Market Basket Analysis

Master in Data Science

Alberto Bertoncini 983833

Massimo Cavagna 9838??

Github repository

June 1, 2021

Contents

1 Introduction

The purpose of this paper is to present some of the techniques used in order to perform the so called "market/basket" analysis for a huge amount of data. At first these techniques were exploited for the analysis of purchases in markets, trying to find some relationships among the goods bought by customers. The idea behind these algorithms is to find associations between goods so that can be claimed that if a customer buy item A he is also likely to buy item B and vice versa. This concept could be extended to association between sets of goods (not only single item pairs) and to generic items instead of just goods, so that, in the end, the aim of the algorithms that will be presented is to find frequent sets of items in all the baskets available. In particular, three main algorithms will be implemented:

- A-priori:
    - base
    - PCY
- SON
- Toivonen

Once the frequents sets are found, it is also important to check if all the items in one of these sets are actually associated one another: indeed, considering the environment these techniques come from, we could find that some goods, such as "milk" or "bread" are always frequent, but it cannot be claimed that there is a actual relationship with all the other items in the same frequent set, since it will be bought independently from the other.

2 Dataset

The dataset in analysis is called "IMDb dataset" (version 6) created by data collected from the homonymous site IMDb. This dataset contains information about movies, their ratings and workers that have taken part in them. The data is divided into several files to simplify the analysis over specific aspect.

- *title.akas.tsv* contains informations about the localized version of the movies.
- *title.basics.tsv* contains general information about the movies, not influenced by the localization.
- *title.principals.tsv* contains information about the cast and the crew for each movie.
- *title.ratings.tsv* contains the IMDb rating informations about the movies.
- *name.basics.tsv* contains informations about the cast and the crew.

The aim of the analysis presented is to find sets of actors that have frequently worked together so only a few of these datasets will be used: in particular *title.principal.tsv* from which is possible create baskets of actors for every movie and *name.basics.tsv* from which is possible retrieve the names of the actors from their IDs.

3 Data structure

title.principal.tsv is a tsv file with the subsequent structure:

- *tconst* (string) is an alphanumeric identifier of the movie.
- *ordering* (integer) is a number used to uniquely identify rows for a given movie.

- *nconst* (string) is an alphanumeric identifier of the cast/crew person.
- *category* (string) is the role of that person in the movie (a person can do different roles in the same movie).
- *job* (string) is a further specification of the role (can be empty, with the symbol "\N").
- *characters* (string) is the name of the character played (can be empty, with the symbol"\N").

There are 36.499.704 rows for 5.710.740 different movies.
Only rows with an actor are considered, so the analysis is done over 14.830.233 rows.
Movies without any registered actor need to be filtered, reducing the number of movies to 3.602.200.
In the end the number of different actors in this dataset is 1.868.056.
The size of *title.principal.tsv* is 1.6 GB.

name.basics.tsv isa tsv file with the subsequent structure:
- *nconst* (string) is an alphanumeric identifier of the cast/crew person.
- *primaryName* (string) the name of the person.
- *birthYear* (YYYY) year of birth.
- *deathYear* (YYYY) year of death.
- *primaryProfession* (array of strings) the top 3 profession of the person.
- *knownForTitles* (array of tconst) movies the person is known for (can be empty).

There are 9.711.022 rows in this file with only 3.625.895 people marked as actor/actress.

4 Preprocessing

These datasets need to be formatted into precise structure in order to apply the market/basket analysis algorithms. For each movie is created the basket (a list) containing all the IDs for the actors that have played a role in it.
The actors are identified, into the baskets, with an integer ID, obtained from their alphanumeric ID (nconsts).

5 Algorithms applied

The algorithms implemented aim to obtain the list of frequent itemsets (sets of actors) in our preprocessed baskets exploiting the property of monotonicity in different ways, from trivial form to more complex but more efficiently. Since these algorithm are created to work well over big amount of data is important to underline the number of times an algorithm pass over the data, and the number of candidates itemsets created to be checked frequent or not. It is useful to represent each item with an integer value to simplify the handling, reducing the space and to be able to hash them.

5.1 A-priori

The most trivial approach is the A-priori algorithm, which scans multiple times the baskets in order to construct and filter itemsets of increasing size until there aren't more. In this way are done n passages over the data, one for every size of itemset reached. The number of candidates

frequent itemsets can become very high, because for every frequent itemset of size k need to create every possible combination of size k+1 from them, count the frequency for each candidates and filter only frequent ones, using a lot of space for candidates that will be discarded if not frequent.

5.2 PCY (single hash

An improvement of this algorithm is the PCY (Park, Chen, Yu) that exploit the free space present in memory while counting the frequency of itemsets of size k, creating an hashmap in which counts, in the position indicated by the hash of the itemset of size k+1, the number of times the latter occurs in the baskets (is possible have collisions, but this approach aims to reduce the number of itemsets to be saved in memory and for which need to count the frequency, not to directly delete every not frequent itemsets). The hashmap is compressed into a bitmap to save space and permitting to create only itemsets marked as possible frequent in this map. To reach frequent itemsets of size k need to pass k times over data, as the previous algorithm, but is possible reduce the candidates at every iteration.

5.3 SON

This algorithm is developed to parallelize the computation of frequent itemsets, working over chunks (partition) of the data. It's not a proper sampling algorithm because it will work on every chunk and, in the end, the totality of data will be elaborated.

It'a a way to use the previous implementations (A-priori/PCY) in a distributed form, to reduce the single workload over a chunk and parallelize the computation, and then put together the results. Is important enforce appropriate adjustment to run the A-priori over every chunk, like the repositioning of the threshold that divide frequent itemsets fron not frequent.

5.4 Toivonen

This algorithm, as the previously, exploits the A-priori or PCY implementation to obtain the frequent itemsets of a sample of the complete data (providing the adjusted threshold). This algorithm exploits the notion of **Negative Border** that once created will be used to check if the frequent itemsets in the sample are frequent even globally. The negative border is built extracting every itemsets that is not frequent but that have frequent immediate subsets. The algorithm, operating just over the sample can say if the candidate frequent itemsets are the right ones even for the totality of data, with a single passage over the complete dataset.

5.5 Implementation

These algorithms have been implemented with python using a sequential way for the PCY and the MapReduce paradigm to apply the PCY according the SON and Toivonen philosophy.

6 Scaling of proposed solutions

7 Experiments

8 Results

I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work. I/We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.