obtain a belonging degree to each discovered community as we described in section 3.9.

Coupon Generation

We obtained, in a previous stage, a belonging degree per customer to each community. Now, we choose products from communities –with the higher belonging degree– that make a match between the available elements (chosen by the retailer) and the previous item purchased. These items will be the *frequent* elements of the coupon. In order to obtain the *associated* products, we will select items from communities with high degree[2], that are related to other products. This relationship will be discovered using the *product network*, the only requirement is that these products have to be from different subcategories of products to avoid the recommendation of the same product but in a different flavor (for example: Frequent: yogurt brand A, flavor B. Associated: yogurt brand A, flavor C).

[2]Can be selected with low degree if it is desired by the retailer

3.11 Evaluation

After the communities are discovered, the customers characterized and the recommendations generated, it is necessary to measure the quality of the results. To do so, the communities are measured in terms of the *modularity* [98], also both communities and customer characterization are introduced to the analysts. Based on they expert judgment the quality of the results –both communities and costumer characterization– can be qualified because there are no quantitative metric to measure that can rate it. *Modularity* can measure the quality of the community structure but the meaningfulness, in terms of real benefits for the retail, can only be given by the analysts.

Recommendation's quality will be measured in terms of the available data. We have data from previous recommendations offered to each customer and their posterior usage. A recommendation based on the information available will be generated and its quality measured. This is a sub estimation of the benefits for applying this methodology. Also, it can become a new way to generate recommendations. In fact, we can estimate the incremental income generated by the usage of this method.

3.12 Deployment

Finally, once the results are measured, all the information obtained are introduced into a computer system that can help analysts to manage both communities and customer characterization. The system will help them to handle and improve the results obtained by the whole process. It only will handle the customer characterization, recommendation part is not included at this time because it is only experimental.

Chapter 4

Application over real retail data

"Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning."

Albert Einstein

In this chapter an application over real data from two supermarket chains will be presented. The structure of the chapter will be based in the methodology described in chapter 3. In the first place, we will present the supermarkets which we work with. Then will be exposed the characteristics of the data, the cleaning process. Once the data is ready, we will explain how are generated the networks of products, how are applied the algorithms to discover overlap communities, the process to characterize customers based on their previous purchases. Next, the developed system to store and to manage the results is presented. Finally, the results are introduced to the retail analysts, in order to measure the usefulness of the work performed.

4.1 Retail Data

The data was obtained from two retail chains in Chile. One is a wholesale supermarket oriented to supply products to grocery store owners, hereafter, referred to as *Retail A*. The second is member of one of the biggest retail holdings in Chile called *Retail B*.

Our data was gathered within a period of thirty months, around 238 million transactions, approximately 160 thousand clients and over 11 thousand SKUs[1] in the case

[1]SKU : Stock Keeping Unit

of *Retail A* chain. For *Retail B*, the gathered period was two months, with 128 million transactions, almost 2 million customers and 31 thousand different SKU.

We presented in section 3.1 that data is obtained over a period of time, denominated K. This period is partitioned into smaller sets of time periods. The set of time periods chosen is : $M = \{daily, weekly, monthly, quarterly, biannual, yearly\}$, this set is sub–indexed, in a way that can be manageable, obtaining the following sets:

$$M_a = \{day_1, \dots, day_{1229}, \dots, month_1, \dots, month_{41}, \dots, year_1, year_2, year_3, year_4\}$$

$$M_b = \{day_1, \dots, day_{62}, week_1, \dots, week_9, month_1, month_2\}$$

in the case of *retail A* and *retail B* respectively. We only have three years in M_a because the data is available since 3.5 years. The set of applicable time periods, in the case of *retail B*, are only *daily, weekly* and *monthly*, because we have two month of data available.

According to above, we have subsets of transactions that belong to a particular time partition k, these can be expressed by

$$T_k = \{t_1, t_2, \dots, t_q\}$$

as an example for a particular day *d1* we will have: $T_{d1} = \{t_{7522}, \dots, t_{14211}\}$ where t_{7522} and t_{14211} are the first and the last purchase of the day respectively. It is important to remember that each purchase represented by a transaction t like $t_{7522} = \{p_{21}, p_{34}, p_{675}, p_{982}\}$

In the datasets, from these two retailers, products are organized in a three hierarchical level structure. Each level belongs to its predecessor based on an ad–hoc developed taxonomy by each retailer. Figure 4.1 shows a subset of one of our taxonomy and table 4.1 shows an example of product information with its hierarchy.

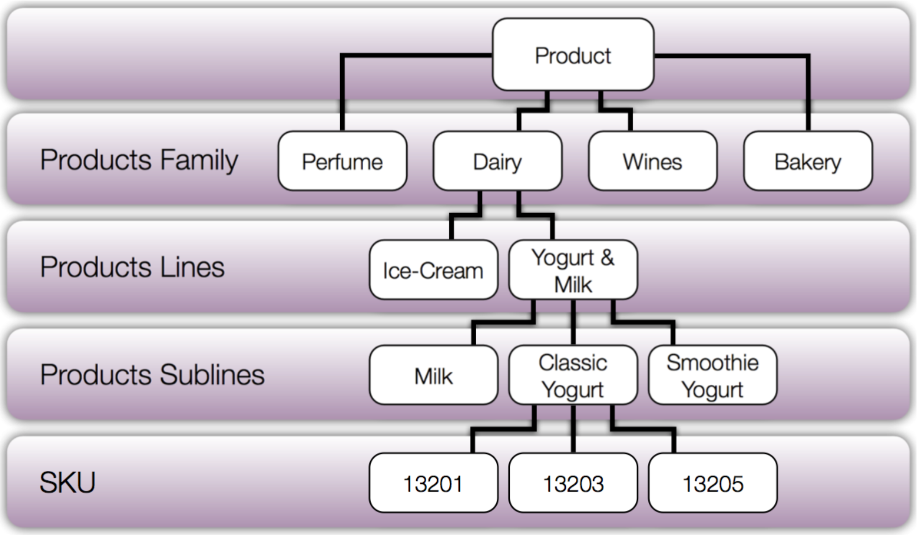


Figure 4.1: Hierarchy of products

Retail A has 23 product families, 150 lines of products and 415 subcategories of products. *Retail B* has 50 product families, 287 lines and 1032 subcategories of products.

SKU	Product name	Product Family	Product Line	Product Sub-Categories
13231	Milk "The Happy Cow"	Dairy	Yogurt & Milk	Milk
13201	Yogurt "Fancy Yogurt"	Dairy	Yogurt & Milk	Classic Yogurt
13245	Yogurt "Smoothiest"	Dairy	Yogurt & Milk	Smoothie Yogurt

Table 4.1: Products characterization available

In the following sections, we will explain how data is processed. In order to be in a format that enables us to perform our experiments.

4.2 Data Transformation

To develop our proposal, it is important to have data, but not all data is necessary. Retailers have databases with several millions or billions of records. For the development of this work, is only necessary transactional data, specifically products purchased in each buying opportunity and whom bought the products, if available. Also, the information related to each product, such as family, line or sub–line, just to make the results helpful to the analysts, because in the processing we only need the set of products present in each transaction.

We received the data from retailers directly in dump files from their databases. These data, depending on the retailer, have a particular structure. We had to transform this data, in order that fit into our data model. The data model is described in figure 4.2.

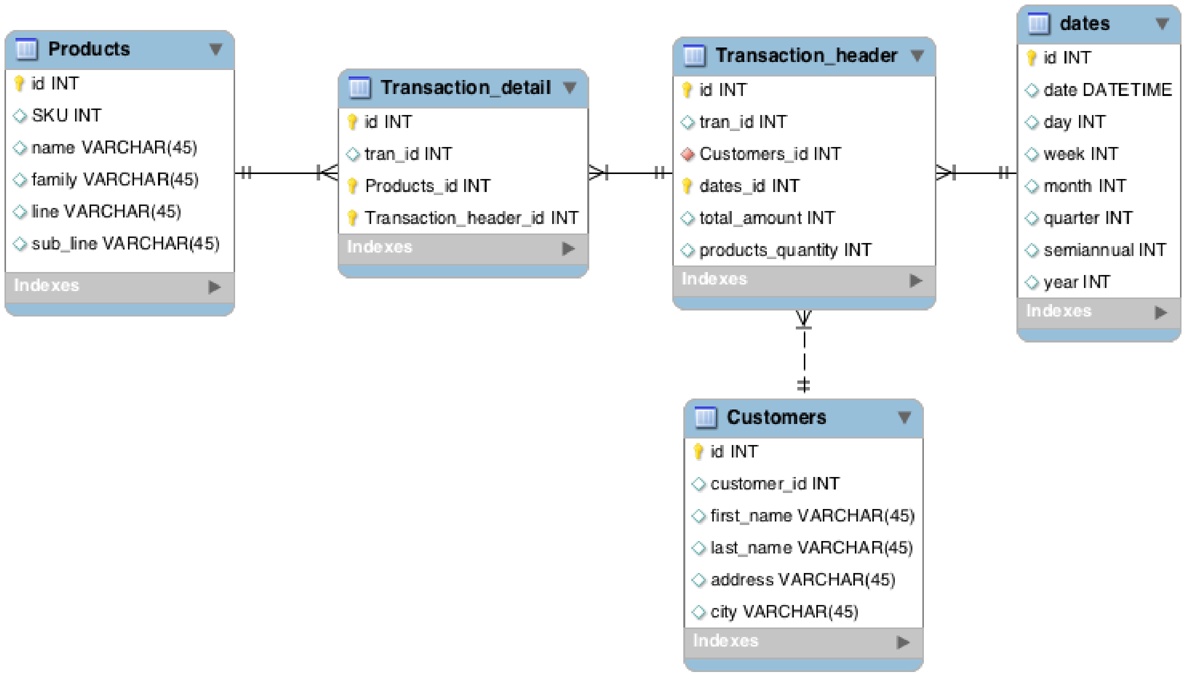


Figure 4.2: Basic Database Model

4.2.1 Data Model Explain

In this subsection, we will describe each of the tables contained in the model of the figure 4.2.

Products is the table used to stored the products related information, such as, name, family, line, sub–line, etc.

Transaction Header is the table were the main details of each purchase are stored, like, the customer id –if available–, quantity of products purchased, total amount of the transaction and the date of the purchase.

Transaction Detail is the table used to store the detail of the products purchased in a particular transaction, is a relation between *Products* and *Transaction Header*.

Dates is a table were a representation of the different dates are organized by different criteria such as, day, week, month, etc.

Customers is the table that handle all the information related to customers, like first name, last name, address, etc.

To be able to reconstruct a transaction with all the information, is necessary to make a cross between the different tables involved.

Once the data is stored in our system, is necessary to study the quality of the data, as we mentioned in section 3.3, in terms of missing values, noisy data and tuple duplication. The data present no missing values. Also, as the information comes from data warehouses does not present noisy data. Finally, we made a deep study over the data, in order to discover tuple duplication. No tuple were repeated in the data sets available.

4.3 Classical Approach

In this section, we will explain how we tried to address the problem in the first instance. Starting with the process to generate clusters of products that are purchased together. To accomplish this objective, we mentioned in section 2.5 the principal clustering methods, and in section 2.5.5 we described the two principal algorithms applied over transactional data, K–Means [60] and SOFM [73].

In order to apply K–Means algorithm over transactional data, we had to construct a representation of the data. This is depicted in table 4.2. Is a matrix, whose rows are vectors of purchases. Each vector is composed by transactions and the set of products available. The first column stored the transactional id (*TID*) and in the following columns stored a number 1 or 0 which represents whether the product was purchased or not in that particular transaction.

Transaction ID	P1	P2	P3	P4
925	1	1	0	1
926	1	0	1	1
927	1	0	1	0

Table 4.2: Example of a transaction set as a vector of purchase for a particular time partition

Once the data are sorted, according what we need, we proceed to apply the K-Means algorithm over these datasets. The results obtained, independent the time partition chosen, are far from being useful. From K-means we basically obtained the same cluster independent of the number of clusters K that we required. To improve the results we tried to apply a dimensionality reduction as we mentioned in 3.4. We constructed a new matrix, but this time instead of vectors composed by products purchased we used families of products. The results, again, were far from being useful. Table 4.3 shows the top 18 categories found by K-means, each column displayed is sorted by the membership of each product family to the respective cluster.

It is clear from table 4.3 that clusters are very similar and showing no real difference between them. Also, analysts said that these clusters are meaningless to them, resulting in no new information from this clusterization. As a manner to depict this fact: Cluster 5 and Cluster 6 are very similar. (In fact the top three families are the same in both clusters.). Only the Cluster 2 provides information, because the belonging coefficient of *Soft Drinks* is 1.0 and the second and third families has a belonging coefficient of 0.15 and 0.13 respectively. Analysts said that this Cluster represents the transactions were the principal products are members of the *Soft Drinks* family.

Clusters	**Cluster 1**	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**	**Cluster 6**	**Cluster 7**
% Transactions Involved	5,1	10,8	46,8	5,5	11,4	13,4	7,0
Families	Yoghurt	Soft Drinks	Cigarettes	French Fries	Milk	Yoghurt	Long Noodles
	Milk	Nectars	Milk	Cookies	Yoghurt	Frozen Desserts	Sugar
	Sugar	Milk	Cheese	Snacks	Frozen Desserts	Milk	Rice
	Toilet Paper	Cookies	Cookies	Milk	Nectars	Cookies	Tomato Sauce
	Vegetable Oil	Yoghurt	Nectars	Souffle	Cheese	Nectars	Vegetable Oil
	Tomato Sauce	Mineral Water	Toilet Paper	Yoghurt	Margarine	Cheese	Short Noodles
	Margarine	Beers	Margarine	Nectars	Butter	Milk	Toilet Paper
	Juice (Powder)	Cheese	Sugar	Soft Drinks	Cookies	Margarine	Yoghurt
	Cheese	Toilet Paper	Vegetable Oil	Biscuits	Soft Drinks	Sausage	Milk
	Short Noodles	Sausage	Sausage	Soft Candy	Juice (Powder)	Toilet Paper	Tea
	Rice	Wine	Juice (Powder)	Pretzels	Milk	Juice (Powder)	Margarine
	Long Noodles	French Fries	Beers	Cookies	Toilet Paper	Biscuits	Salt
	Cookies	Juice (Powder)	Frozen Desserts	Frozen Desserts	Sausage	Pretzels	Juice (Powder)
	Bleach	Sugar	Wine	Hard Candies	Sugar	Soft Drinks	Sausage
	Frozen Desserts	Vegetable Oil	Paper Towels	Chocolate	Sausage	Butter	Bleach
	Nectars	Biscuits	Flour	Juice (Powder)	Vegetable Oil	Sugar	Detergent
	Mayonnaise	Margarine	Chocolate	Cheese	Bleach	Cereals	Cookies
	Tea	Pretzels	Pretzels	Toilet Paper	Delicacy	Sausage	Mayonnaise

Table 4.3: Top 18 families per cluster after execute k-means algorithm with $K = 7$

In the case of SOM we obtained basically meaningless information. The main problem of these results are, the impossibility of clusterize customers with these meaningless groups of products.

These facts motivated us to propose a new methodology to generate clusters of products related between them as we describe in section 4.4. This methodology tries to find a meaningful item set with an approach based on product network with overlapping community detection.

We generated sets of *association rules* (explained in 2.6.3) where we obtained a set of rules of the type $A \rightarrow B$, where A and B are products purchased in the same buying opportunity. These rules can be ordered by their *support, confidence* or *lift*. However, the underlying relationship between these products is not clear in a simple way, after a deep analysis, analysts can recognize groups of rules that are related, for example, *soft drinks* group, could be recognized after studying a set with more than 500 rules, but other rules present no meaning for analysts. Instead, our methodology delivered as an output the group of soft drinks implying no effort for the analysts, also other groups that not were shown by association rules were discovered by our methodology.

4.4 Proposed Methodology

Previously, in section 4.3, we depicted the problems in our first attempt to address the problem. In this section we will show how are generated these products network and will show our proposal of a *Temporally Transactional Weighted Products Networks*. Later we will present what is understood as community and their relation with frequent item set.

We present in this section our approach that is a novel way of generating frequent itemsets through community discovery. From now on we will refer to frequent itemsets as *community*.

4.4.1 Product Network and Graph Construction

We used a network as a way to represent set of elements interconnected between each other. As we shown in 2.7.1 a common way to represent a network is using a *graph*. A *graph* is a manner to specify relationships between sets of items. Formally, a graph consists of a set of objects, called *nodes* with some pairs of them connected by links called *edges*.

A *product network* is defined as a network where nodes represent products and edges represent relationships between a pair of them. We have to define what kind of relationship is represented by an edge. In this case, an edge between two products represents that both products are present in the same ticket from the same buyer opportunity.

We use a network representation based on transactional data shown in figure 4.2. In this subsection we will show how we build our product network, following the methodology proposed in 3.6. We start building our transactional product bipartite network where each transaction is linked to the products that are purchased in that particular transaction (as in figure 4.3 (a)). Here, the set of transactions T and products P represents the two disjointed sets required to build a bipartite graph. Finally, we move from that bipartite network to the co-purchased product network as shown in figure 4.3 (b).

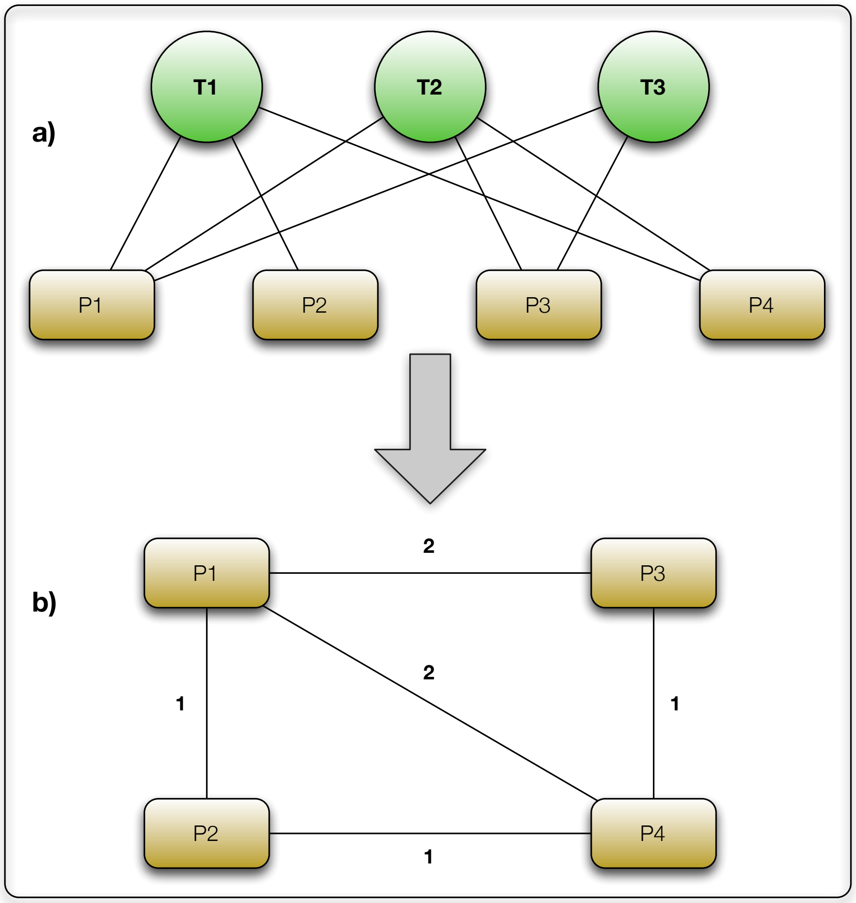


Figure 4.3: From Bipartite Transaction Products Network (a) to Product-to-Product undirected weighted network (b)).

After this processing we obtained a product–to–product weighted network. These networks can be represented by an adjacency matrix showing the weight between each pair of products. The weight is the number of tickets, in which a couple of products are present simultaneously. In table 4.4 we show this representation.

	P1	P2	P3	P4
P1	–	1	2	2
P2	1	–	0	1
P3	2	0	–	1
P4	2	1	1	–

Table 4.4: Adjacency matrix representing a product-to-product weighted network.

Similarly to [107] we found the same problems, such as very dense products networks with highly degree nodes. Most of the time these edges make no sense and represent spurious associations. For example, a month of data has 485,136 transactions, the adjacency matrix of products network obtained has 5,359 products with 5,362,906 edges. Like the adjacency matrix is symmetrical we only consider one half of it. The most heavy weighted edge is 31,224. From those edges, 4,235,093 have a weight lower than 10. This should not be considered because it represents only a sporadic and not frequent relation between products.

4.5 Network Filtering

We mentioned in section 3.7 that is important to apply a filter over the networks, in order to remove spurious edges that contribute nothing. We will explain in the following subsections our proposed methodology.

4.5.1 Threshold Setup Methodology

We showed that product networks presents high degree nodes with spurious edges between them. To remove spurious edges, a threshold θ has to be defined, then the graph is fully revised in the search of edges with a weight θ' lower than θ ($\theta' \leq \theta$). The edges that match with this criteria are removed. Raeder et al., [107] decided to filter those edges that have a weight lower than 10. Kim et al., [70] filter the *co–purchased* network by choosing a threshold θ equal to the average value of all links.

We found that there is no common criteria to choose a threshold. This makes it highly necessary to remove these spurious edges, in an objective way, because it is clear that a particular number (constant) like 10 or the average value of all links are very particular thresholds that apply to particular instances or certain data. For instance, in our case, 10 is not a good threshold because the network obtained after applying this threshold still contain spurious edges and isolated nodes that does not produce communities of good quality.

We generate this threshold based on a process denominated *top three heavy edges threshold* (tthet) which was used in both retailer data, proving its effectiveness. This approach consists in ranking the edges $E = \{E_1, E_2, ..., E_m\}$ based on the weight of these in a descendant order. Then *tthet* is equal to the average of the top three edges.

$$tthet = \frac{E_{max} + E_{2nd\ max} + E_{3rd\ max}}{3} \tag{4.1}$$

where E_{max} makes reference to the heaviest edge, $E_{2nd\ max}$ and $E_{3rd\ max}$ to the second and third heaviest edges respectively.

4.5.2 Network Filter Methodology

In the case of *Retail A* we obtained 1,492 *Temporally Transactional Weighted Product Networks* and in the case of *Retail B* we obtained over 12,000. To each one of these *Temporally Transactional Weighted Product Networks* we computed a threshold using equation 4.1.

If we apply the obtained *tthet* to its corresponding network, only one or two elements would satisfy the minimum edge weight imposed by the threshold. Since tthet allows us to keep the most relevant part of the *Temporally Transactional Weighted Product Network*, making useless the analysis.

This fact prompted us to generate a set of filters using the tthet, that allow gradually incorporating relevant edges and nodes into our analysis. These filters are a proportion of the *top three heavy edges threshold* (proportion is a percentage of the threshold). The percentages are: $Percentage = \{5\%, 10\%, \ldots, 95\%, 100\%\}$; these percentages give 20 filters (or new thresholds), as a result of a dot product between *percentage* and *tthet* resulting in:

$$filters = percentage \otimes tthet \tag{4.2}$$

equal to:

$$filters = \{0.05 * tthet, 0.1 * tthet, \ldots, 0.95 * tthet, tthet\} \tag{4.3}$$

It is clear that for each threshold we have a set of twenty filters associated with the same *Temporally Transactional Weighted Product Network*.

We applied immediately these filters to *Temporally Transactional Weighted Product Networks* giving as a result a new set that we denominated as *Filtered Temporally Transactional Weighted Product Networks* composed of a 29,500 filtered networks in *Retail A* and over 200,000 in the case of *Retail B*.

Figure 4.4 depicts the number of nodes and edges from one month of transactional data after applying *filters* obtained from *top three heavy edges threshold*. Is clear that when the percentage goes up; the number of nodes and edges go down in a power law figure. This fact –power law figure– is important, because allow us to apply the overlapping community discovering techniques.

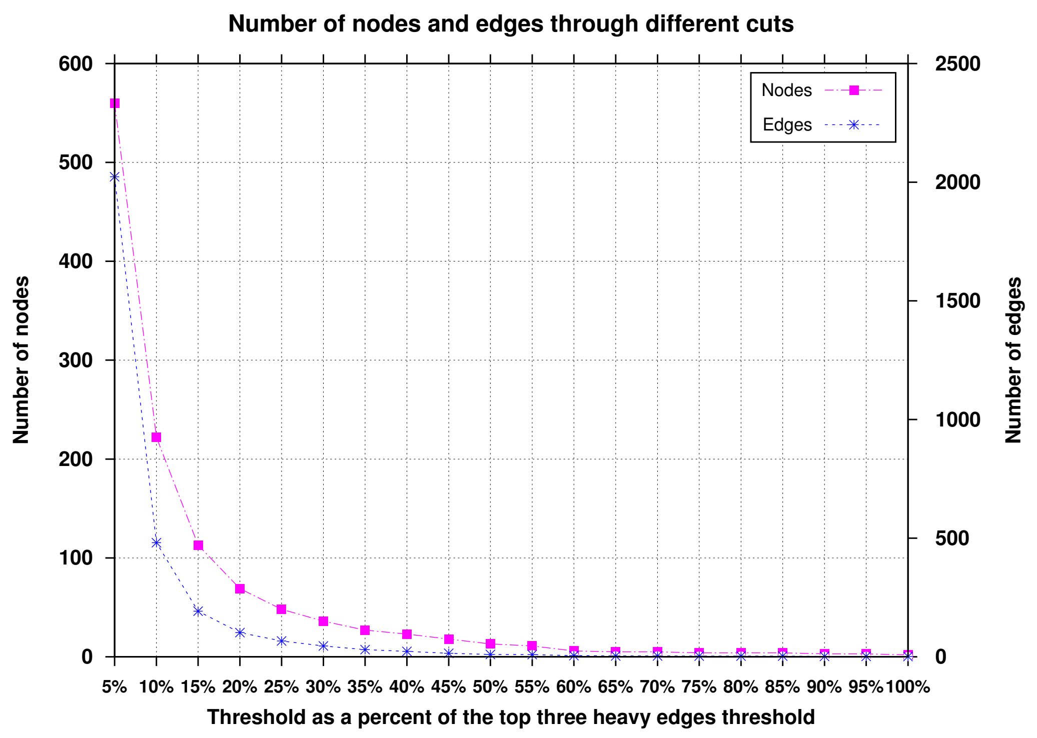


Figure 4.4: Graph depicting the number of nodes and edges through different cuts over a month period.

One of the main advantages of choosing a threshold this way is their independence of the underlying data. This means that our threshold and filters are independent of the quantity of nodes and work very well with big networks both in number of nodes or edge weight. It is also objective because it only depends on the data and requires no intervention from the analyst or a thorough understanding of the business, which is desirable, but not a prerequisite. Thus, our methodology can be reproduced by other works, allowing them to compared their results with ours.

4.6 Overlapping Communities of Products

Graph theory is applied in many fields including analysis of social networks [112], the world wide web [41], epidemiology [92], scientific collaboration [84, 108].

Figure 4.5 depicts a product-to-product network for *Retail A* on a particular day before any filter was applied. As we can see, this network is meaningless because it has many edges with little value (as we explained in section 4.5.1).

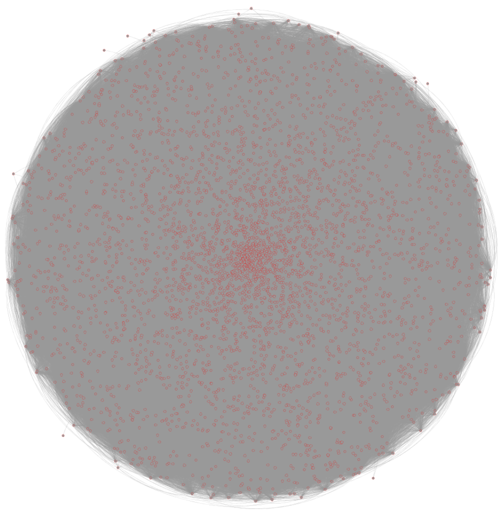


Figure 4.5: Visualization of the product-to-product network without filters.

Now, when a filter is applied, the network shows meaningful zones as we can see in figures 4.6 and 4.7. These figures were obtained after we applied a filter equal to the 5% and 10% of the *top three heavy edges threshold*.

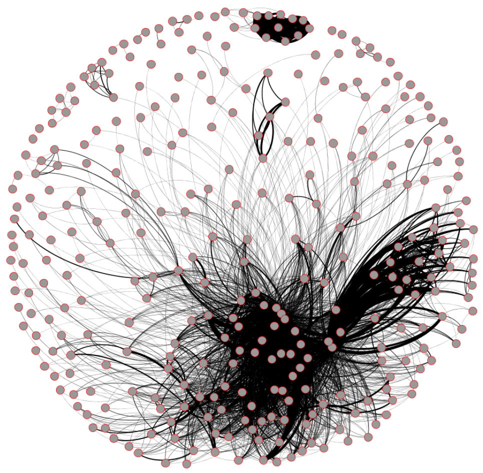


Figure 4.6: Product–to–Product Network with a 5% filter

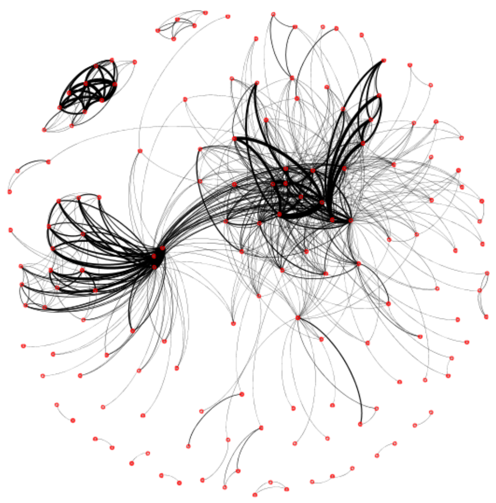


Figure 4.7: Product–to–Product Network with a 10% filter

These zones describes products with a powerful relationship between them. To understand these relationships and giving them a meaningful interpretation is that we applied an overlap community detection process. This process, with overlapping, allows that a *product* (node) can belong to more than one community, After we have generated our set of *Filtered Temporally Transactional Weighted Products Networks* it is time to apply the algorithms for overlapping community detection, described previously in section 2.8.1. We apply to each of the *Temporally Transactional Weighted Product Networks* (over 29,000 and 12,000 for *Retail A* and *Retail B* respectively) COPRA and SLPA algorithms.

Both algorithms take a network as an input and generate a file as an output with the communities discovered inside. COPRA generates only one file with the communities and SLPA generates a number of files that are the product of the number of repetitions that a user selects –10 in our case– and a threshold r. Taking values in $[0, 1]$ ($r \in [0, 1]$ specifically (0.01;0.05;0.1;0.15;...;0.5)) and used as a filter to the number of labels that, a particular node can have checking if the probability of seeing a particular label during the whole process is lower than r. If that occurs, this label is removed from node's memory. When $r \to 1$ the algorithm tends to find disjointed communities. This is explain because only nodes in one community can have a high probability that can overcome the threshold represented by r. After the entire process the nodes are grouped into communities with the same label. If a node has more than one label it is grouped into several communities.

SLPA is executed in a series of runs. A run is an execution of an SLPA algorithm. In each run different values of r ($r \in \{0.01, \ldots, 0.5\}$) are used. Every time the algorithm is executed, a new node is chosen randomly making it necessary to run the SLPA algorithm several times, to avoid that because SLPA started from a '*good*' node the results were improved. We try to find the results that are maintained over time or between executions.

Once the *Filtered Temporally Transactional Weighted Product Networks* are processed by the algorithms we have as result over 3.2 million files in *Retail A* and over 18 million files in *Retail B*.

4.7 Parallel Processing

Previously, we presented the process developed in order to generate products networks (4.4), also how was generated a filter to reduce the number of spurious edges (4.5) and the application of the techniques to discover overlapping communities (4.6). The large amounts of files involved make that a sequential processing is unfeasible because of the time required. This entailed we developed a new way to address the problem through parallel processing techniques. In this section, we show the techniques applied to address this objective.

To process the tasks in parallel, we used a dynamic mapping, specifically a centralized schema as is depicted in [76]. This schema is based in that all executable tasks are maintained in a common central data structure or they are maintained by a special process or a subset of processes. When a special process is designated to manage the pool of available tasks, then it is often referred to as the *master* and the other processes that depend on the master to obtain work are referred to as *slaves*. Whenever a process has no work, it takes a portion of available work from the central data structure or the master process. Whenever a new task is generated, it is added to this centralized data structure or reported to the master process.

The adequacy of the scheme made by us to execute parallel is: first, generate a *master* process that will fill an empty queue of jobs with the tasks to be realized. Second, create a set of *slaves workers* that will realize that tasks. Is important to note that, each worker immediately –as soon as they are created– will start executing a task. The number of concurrent jobs K that can be executed in parallel is limited by the writing capacity of the hard disk drive in this case, because as we have to write the results to files, this was our bottleneck. We try with several values of K, but the best results were obtained with $K = 2P - 1$, where P is the total number of cores availables in the CPU (including virtual cores).

First, we decided the time window to be applied over the transactional data. Then, we generated products networks obtaining the results from the database and generating the product networks for the time windows desired, this process can be done sequentially. After this, we generate and apply the filter over the product networks.

Next, it is necessary to generate a set of new products networks in what we denominate *Filtered Temporally Transactional Weighted Products Networks*. To do so, we generated the twenty filters as a proportion of the threshold θ, then we review the set of edges checking in parallel if the value of the edge exceeds some elements from the set of filters, if it does, both nodes and edges are written into the respective files.

Figure 4.8 presents a graphical representation of the parallel execution using the master–slave scheme for the process of filtering a *temporally transactional weighted products network*.

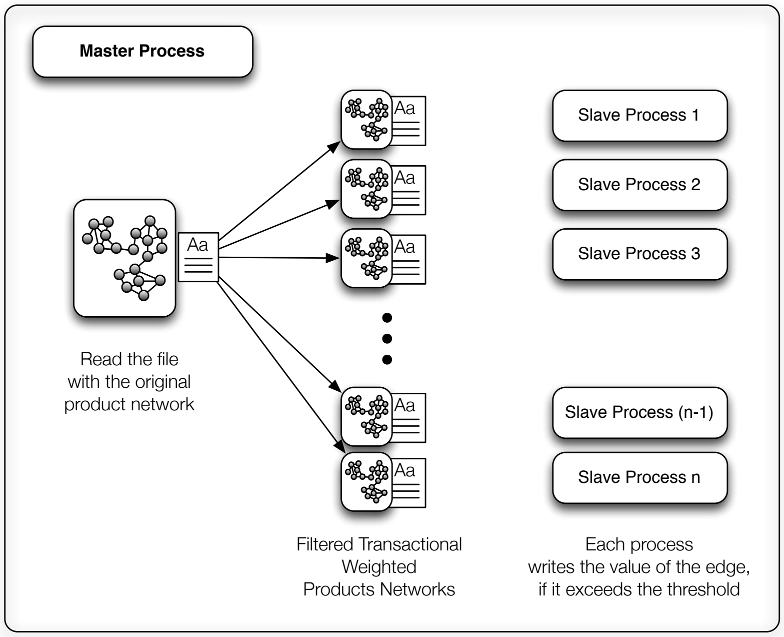

Master Process
Read the file with the original product network
Filtered Transactional Weighted Products Networks
Slave Process 1
Slave Process 2
Slave Process 3
Slave Process (n-1)
Slave Process n
Each process writes the value of the edge, if it exceeds the threshold

Figure 4.8: Master–Slave process to filter in parallel the *Temporally Transactional Weighted Product Networs.*

After the filtered networks are generated, it is time to run the algorithms to detect overlap communities. As we mention previously, the filtered networks are over 3.2 million files in retail A and over 18 million in retail B. We try to process these files sequentially and we obtained results after five days of processing, i.e. over 120 hours, then with the parallel version of the execution we obtained results in 20 hours. These is a quantitative reduction of the processing time.

Algorithm 9 shows the pseudo–code of the procedure applied to discover overlap communities.

Algorithm 9: Master–Slave scheme selected to discover communities in parallel.

```
Input:
        $F$: set of files to be process.
Output:
        $P$: set of files with the overlapping communities discovered.
Method:
        Master Process                                Slave Process
        $Q = \{empty\ queue\}$
        $P = \{empty\ set\ of\ files\}$               $discoverCommunities()$
        $inicialize\_K\_Workers()$                    $writeResultsToDisk(P)$
        foreach File $f \in F$ do                     $notifyMasterProcess()$
            $addToQueue(f, Q)$
            $notifyWorkers()$
        end
        $waitEndOfAllProcess()$
        return $P$
```

This reduction –in the processing time– becomes relevant, because retailers would like to run the developed methodology and its algorithms with a frequency that would be unreacheable with secuencial processing.

4.8 Communities Results

Having 101 files for each *Filtered Temporally Transactional Weighted Product Networks* it is necessary to find a way to discover which file (from 101 available) has the best community representation based on a criteria. As explained in section 2.8.2.2, this criteria is *modularity*.

To obtain modularities we apply a program to each file with their corresponding *Filtered Temporally Transactional Weighted Product Network* and obtained an equal number of modularity text files. These were parsed and inserted into a column oriented database so we could filter by different criteria such as time window, number of nodes, number of edges and modularity value.

As a manner to depict the process we show in table 4.5 the metadata of a Temporally Transactional Weighted Product Network for November, 2012 for a set of supermarket stores from *Retail B*.

In table 4.6 the metadata of one of the twenty filtered networks obtained after apply a threshold is shown. The threshold applied is equal to 1,286 –equivalent to the 5 % of the *top three heavy edges threshold*–. Finally, in table 4.7, the information of the results gathered from the appliance of SLPA and COPRA algorithms are presented.

Network	# of Nodes	# of Edges	Period	Begin Date	End Date
graph_month_201211	23,615	24,153,638	Monthly	2012-11-01	2012-11-30

Table 4.5: Metadata of a Temporally Transactional Weighted Product Network for November, 2012.

Network	**# of Nodes**	**# of Edges**	**Threshold**	**Density**	**Degree**
filtered_graph_month_201211	356	1,133	1,286	0.0179	6,365
	Cluster Coefficient	**Barycenter Scorer**	**Betweenness Centrality**	**Closeness Centrality**	**Degree Scorer**
	0,529	0,078	404,60	0,473	0.078

Table 4.6: Metadata of a Filtered Temporally Transactional Weighted Product Network for November, 2012.

We present a subset of the results from SLPA (10 from 101 available) and the result obtained from COPRA in table 4.7. This table shows the results for SLPA algorithm after one run.

Network	Modularity	# of communities	# of products	# overlaps	r
SLPA_1	0,586	28	358	2	0.01
SLPA_2	0,602	29	360	4	0.05
SLPA_3	0,607	29	358	2	0.10
SLPA_4	0,607	29	358	2	0.15
SLPA_5	0,607	29	358	2	0.20
SLPA_6	0,607	29	358	2	0.25
SLPA_7	0,607	29	358	2	0.30
SLPA_8	0,610	29	357	1	0.35
SLPA_9	0,617	29	356	0	0.40
SLPA_10	0,617	29	356	0	0.45
SLPA_11	0,607	28	356	0	0.50
COPRA_1	0,394	45	361	5	N/A

Table 4.7: Results from one iteration of SLPA and COPRA algorithms.

As we can see from table 4.7; Both algorithms utilize all products available –356 in this case–. For example, in SLPA_7 the number of products is 358, because 2 are overlapped. The COPRA algorithm discovered more communities than the SLPA algorithm, but modularity obtained from COPRA is worst than SLPA due to the fact that 30 communities from COPRA are singletons. On the other hand, the larger number of overlapped products found by COPRA is explained by the fact that COPRA found similar communities that only differed in one product. For example, for a pair of communities found there is $a = \{269324, 901093, 901095\}$ and $b = \{269324, 901096, 901095\}$, which are almost the same community except, for the middle product. A product identified by SKU equal to 901096 is missing in the case of a and SKU equal to 901093 in the case of b.

Analysts said that overlapped nodes found makes sense, because they expected products that are present in more than one community. They referred to this products as *nexus*.

We also found a relationship that is present in almost all results obtained from SLPA, that the higher number of overlapped products is found when $r \leq 0.2$. This is because when r is small ($r \to 0$), too many nodes can overcome this threshold, as explained previously.

Now, once the communities are identified it is time to analize them. To check, if they are meaningful in terms that a description can be given to each one of them.

4.8.1 Discovered Communities

Once the algorithms were applied, we had as a result a set of communities of products, where each product is related to each other. In this section we will describe the communities found, in terms of the meaning of this products within the community.

Table 4.8 depicts the top ten communities ordered by the number of products inside. We also provide a description from the analysts from *Retail B* who study the products involved and gave this description.

Community	# of products	Description
1	242	Grocery
2	15	Soft Drinks & Beers
3	10	Convenience Food
4	6	Juice Powder Brand A
5	6	Juice Powder Brand B
6	6	Liquid Juice Brand C
7	5	Yoghurt Brand D
8	5	Yoghurt Brand E
9	5	Liquid Juice Brand F
10	5	Cookies Brand G

Table 4.8: 10 largest communities discovered (order by number of products inside) which account for 85% of the products in the network.

It is very important to note that these results changed the opinion of the business analysts, because they believed that people were not *loyal* to a specific brand –in the sense of buying products of the same brand–. However, the results showed that people are loyal to a brand in most cases, the only exception being *Drinks & Beers*.

The average number of products inside a community is 7, giving to the analyst a manageable number of products, that can be interpreted and characterized. This is one of the advantages of our work, because for a lower quality information from the association rules. A deep and complex analysis must be performed.

We previously present in section 4.6 how a graph looks after we apply our *top three heavy edges threshold*. Now in figure 4.9 we show how the graph looks with each node colored according their corresponding community. We found a big community of groceries according to an analyst description depicted in green, and the rest of communities in different colors.

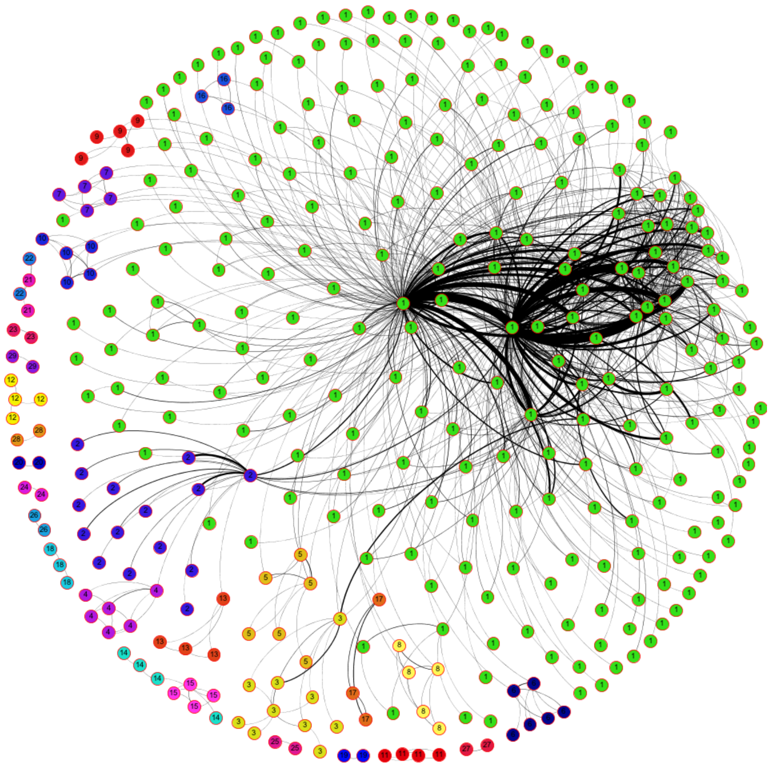


Figure 4.9: Visualization of the product-to-product network with each product associated to their corresponding community.

We found a property also present in [11, 33] that is when a network is partitioned in such a way to maximize modularity, the community sizes q appears to have a power-law form $P(q) \sim q^{-w}$ for some constant w. In this case this constant is $w \simeq 1.3$ considering all the communities available and $w \simeq 0.63$ if we exclude the first community. The number of products in each community are depicted in figure 4.10. The respective power-law function is also plotted.

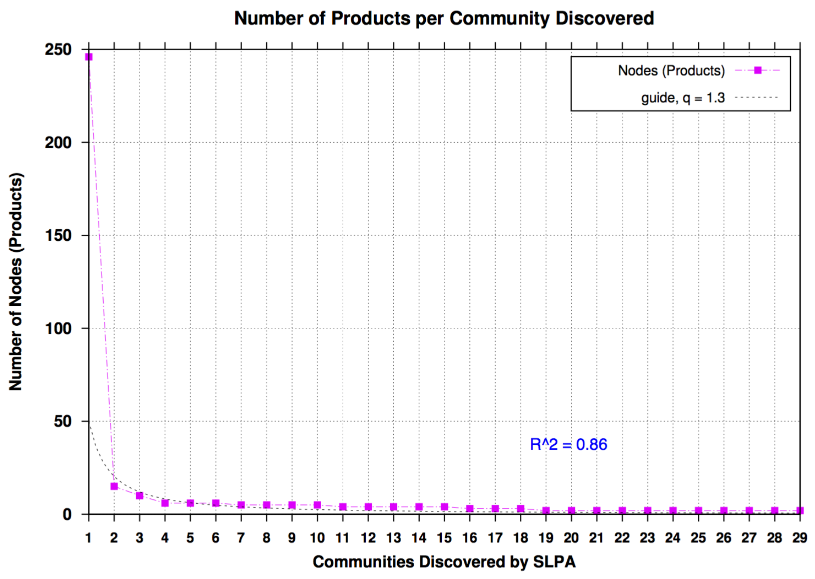


Figure 4.10: Number of products per community discovered by SLPA.

Figure 4.11 is a zoom of the plot presented in figure 4.10. It leaves the first community out of the plot and only shows the 28 communities remaining.

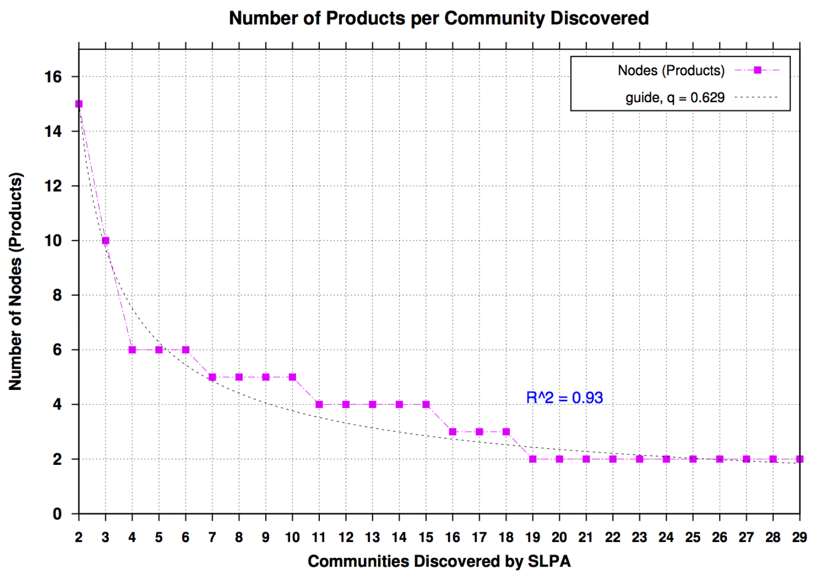


Figure 4.11: Zoom to communities discovered by SLPA.

The *coefficient of determination* denoted by R^2 is 0.86 considering all the communities and 0.93 if we remove the first community. From both figures and the value of R^2, it is clear that power-law function fits very well with the number of elements found in each community.

4.9 Communities Stability

One important aspect when communities are studied is their stability over time. We studied how evolved the communities found, during the period of one year. We compared the similarity between different periods of time over this year of study. This information will help to determine which is a representative period of time, containing communities that do not vary over time.

The process followed to analyze the stability was taking a particular *Temporally Transactional Weighted Product Network* from a specific time window (TW_1), for example a *day*, then we took another time window, for instance a *week* (TW_2), then we iterate over different communities from TW_1 and TW_2 searching for the most similar community. That is defined by us, as the community c_i that contains the bigger number of products both in c_1 and c_2. These process is repeated for all the time windows considered, for example day, week, month, quarter, semester and year.

Mathematically, we have a set $TW = \{tw_1, tw_2, \ldots, tw_n\}$ of *Temporally Transactional Weighted Product Network* indexed by a particular time window. Each tw_i contains a set C_i of communities and each community C_i^j contains a set P_j of products associated.

We calculated the Jaccard Index [63] between the common products from two communities C_i^j and C_r^q. Then, we defined the similarity between those two communities $C_i^{j'}$ and $C_r^{q'}$ as the maximum Jaccard Index. Is important to note that community $C_r^{q'}$ is the most similar to community $C_i^{j'}$.

To obtain the most representative community over different time windows, is necessary to calculate the similarity between communities. Figure 4.12 depicts the similarity of three communities from a particular day $d1$ in comparison with the rest of the periods.

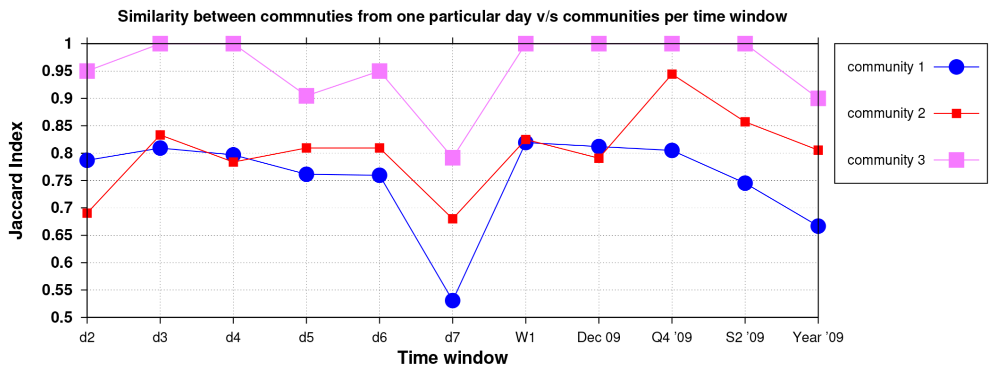


Figure 4.12: Similariity between three communities from day $d1$ compared to different time window.

Then, we repeated the process for the each time window. Comparing the three most important communities with the three most relevant communities of the following periods. Figure 4.13 shows the similarity between a particular week (November 7th, 2009 to November 13th, 2009) and the rest of the time window. In figure 4.14, is depicted the information from a month (November 2009), a quarter (October, 2009 to December, 2009), a semester (July, 2009 to December, 2009) and finally, a year (January, 2009 to December, 2009) as the pivot to make comparisons respectively.

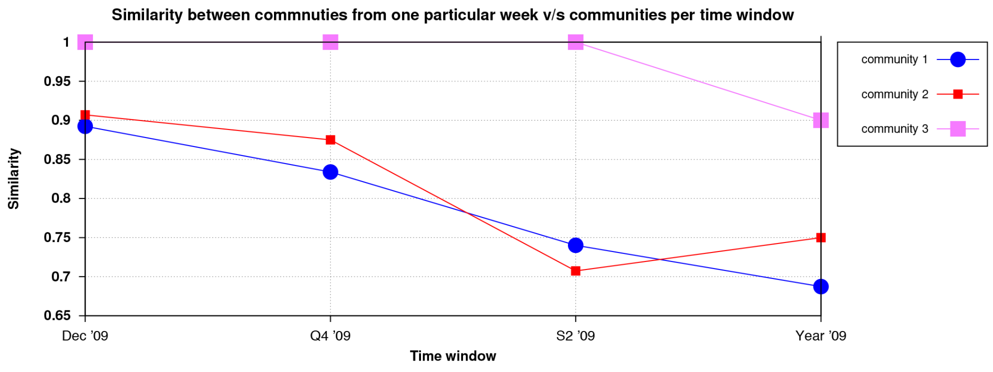
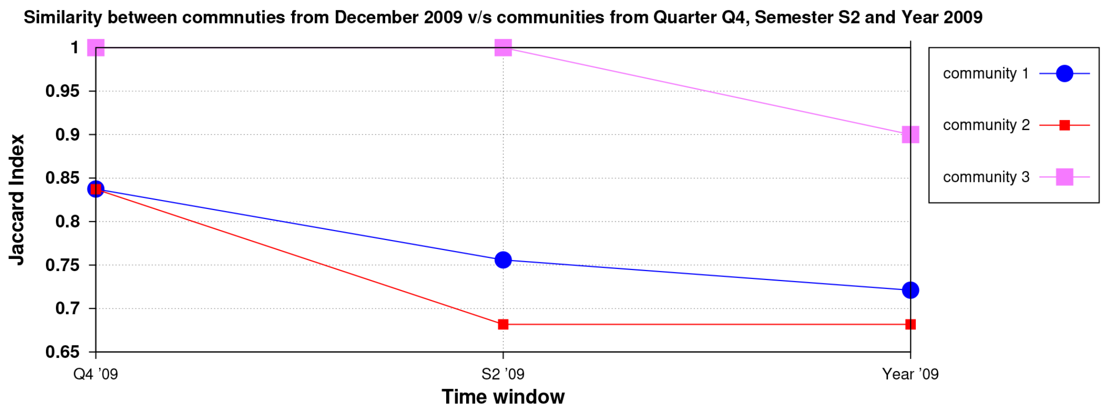
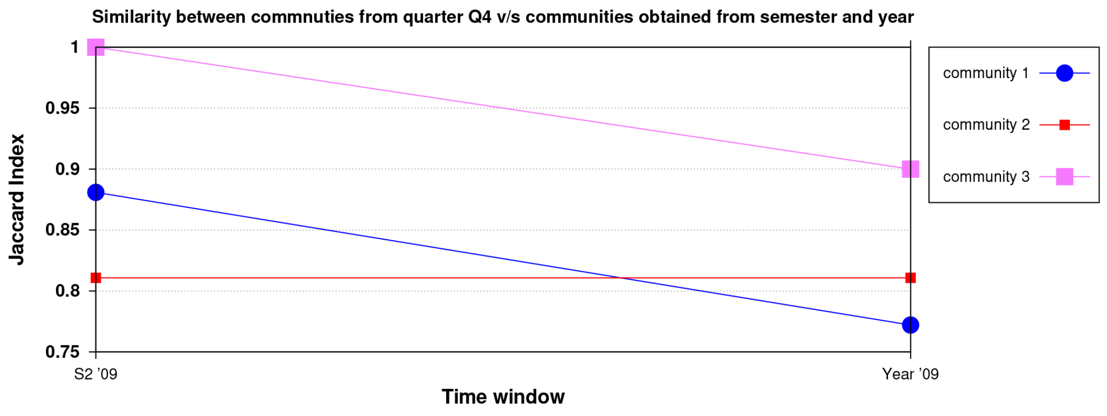


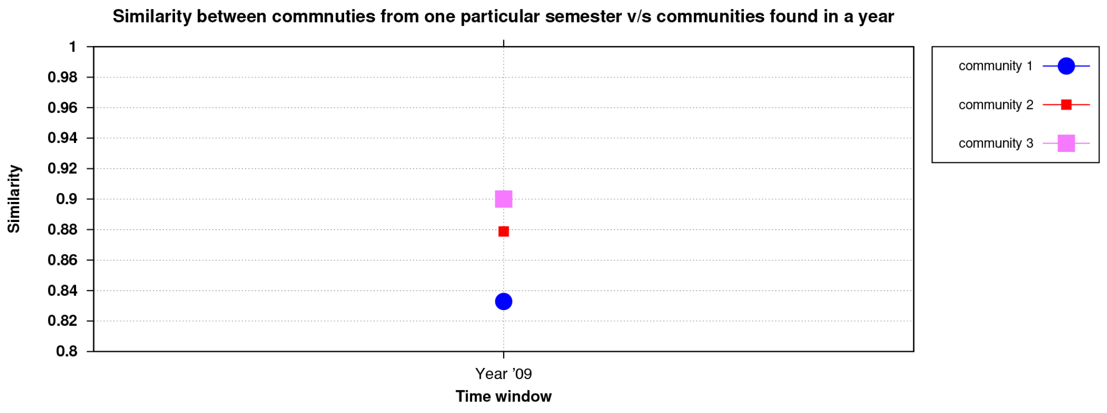
Figure 4.13: Similariity between three communities from week $w1$ compared to different time window.



(a) Month: December 2009



(b) Quarter: October to December 2009



(c) Semester: July to December 2009

Figure 4.14: Similarity between three communities over different time windows.

We presented this information to analysts, who studied the results and concluded that the best time window found is a *month*, because the percentage is relatively similar over time, also is a time window very manageable and aligned with the time window used by the retail. In fact, every model used by the retail is re–calibrated each *month*. Finally, this result is aligned with the results discovered by Ríos and Videla–Cavieres in [113].

4.10 Customer Characterization

We described in section 3.9 how customers will be characterized. In this section we will show the results from that characterization.

Once the communities are found we can begin to characterize customers based on their previous purchases. To do so, we need to identify a period of time to be analyzed. Next, choose a particular customer, then select the purchases realized over that period of time for that particular customer. After that, it is necessary to choose the *filtered temporally transactional weighted products network* to be used, also the algorithm used to find communities.

To depict the process we will realize an example, for two customers identified by the *ids* equal to 7343 and 56181. The period to be considered will be one month, March 2010. We chose the specific *filtered product network* with a threshold of 3030. The algorithm will be SLPA and the output results of SLPA chosen is the one with higher value of *modularity*. Table 4.9 depicts the quantity of purchases to each one of the communities identified by the customers selected, the three highest values are bolded.

Community (k)	**Number of purchases Q^k_{7343}**	**Number of purchases Q^k_{56181}**
Groceries	**142**	**45**
Soft Drinks	**11**	**12**
Cigarrettes	0	**63**
Yoghurts Brand A	4	2
Jelly Brand B	3	7
Nectar Juice Brand C	**10**	0
Juice Brand D	7	3
Nectar Juice Brand E	5	8
Nectar Juice Brand F	4	0
Compote Brand B	3	6
Powder Juice Brand G	6	8
Jelly Brand H	2	0
Semolina with Milk Brand B	1	5
Yoghurt Brand E	0	0
Cookies Brand J	7	6

Table 4.9: Communities identified and the number of purchases Q^k_i. For customer $i = 7343$ and $i = 56181$.

The total number $\sum_{k \in K} Q^k_i$ is equal to 205 and 165 for the customers 7343 and 56181 respectively. Table 4.10 shows the belonging degree to each one of the respective communities discovered.

Community (k)	**Belonging Degree Customer** $id = 7343$	**Belonging Degree Customer** $id = 56181$
Groceries	**0,692**	**0,272**
Soft Drinks	**0,0536**	**0.0727**
Cigarrettes	0	**0,381**
Yoghurts Brand A	0,019	0,012
Jelly Brand B	0,0146	0,042
Nectar Juice Brand C	**0,048**	0
Juice Brand D	0,034	0,018
Nectar Juice Brand E	0,024	0,048
Nectar Juice Brand F	0,019	0
Compote Brand B	0,014	0,036
Powder Juice Brand G	0,029	0,048
Jelly Brand H	0,00975	0
Semolina with Milk Brand B	0,004	0,03
Yoghurt Brand E	0	0
Cookies Brand J	0,034	0,036

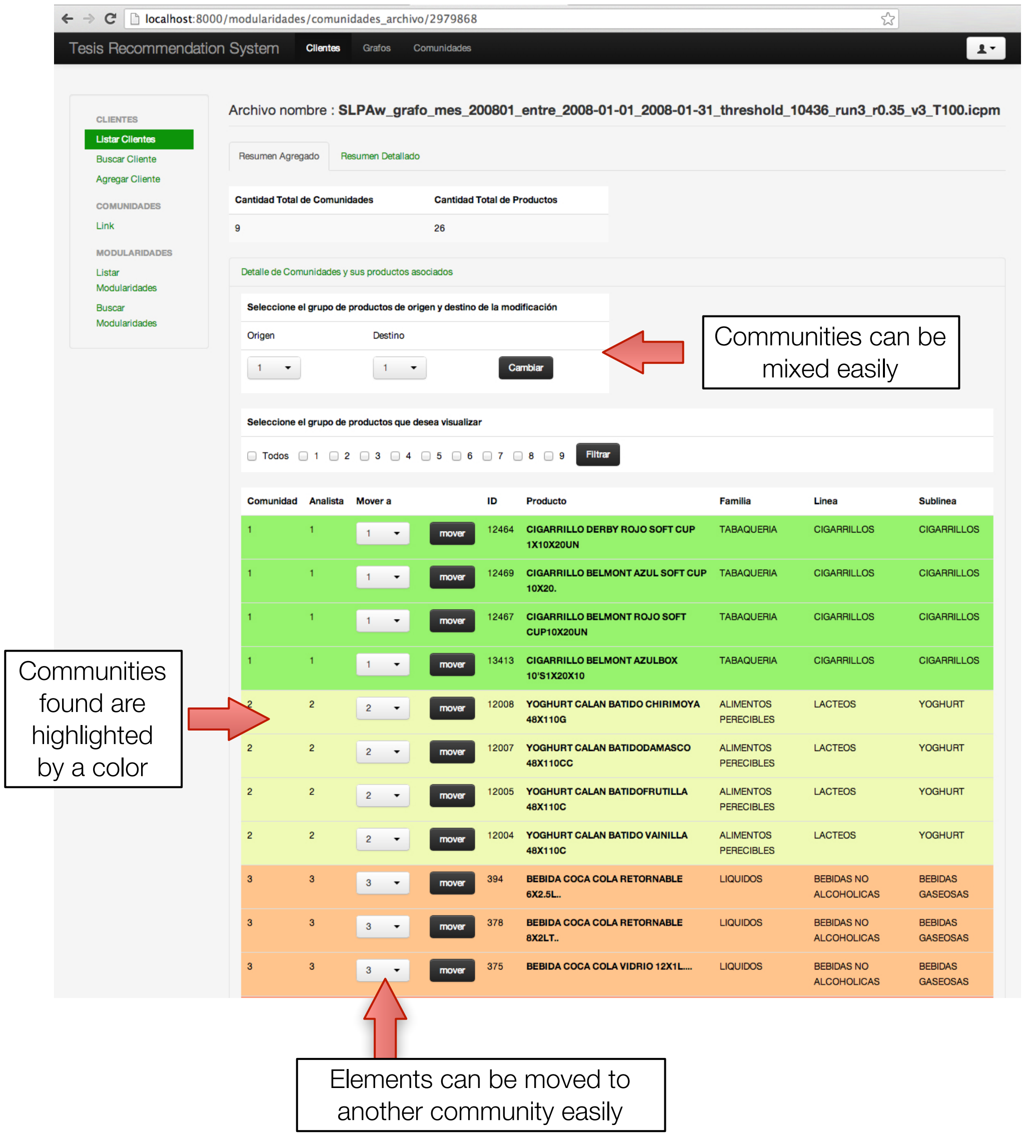
Table 4.10: Belonging degree of customer 7343 and 56181 to each one of the communities discovered.

We can say that costumer ($i = 7343$) prefer to buy *Groceries*, *Soft Drinks* and *Nectar Juice Brand C*, but does not purchase *Cigarretes*. On the other hand, customer $i = 56181$ prefer to purchase *Cigarrettes*, *Groceries* and *Soft Drinks* as its main communities. These method allow that analysts can examine the prefered groups of purchases for each customer, also allow to complement the already existing profiles. We can generate groups of customers that match some criteria, for example, all the customers that have a belonging degree of *Soft Drinks* community over certain threshold θ. Combinations of these criteria are allowed too.

4.11 Software

A system to present information to the analysts was built. The complete description about this system can be found in [124]. We will only depict a few images taken from the system.

Figure 4.15 depicts the communities discovered with the products that belong to them. These communties were found by the process explained in 3.8.


localhost:8000/modularidades/comunidades_archivo/2979868
Tesis Recommendation System
Clientes
Grafos
Comunidades
CLIENTES
Listar Clientes
Buscar Cliente
Agregar Cliente
COMUNIDADES
Link
MODULARIDADES
Listar Modularidades
Buscar Modularidades
Resumen Agregado
Resumen Detallado
Cambiar
Filtrar
Communities can be mixed easily
Communities found are highlighted by a color
Elements can be moved to another community easily

Archivo nombre : **SLPAw_grafo_mes_200801_entre_2008-01-01_2008-01-31_threshold_10436_run3_r0.35_v3_T100.icpm**

Cantidad Total de Comunidades	Cantidad Total de Productos
9	26

Detalle de Comunidades y sus productos asociados

Seleccione el grupo de productos de origen y destino de la modificación

Origen: 1　Destino: 1

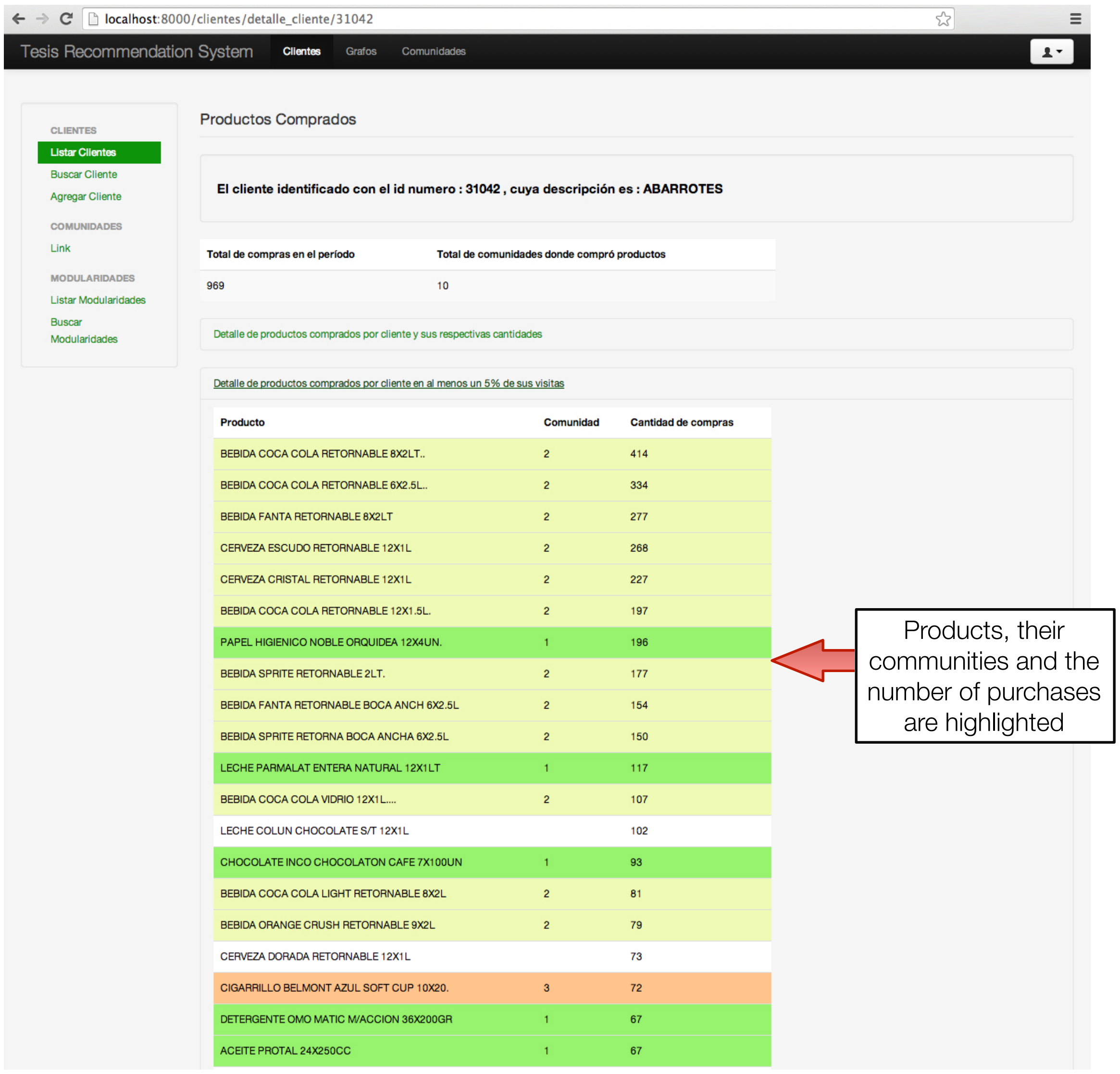
Seleccione el grupo de productos que desea visualizar

☐ Todos ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9

Comunidad	Analista	Mover a		ID	Producto	Familia	Linea	Sublinea
1	1	1	mover	12464	CIGARRILLO DERBY ROJO SOFT CUP 1X10X20UN	TABAQUERIA	CIGARRILLOS	CIGARRILLOS
1	1	1	mover	12469	CIGARRILLO BELMONT AZUL SOFT CUP 10X20.	TABAQUERIA	CIGARRILLOS	CIGARRILLOS
1	1	1	mover	12467	CIGARRILLO BELMONT ROJO SOFT CUP10X20UN	TABAQUERIA	CIGARRILLOS	CIGARRILLOS
1	1	1	mover	13413	CIGARRILLO BELMONT AZULBOX 10'S1X20X10	TABAQUERIA	CIGARRILLOS	CIGARRILLOS
2	2	2	mover	12008	YOGHURT CALAN BATIDO CHIRIMOYA 48X110G	ALIMENTOS PERECIBLES	LACTEOS	YOGHURT
2	2	2	mover	12007	YOGHURT CALAN BATIDODAMASCO 48X110CC	ALIMENTOS PERECIBLES	LACTEOS	YOGHURT
2	2	2	mover	12005	YOGHURT CALAN BATIDOFRUTILLA 48X110C	ALIMENTOS PERECIBLES	LACTEOS	YOGHURT
2	2	2	mover	12004	YOGHURT CALAN BATIDO VAINILLA 48X110C	ALIMENTOS PERECIBLES	LACTEOS	YOGHURT
3	3	3	mover	394	BEBIDA COCA COLA RETORNABLE 6X2.5L..	LIQUIDOS	BEBIDAS NO ALCOHOLICAS	BEBIDAS GASEOSAS
3	3	3	mover	378	BEBIDA COCA COLA RETORNABLE 8X2LT..	LIQUIDOS	BEBIDAS NO ALCOHOLICAS	BEBIDAS GASEOSAS
3	3	3	mover	375	BEBIDA COCA COLA VIDRIO 12X1L....	LIQUIDOS	BEBIDAS NO ALCOHOLICAS	BEBIDAS GASEOSAS

Figure 4.15: Set of products with their respective community.

The products purchased and the communities to which they belong are depicted in figure 4.16. This information is highlighted in order to make this process easier to the analyst.



Productos Comprados

El cliente identificado con el id numero : 31042 , cuya descripción es : ABARROTES

Total de compras en el período	Total de comunidades donde compró productos
969	10

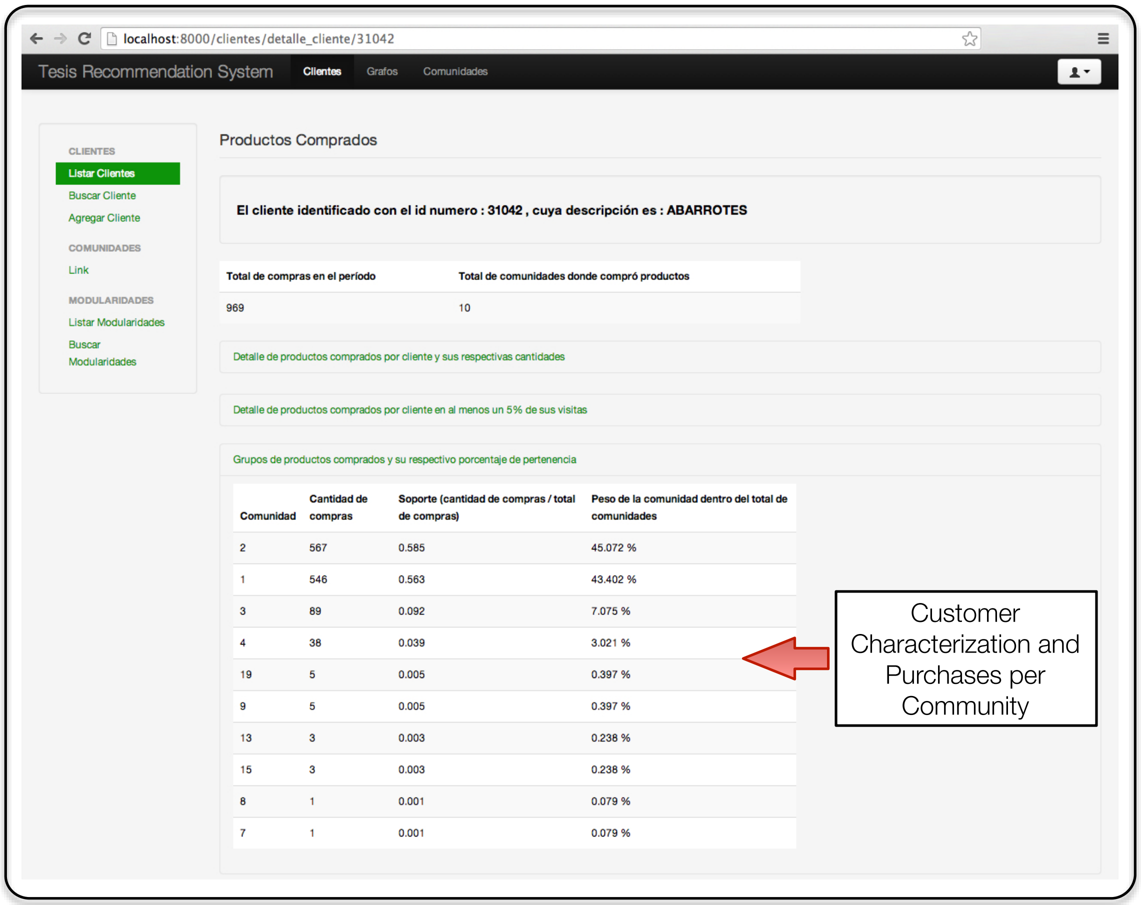
Detalle de productos comprados por cliente y sus respectivas cantidades

Detalle de productos comprados por cliente en al menos un 5% de sus visitas

Producto	Comunidad	Cantidad de compras
BEBIDA COCA COLA RETORNABLE 8X2LT..	2	414
BEBIDA COCA COLA RETORNABLE 6X2.5L..	2	334
BEBIDA FANTA RETORNABLE 8X2LT	2	277
CERVEZA ESCUDO RETORNABLE 12X1L	2	268
CERVEZA CRISTAL RETORNABLE 12X1L	2	227
BEBIDA COCA COLA RETORNABLE 12X1.5L.	2	197
PAPEL HIGIENICO NOBLE ORQUIDEA 12X4UN.	1	196
BEBIDA SPRITE RETORNABLE 2LT.	2	177
BEBIDA FANTA RETORNABLE BOCA ANCH 6X2.5L	2	154
BEBIDA SPRITE RETORNA BOCA ANCHA 6X2.5L	2	150
LECHE PARMALAT ENTERA NATURAL 12X1LT	1	117
BEBIDA COCA COLA VIDRIO 12X1L....	2	107
LECHE COLUN CHOCOLATE S/T 12X1L		102
CHOCOLATE INCO CHOCOLATON CAFE 7X100UN	1	93
BEBIDA COCA COLA LIGHT RETORNABLE 8X2L	2	81
BEBIDA ORANGE CRUSH RETORNABLE 9X2L	2	79
CERVEZA DORADA RETORNABLE 12X1L		73
CIGARRILLO BELMONT AZUL SOFT CUP 10X20.	3	72
DETERGENTE OMO MATIC M/ACCION 36X200GR	1	67
ACEITE PROTAL 24X250CC	1	67

Figure 4.16: Historical purchases and the associated communities to each product.

Finally, the information of purchases grouped by communities is presented to the analyst. The customer characterization that we presented in section 4.10 is depicted also. Figure 4.17 depicts an example from a real customer.



Productos Comprados

El cliente identificado con el id numero : 31042 , cuya descripción es : ABARROTES

Total de compras en el período	Total de comunidades donde compró productos
969	10

Detalle de productos comprados por cliente y sus respectivas cantidades

Detalle de productos comprados por cliente en al menos un 5% de sus visitas

Grupos de productos comprados y su respectivo porcentaje de pertenencia

Comunidad	Cantidad de compras	Soporte (cantidad de compras / total de compras)	Peso de la comunidad dentro del total de comunidades
2	567	0.585	45.072 %
1	546	0.563	43.402 %
3	89	0.092	7.075 %
4	38	0.039	3.021 %
19	5	0.005	0.397 %
9	5	0.005	0.397 %
13	3	0.003	0.238 %
15	3	0.003	0.238 %
8	1	0.001	0.079 %
7	1	0.001	0.079 %

Figure 4.17: Customer characterization based on communities.

4.12 Personalized Recommendations

A synthetic test to prove the quality of our recommendations was done, due to the impossibility to deliver the recommendation to a real customer. It is important to note that we are adapting our recommendations to the products that were recommended by the retailer, this is a limitation to our method and a underestimation of our results.

In this section we will describe how the recommendations were generated for each customer in our system. As we present in section 3.10 there are three main stages in the process. We will review each one in detail.

Available Products

Beginning from the information obtained from the retailer we generate the subset of products, $A = \{p_1, p_2, \ldots, p_m\}$, that can be included into the discount coupon. This is a small set of the total set of products available. In fact, 5359 products were sold in the month of study and only 647 where available to be recommended.

Customer Characterization

For each customer k, we take the set of transactions in the month T_i. From these transactions we calculate the belonging degree to each discovered community as we described in section 3.9. The main problem with this approach was that the best graph of overlapped communities contains 312 products of which only 15 were available to be recommended in the next period (month T_{i+1}). This fact forces us to add additional efforts for generating recommended products.

Coupon Generation

To improve the quality of the products recommended, we analyzed the *co–purchased product network*. This network represents each product and its corresponding related items purchased together. Based on this fact, we marked, in the co–purchased product network, each product available to be recommended in the next month.

Next, we chose the products purchased in the month T_i by the customer k and obtained the most related products that match with the available ones, descending ordered by the edge weight. We applied certain restrictions to the products such as, products should become from different sub-categories, lines and family respectively. In certain cases, recommended products should not have been purchased in month T_i by customer k.

In figure 4.18 the whole process is depicted. Blue nodes represent products purchased by the customer in the month T_i and purple nodes represent products available to be included in the recommendation for the next month T_{i+1}.

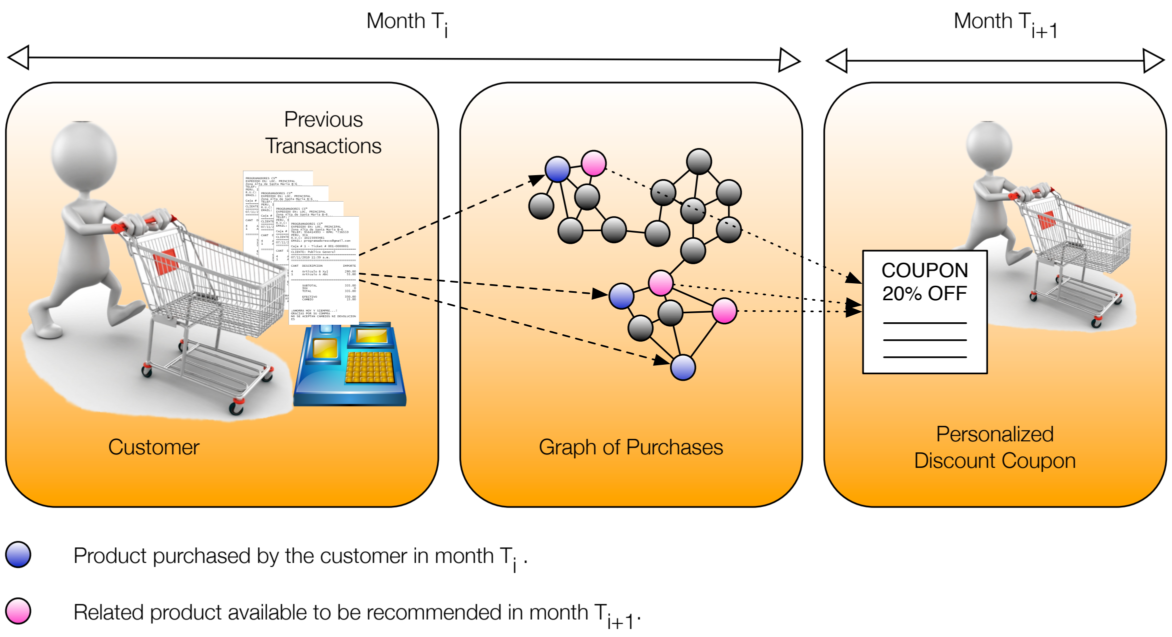

Month T_i
Month T_{i+1}
Previous Transactions
Customer
Graph of Purchases
COUPON 20% OFF
Personalized Discount Coupon
Product purchased by the customer in month T_i .
Related product available to be recommended in month T_{i+1}.

Figure 4.18: Recommendation Process.

The method currently used by the retailer to generate recommendations is based in *association rules*, they generate recommendations from purchased products as frequent items and their associated products from association rules.

4.12.1 Results from Personalized Recommendations

In order to calculate the quality of our results. We calculated the recommendations using *January* 2011 as a pivot month and our measure the results comparing our recommendations with the products bought by the customer k in *February* 2011. It is important to note that the same group of customer were used in order that the results can be compared between each other.

First we calculated the performance of the recommendations offered by the retailer and we obtained that on average hits 1.2116 of a total of 48, which represents the 2.52% of the recommended products.

We calculate the performance of our recommendations. First, we limited the recommendation, forcing that both products (origin and recommended) have to be purchased in the previous month T_i. The results were 1.95 as an average. Our second limitation was that products can not proceed from the same sub-category, then line and finally the same family. Results obtained were: 2.8699, 2.8623 and 2.8636 respectively.

The last process was obtained from mixing the best community, in terms of the number of products and modularity, and the co–purchased product network. The results obtained in average were : 2.9152 and 1.956 products.

Table 4.11 summarize the results obtained by each of the approaches proposed.

Recommendation	**Hits**	**Percentage**	**Difference to Retailer (%)**
Retailer	1.2116	2.52%	0%
Both Purchased	1.95	4.06%	60.94
Purchased and Not	2.0448	4.26%	68.76
Subcategories Limitated	2.8699	5.97%	136.86
Line Limitated	2.8623	5.96%	136.24
Family Limitated	2.8636	5.97%	136.34
No Limitation	2.9188	6.08%	140.90
Communities	2.9152	6.07%	140.60
Communities Both Purchased	1.9560	4.08%	61.43

Table 4.11: Different approaches and their results.

It is clear that our method outperforms the current method used by the retailer. In fact, if we take the retailer as a reference and calculate the incomes from our best two method, from co–purchased product network without limitation in the recommended product and the mix from communities and co–purchased product network. The difference is 460% and 430% respectively. This fact represents potential incomes that can be generated enhancing the recommendations with this method. Maintaining the same set of available products to be recommended.

These results show that exists a tremendous opportunity to improve the quality of the recommendations following this path.

In this chapter we have presented an application of our novel methodology, in order to develop market basket analysis and customer characterization. This process allows analysts, the generation of groups of products that are related each other because are purchased in the same buying opportunity. Also, allows to obtain a better understanding of the customer, based on their previous purchases. We presented our approach to generate personalized recommendations and our results that outperforms current recommendation system.

Chapter 5

Conclusions and future work

"In the end, it's not the years in your life that count. It's the life in your years."

Abraham Lincoln

It is commonly known that retaining a customer is cheaper than acquiring a new one. This fact motivates companies to develop new ways of interacting with their clients in order to get a better understanding of them. Most Chilean retailers currently cluster consumers based in the amount they expend within a certain period of time. On the other hand, literature recommends generating groups of products found in market baskets, then associate customers to groups according to their previous purchases. We attempted this problem using traditional techniques, such as K-Means, SOM and market basket analysis, but results were impractical.

That first approach did not go well, so a novel approach is developed. The method included graph mining techniques, for performing market basket analysis, and overlapping community detection algorithms as frequent item set discovery technique. It is also presented way to extract useful information from millions of product sales transactions.

We introduce the concept of *Temporally Transactional Weighted Product Network* which allows carrying different information needed by retailers and other organizations. These networks can be sliced by a mix of different criteria, such as time window (daily, weekly, monthly, etc), a particular store, a cluster of stores or the entire *retail* transactions.

We propose an objective methodology for thresholds and filters setup in order to reduce noisy data (independent from the nodes and edges quantity). Firstly, we proposed the *top three heavy edges threshold* method. An unsupervised threshold that only depends on the data available. Secondly, we defined filters which are a proportion of the *top three heavy edges threshold*. Subsequently, we propose the application of overlapping community detection algorithms as a manner to generate frequent item sets.

Our approach has shown that the method presented can be used as a valid technique to discover frequent item sets present in transactional data. It is important to remark that the information delivered is very useful for retail, depicting relationships that were not obvious for the analyst of the *retail*. For example, that people is loyal to a particular brand instead of a mix of brands as expected.

The main contribution is a methodology that can be used as a framework for completing customer profiles based on transactional data. This methodology not only applies to retailers. It can also be applied in other industries that need to generate groups built from transactional data. The steps are depicted in chapter 3, amongst the advantages we can mention that each part can be modified according specific needs.

The main objective of this thesis, to enhance the recommendation system used in a wholesale supermarket chain applying graph mining techniques over million of transactional entries, was achieved. The methodology required was introduced and developed in this work. Indeed, in section 4.10 we present a real case of customer profiling using the proposed techniques, which allows to effectively characterize customers based on their previous purchased, and in section 4.12 we present our approach to generate personalized recommendations.

We have specific objectives also completed. Next, we will explain each one of them showing that they were totally fulfilled and indicate their contribution to this work.

In section 2.6, a research about the state–of–the–art in market basket analysis and its techniques applied over transactional data was shown. We introduced a definition of the concept, the data used to apply market basket analysis and the meaning of association rules. We presented the most common techniques to generate frequent item sets, such as apriori algorithm and the principal measures to evaluate the quality of rules generated.

Continuing, in section 2.8.1 we presented the algorithms related to overlapping community detection. Existing algorithms were classified in five major classes. The importance of this section is that we determined the best two algorithms to be used in the rest of this thesis.

Chapter 4 depicts the application of the methodology described in chapter 3 over real retail data. Section 4.3 shows traditional clustering and market basket analysis techniques, the results obtained by this classic approach made unfeasible the generation of meaningful groups of products, forcing us to move. Then, in section 4.4 we described the application of our methodology, depicting each step performed over the process. Finally in section 4.12 we show different approaches and the versatility of our recommendation method, with its results.

We benchmarked using state–of–the–art algorithms COPRA and SLPA. We also applied state–of–the–art frequent item set algorithms such as K-means, SOM and Apriori. The benchmark was done using data of two supermarket chain, processing over 400 million records.

However, with our own methodology we were able to produce meaningful and useful frequent item sets. For example, using K-means we obtained mainly 2 representative clusters and one of these concentrating 93% of the products (around 14,000), versus our method, which found –in the same data– 30 clusters (communities) with an average of 7 products. It is clear that 7 products is a manageable number for any analyst. Another example is the Apriori algorithm results which gave several rules with very low support and confidence, making unfeasible to generate groups of products that can be easily labeled.

We asked the *retail* business experts to compare our methodology results with traditional market basket analysis algorithms like Apriori, K-Means and SOM. We discovered that results from traditional techniques were far from being useful, because clusters were formed by huge amounts of mixed products, thus, a segmentation based on these results was meaningless. The main reason was the spars nature of supermarket data and the big size of information involved.

We completed the customer profile based on mixing previous purchases and the communities obtained from the developed methodology. We characterized the customer based on the belonging degree to each community as was explained in section 4.10.

Finally, in section 4.12 we depict process to generate personalized customer recommendations and their results. Results are amazing, we increase the number of hits by a 140%, using the same available products. Showing that an enhancement to the recommendation process its possible and could mean revenues over the 430% with respect today recommendations.

Future Work

Several interesting directions for future work can be considered and noted. In this thesis we made a characterization and completion of the customer data and a enhancement of the current recommendation process based on previous purchases and overlapped communities found from industry's transactional information.

In future work, one could try to include products from overlapped communities that can enhance the results obtained from the recommendation process. The idea is to include products in the recommendation that are deeply related because they belong to the same community.

In this work, we used undirected weighted networks, but for some retailers could be useful to use directed weighted or directed unweighted networks. In the same line, non-overlapped communities could work. Comparing the results obtained in terms of number of communities, number of products, stability, just to name a few.

Detecting key elements inside communities is an interesting topic for future work. This could be done through the application of algorithms for discovering key members, such as HITS [71], PageRank [100] to detect products that are relevant inside a community.

It might be important to study the relation between important products inside a community and breaks of stock. Evaluating the effect on related products that may not be purchased because the main product was sold out, it would be interesting to investigate this fact and quantify it in economic terms. The idea of applying the proposed methodology to another retailer such as supermarket, convenience store or home improving, in order to prove the proposed methodology with a new case of study, is significant.

From the recommendations process, would be great to develop a pilot with the new recommendations, this would allow the real measure of the quality of our recommendations instead of the synthetic test that we made.

There is also a big opportunity for taking advantage of the communities that can be discovered with our method for generating novel strategies of in-store offers that are relevant to the customer.

Acknowledgments

The author would like to acknowledge the support of the Chilean Millennium Institute of Complex Engineering Systems (ICM: P05-004-F FIN. ICM-FIC). Also would like to acknowledge to the Business Intelligence Research Center (CEINE) for their continuous support.

Journal Publications

[1] Videla-Cavieres, I. F., & Ríos, S. A. (2014). Extending market basket analysis with graph mining techniques: A real case. *Expert Systems with Applications*, 41(4), 1928-1936.

Conference Publications

[1] Ríos, Sebastián A., and Ivan F. Videla-Cavieres. Generating groups of products using graph mining techniques. *Procedia Computer Science*, 35(2014): 730-738.

Bibliography

[1] Zeinab Abbassi and Sihem Amer-Yahia. Getting recommender systems to think outside the box. *Proceedings of the third ACM conference on Recommender systems. ACM*, page 285, 2009.

[2] Ajith Abraham. BUSINESS INTELLIGENCE FROM WEB USAGE MINING. *Journal of Information & Knowledge Management*, 2003.

[3] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005.

[4] G. Adomavicius and a. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005.

[5] Gediminas Adomavicius and Alexander Tuzhilin. Using Data Mining Methods to Build Customer Profiles. *Research Feature*, 2001.

[6] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering*, 5(6), 1993.

[7] Rakesh Agrawal and R Srikant. Fast algorithms for mining association rules. *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 1–32, 1994.

[8] Yong-Yeol Ahn, James P Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–4, August 2010.

[9] Sameer Antani, Rangachar Kasturi, and Ramesh Jain. A survey on the use of pattern recognition methods for abstraction, indexing and retrieval of images and video. *Pattern Recognition*, 35(4):945–965, April 2002.

[10] M Antonini and M Barlaud. Image Coding Using Wavelet Transform. *IEEE Transactions on Image Processing*, (205-220), 1992.

[11] A. Arenas, L. Danon, A. Díaz-Guilera, P. M. Gleiser, and R. Guimerà. Community analysis in social networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, 38(2):373–380, March 2004.

[12] Lars Backstrom, Paolo Boldi, and Marco Rosa. Four degrees of separation. *Proceedings of the 3rd Annual ACM Web Science Conference*, 2012.

[13] M Balabanović and Yoav Shoham. Fab: Content-Based, Collaborative Recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[14] Jeffrey Baumes, Mark Goldberg, and Malik Magdon-ismail. Efficient Identification of Overlapping Communities. 2005.

[15] Jeffrey Baumes and MK Goldberg. FINDING COMMUNITIES BY CLUSTERING A GRAPH INTO OVERLAPPING SUBGRAPHS. *IADIS AC*, pages 97–104, 2005.

[16] Roberto J. Bayardo Jr and R Agrawal. Mining the most interesting rules. *Knowledge discovery and data mining*, pages 145–154, 1999.

[17] A Ben-Dor, R Shamir, and Z Yakhini. Clustering Gene Expression Patterns. *Journal of Computational Biology*, 6(3-4), 1999.

[18] MJ Berry and GS Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management.* 2004.

[19] Robert C. Blattberg, Byung-Do Kim, and Scott A. Neslin. *Database Marketing: Analyzing and Managing Customers.* Springer, December 2010.

[20] B Bollobâas, R Kozma, and D Miklós. *Handbook of Large-Scale Random Networks.* 2009.

[21] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. *Proceedings of the 1997 ACM SIGMOD international conference on Management of data - SIGMOD '97*, pages 255–264, 1997.

[22] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph structure in the Web. *Computer Networks*, 33(1-6):309–320, June 2000.

[23] JR Bult and Tom Wansbeek. Optimal selection for direct mail. *Marketing Science*, 14(4):378–394, 1995.

[24] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, pages 1–23, 2000.

[25] Robin Burke. Hybrid Recommender Systems : Survey and Experiments. *User modeling and user-adapted interaction*, 2002.

[26] Tadeusz Calinski and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1—-27, 1974.

[27] Remy Cazabet, Frederic Amblard, and Chihab Hanachi. Detection of Overlapping Communities in Dynamical Social Networks. *2010 IEEE Second International Conference on Social Computing*, pages 309–314, August 2010.

[28] Deepayan Chakrabarti and Christos Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys (CSUR)*, 38(March), 2006.

[29] PK Chan and W Fan. Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems' Special Issue on Data Mining.*, pages 1–17, 1999.

[30] PK Chan and SJ Stolfo. Toward Scalable Learning with Non-uniform Class and Cost Distributions : A Case Study in Credit Card Fraud Detection. *KDD*, 1998.

[31] JA Chevalier, AK Kashyap, and PE Rossi. Why don't prices rise during periods of peak demand? Evidence from scanner data. 3, 2000.

[32] Ibrahim Cil. Consumption universes based supermarket layout through association rule mining and multidimensional scaling. *Expert Systems with Applications*, 39(10):8611–8625, August 2012.

[33] Aaron Clauset, M. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111, December 2004.

[34] DJ Cook and LB Holder. *Mining Graph Data.* John Wiley & Sons, INC. Publication, 2006.

[35] TM Cover and JA Thomas. *Elements of Information Theory.* 2012.

[36] Lautaro Cuadra. Metodología de búsqueda de sub-comunidades mediante análisis de redes sociales y minería de datos. Master's thesis, University of Chile, 2011.

[37] Instituto Nacional de Estadísticas. ÍNDICE DE VENTAS DE SUPERMERCADOS (ISUP), BASE PROMEDIO AÑO 2009 CIFRAS DESDE ENERO 2012 A ABRIL 2013. Technical report, Instituto Nacional de Estadísticas, INE, 2013.

[38] Scott Deerwester, ST Dumais, and TK Landauer. Indexing by Latent Semantic Analysis. *JASIS*, 41(6):391–407, September 1990.

[39] El Mercurio Economía Y Negocios. Supermercados crecen cerca de 6% en 2013 con ventas por us$ 45 millones diarios [online: accessed 15-mar-2015]. *http://www.economiaynegocios.cl/noticias/noticias.asp?id=115963*, 2014.

[40] Magdalini Eirinaki and Michalis Vazirgiannis. Web mining for web personalization. *ACM Transactions on Internet Technology*, 3(1):1–27, February 2003.

[41] M Faloutsos, P Faloutsos, and C Faloutsos. On Power-Law Relationships of the Internet Topology. *ACM SIGCOMM Computer Communication*, 29(4):251–262, 1999.

[42] Laurene Fausett. *Fundamentals of Neural Networks: Arquitectures, Algorithms, and Applications.* Pearson Education India, 2006.

[43] Usama Fayyad, G Piatetsky-Shapiro, and Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. *AI magazine*, 17(3):37–54, 1996.

[44] A Felfernig and R Burke. Constraint-based recommender systems: technologies and research issues. *Proceedings of the 10th international conference on Electronic commerce. ACM*, 2008.

[45] Santo Fortunato. Community detection in graphs. *Physics Reports*, 2010.

[46] Santo Fortunato and M Barthelemy. Resolution Limit in Community Detection. *Proceedings of the National Academy of Sciences*, 104(1):36, 2007.

[47] LC Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 1, 1977.

[48] LC Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1:215–239, 1979.

[49] MM Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. Mining data streams: a review. *ACM Sigmod Record*, 34(2):18–26, 2005.

[50] M Girvan and M E J Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–6, June 2002.

[51] K Goldberg, T Roeder, D Gupta, and C Perkins. Eigentaste : A Constant Time Collaborative Filtering Algorithm. *Information Retrieval*, pages 133–151, 2001.

[52] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics.* 1988.

[53] Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, October 2010.

[54] Steve Gregory. Fuzzy overlapping communities in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(02):P02017, February 2011.

[55] PM Guadagni and JDC Little. A logit model of brand choice calibrated on scanner data. 2(3):203–238, 1983.

[56] Roger Guimerà, Marta Sales-Pardo, and Luís Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101, August 2004.

[57] Isabelle Guyon and A Elisseeff. An Introduction to Variable and Feature Selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[58] R. L. Hair Jr, J. F., Anderson, R. E., & Tatham. *Multivariate Data Analysis: with readings.* Macmillan Publishing Co., Inc.., seventh ed edition.

[59] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques.* 2006.

[60] JA Hartigan and MA Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[61] Janny C Hoekstra, Peter S.H Leeflang, and Dick R Wittink. The Customer Concept: The Basis for a New Marketing Paradigm. *Journal of Market Focused Management*, 4:43–76, 1999.

[62] IBM. What is big data? [online: accessed 26-06-2013]. *http://www-01.ibm.com/software/data/bigdata/*, March 2013.

[63] Paul Jaccard. *Distribution de la Flore Alpine: dans le Bassin des dranses et dans quelques régions voisines.* Rouge, 1901.

[64] Josh James. How much data is created every minute? [online: accessed 26-06-2013]. *http://www.domo.com/blog/2012/06/how-much-data-is-created-every-minute/*, June 2012.

[65] D Jannach, M Zanker, A Felfernig, and G Friedrich. *Recommender systems: an introduction.* 2010.

[66] I Jolliffe. *Principal Component Analysis.* 2005.

[67] Brian Karrer, Elizaveta Levina, and M. E. J. Newman. Robustness of community structure in networks. *Physical Review E*, 77(4):046119, April 2008.

[68] NK Kasabov. *Foundations of neural networks, fuzzy systems and knowledge engineering.* Marcel Alencar, 1996.

[69] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, (January):263–286, 2001.

[70] Hyea Kyeong Kim, Jae Kyeong Kim, and Qiu Yi Chen. A product network analysis for extending the market basket analysis. *Expert Systems with Applications*, 39(8):7403–7410, June 2012.

[71] JM Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, (September):604–632, 1999.

[72] Aaron Knott, Andrew Hayes, and Scott a. Neslin. Next-product-to-buy models for cross-selling applications. *Journal of Interactive Marketing*, 16(3):59–75, January 2002.

[73] T Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[74] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM*, 2008.

[75] Yehuda Koren, Robert Bell, and Chris Volinsky. MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS. *Computer*, pages 42–49, 2009.

[76] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to parallel computing*, volume 110. Benjamin/Cummings Redwood City, 1994.

[77] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, pages 1–9, 2009.

[78] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, March 2009.

[79] Andrea Lancichinetti, Filippo Radicchi, José J Ramasco, and Santo Fortunato. Finding statistically significant communities in networks. *PloS one*, 6(4):e18961, January 2011.

[80] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. *Proceeding of the 17th international conference on World Wide Web - WWW '08*, page 695, 2008.

[81] Jure Leskovec, KJ Lang, and M Mahoney. Empirical comparison of algorithms for network community detection. *Proceedings of the 19th international conference on World wide web*, pages 631–640, 2010.

[82] Weiyang Lin, SA Alvarez, and C Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data mining and knowledge discovery*, pages 83–105, 2002.

[83] Greg Linden, Brent Smith, and Jeremy York. Amazon.com Recommendations Item-to-Item Collaborative Filtering. *Internet Computing, IEEE*, (February), 2003.

[84] Wangzhong Lu, J. Janssen, E. Milios, N. Japkowicz, and Yongzheng Zhang. Node similarity in the citation graph. *Knowledge and Information Systems*, 11(1):105–129, April 2006.

[85] Anne W. Magi. Share of wallet in retailing: the effects of customer satisfaction, loyalty cards and shopper characteristics. *Journal of Retailing*, 79(2):97–106, January 2003.

[86] H Mannila, H Toivonen, and AI Verkamo. Efficient Algorithms for Discovering Association Rules. 7, 1994.

[87] CD Manning, P Raghavan, and H Schütze. *Introduction to Information Retrieval.* Number c. 2008.

[88] S Milgram. The small-world problem. *Psychology today*, (May 1967), 1967.

[89] GW Milligan and MC Cooper. AN EXAMINATION OF PROCEDURES FOR DETERMINING THE NUMBER OF CLUSTERS IN A DATA SET. *Psychometrika*, 50(2):159–179, 1985.

[90] Bamshad Mobasher, Robert Cooley, and Jaideep Srivastava. Automatic Personalization Based on Web Usage Mining. *Communications of the ACM*, 43(8):142–151, 2000.

[91] AL Montgomery. Creating micro-marketing pricing strategies using supermarket scanner data. *Marketing Science*, 1997.

[92] Cristopher Moore and MEJ Newman. Epidemics and percolation in small-world networks. *Physical Review E*, 2000.

[93] Ricardo Muñoz. DISEÑO, DESARROLLO Y EVALUACIÓN DE UN ALGORITMO PARA DETECTAR SUB-COMUNIDADES TRASLAPADAS USANDO ANÁLISIS DE REDES SOCIALES Y MINERÍA DE DATOS. Master's thesis, University of Chile, 2013.

[94] Banco Mundial. Pib (us $ a precios actuales) [online: accessed 15-mar-2015]. *http://datos.bancomundial.org/indicador/NY.GDP.MKTP.CD*, 2014.

[95] T Nepusz, A Petróczi, L Négyessy, and F Bazsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 1:1–13, 2008.

[96] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, February 2004.

[97] M. Newman and Juyong Park. Why social networks are different from other types of networks. *Physical Review E*, 68(3):036122, September 2003.

[98] V Nicosia and G Mangioni. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment (JSTAT)*, March 2009, 2009.

[99] Tore Opsahl, Filip Agneessens, and John Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3):245–251, July 2010.

[100] L Page, S Brin, R Motwani, and T Winograd. The PageRank Citation Ranking: Bringing Order to the Web. pages 1–17, 1999.

[101] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–8, June 2005.

[102] Juyong Park and M. Newman. Origin of degree correlations in the Internet and other networks. *Physical Review E*, 68(2):026112, August 2003.

[103] M Perkowitz and Oren Etzioni. Adaptive web sites: Automatically synthesizing web pages. *Proceedings of the National Conference on Artificial Intelligence*, pages 727–732, 1998.

[104] JT Plummer. The concept and application of life style segmentation. *the Journal of Marketing*, 1974.

[105] L Portnoy, E Eskin, and S Stolfo. Intrusion detection with unlabeled data using clustering. *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security*, pages 5–8, 2001.

[106] J. Ross Quinlan. Introduction of decision trees. *Machine learning*, 1:81–106, 1986.

[107] Troy Raeder and Nitesh V. Chawla. Modeling a Store's Product Space as a Social Network. In *2009 International Conference on Advances in Social Network Analysis and Mining*, pages 164–169. IEEE, July 2009.

[108] S Redner. How popular is your paper? An empirical study of the citation distribution. *The European Physical Journal B - Condensed Matter and Complex Systems*, 200(i):1–4, 1998.

[109] FF Reichheld, RG Markey Jr, and C Hopton. The loyalty effect - the relationship between loyalty and profits. *European Business Journal*, 2000.

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

[111] SA Rios. *A study on web mining techniques for off-line enhancements of web sites*. PhD thesis, 2007.

[112] SA Ríos and Ricardo Muñoz. Dark Web portal overlapping community detection based on topic models. *Proceedings of the ACM SIGKDD Workshop on Intelligence and Security Informatics*, pages 1–7, 2012.

[113] Sebastián Ríos and Ivan F. Videla-Cavieres. Generating Groups of Products Using Graph Mining Techniques. *Procedia Computer Science*, 35:730–738, 2014.

[114] Gert Sabidussi. THE CENTRALITY INDEX OF A GRAPH. *Psychometrika*, pages 581–603, 1966.

[115] S. Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. Technical report, 1990.

[116] B Sarwar, George Karypis, J Konstan, and J Riedl. Application of dimensionality reduction in recommender system-a case study. 2000.

[117] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. *EC '00 Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167, 2000.

[118] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th international conference on World Wide Web*, 2001.

[119] B Sharp and Anne Sharp. Loyalty programs and their impact on repeat-purchase loyalty patterns. *International Journal of Research in Marketing*, 14(5):473–486, 1997.

[120] Olaf Sporns. Structure and function of complex brain networks. *Dialogues in clinical neuroscience*, 15(3):247–62, September 2013.

[121] Frank M Thiesing, Ulrich Middelberg, and Oliver Vornberger. A NEURAL NETWORK APPROACH FOR PREDICTING THE SALE OF ARTICLES IN SUPERMARKETS. *3rd European Congress on Intelligent Techniques and Soft Computing*, pages 31–36, 1995.

[122] Mark D. Uncles, Grahame R. Dowling, and Kathy Hammond. Customer loyalty and customer loyalty programs. *Journal of Consumer Marketing*, 20(4):294–316, 2003.

[123] Peter C. Verhoef. Understanding the Effect of Customer Relationship Management Efforts on Customer Retention and Customer Share Development. *Journal of Marketing*, 64(4):30–45, 2003.

[124] Ivan Fernando Videla Cavieres. Characterization And Completation Of The Customer Data From A Retail Company Using Graph Mining Techniques. Master's thesis, Departamento Ciencias de la Computación, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, Santiago, Chile, 2014.

[125] KW Wang, S Zhou, Q Yang, and JMS Yeung. Mining Customer Value: From Association Rules to Direct Marketing. *Data Mining and Knowledge Discovery*, pages 57–79, 2005.

[126] Stanley Wasserman. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[127] Hua-Liang Wei and Stephen a Billings. Feature subset selection and ranking for data dimensionality reduction. *IEEE transactions on pattern analysis and machine intelligence*, 29(1):162–6, January 2007.

[128] RS Winer. Customer Relationship Management: A Framework, Research Directions, and the Future. *Haas School of Business*, April, 2001.

[129] Rüdiger Wirth and J Hipp. CRISP-DM: Towards a standard process model for data mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, April(24959):29–39, 2000.

[130] Eugene Wong. A statistical approach to incomplete information in database systems. *ACM Transactions on Database Systems*, 7(3):470–488, September 1982.

[131] Xiaopeng Xi, Eamonn Keogh, Christian Shelton, Li Wei, and Chotirat Ann Ratanamahatana. Fast time series classification using numerosity reduction. *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pages 1033–1040, 2006.

[132] Jierui Xie, S Kelley, and BK Szymanski. Overlapping Community Detection in Networks : the State of the Art and Comparative Study. *ACM Computing Surveys*, 45(4):1–37, 2013.

[133] Jierui Xie, BK Szymanski, and Xiaoming Liu. SLPA: Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process. *ICDMW 2011, 11th IEEE International Conference on Data Mining*, 2011.

[134] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. *ICML*, 2003.