BIP Project Report

Politecnico di Milano

Andrea Bellotti
Matr. 835988
andrea1.bellotti@mail.polimi.it

Tommaso Carpi
Matr. 835988
tommaso.carpi@mail.polimi.it

Marco Edemanti
Matr. 835988
marco.edemanti@mail.polimi.it

Lorenzo Bisi
Matr. 835988
lorenzo.bisi@mail.polimi.it

Riccardo Mastellone
Matr. 852341
riccardo.mastellone@mail.polimi.it

ABSTRACT

This document aims to describe the algorithms and the techniques used in the **BIP project**. BLABLABLABLA without having any knowledge on the domain, it has been very difficult to make any kind of assumption on the results and especially during the data exploration phase trying to identify the best approach.

1. INTRODUCTION

The given dataset was initially a single csv with 6 columns and 26klines. Subsequently a second csv with...

2. DATA EXPLORATION

2.1 Dataset analysis

We started our work by analyzing the dataset provided, to understand if there were any evident patterns that we should take into account. As expected, a higher there is a higher concentration of *sottoaree* in areas corresponding to Italy major cities, meaning that the coordinates have not been skewed by much and can be considered an useful source of information.

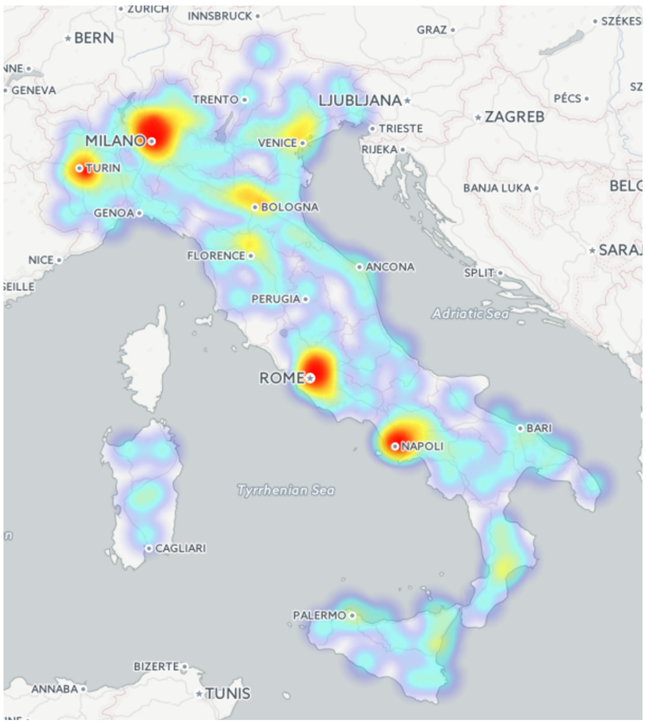

Figure 1: Distribution of *sottoaree*

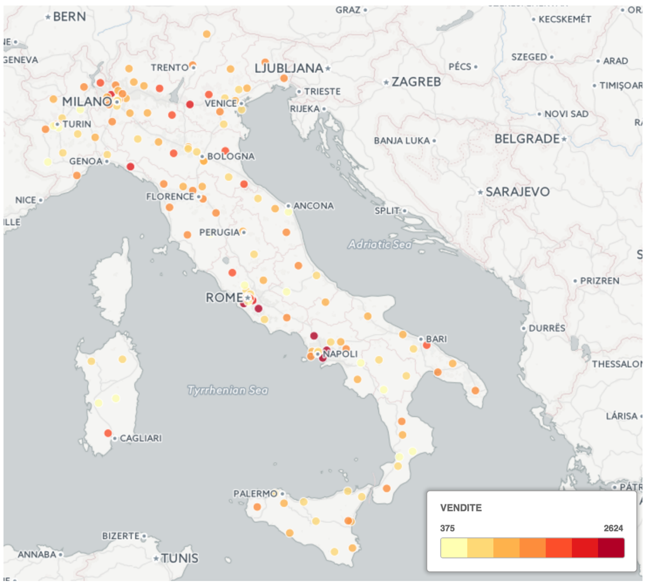

Figure 2: Distribution of *vendite* in 2016

We plotted the data from various perspectives, to understand if there have been any non negligible variation during the 3 different years provided but apart from a general increase in numbers of *sales* nothing else was detected.

2.2 Feature extraction

Next we passed to the features extraction phase. Starting from the first csv, we expanded the *Data* (see Table 1).

The last one, was computed based on the Italian calendar and has been a very important feature, having a high correlation with the *sales*.

Day of the week
Day of the month
Day of the year
Month
Year
Holiday

Table 1: Data new features

Region	The name of the region
Area	North, Centre, South
Island	Whether is an island or not
Population	Inhabitants of the city
MinDistance	Minimum distance to another *sottoarea*
Density	Number of *sottoarea* within 10km
InstructionIndex	Upper secondary education attainment rate
AverageIncome	The average income
IncomeGiniIndex	The Gini index of the distribution of the incomes
WorkForce	Percentage of population that can work
Occupation	Percentage of the work force that has a job

Table 2: Location new features

From the second csv, we were able to obtain a higher number of interesting features, as we integrated and computer additional indexes from datasets extracted from the **ISTAT** data warehouse[1] (see Table 2).

Since the first submissions, the *Item Content Matrix* was first processed with the *Inverse Document Frequency* function (not *Term Frequency*, as it would not make sense in this case). Changing the *Inverse Document Frequency* normalization factor (e.g., from *log* to *log*2) would not result in substantial variations in the score.

To measure the similarity between items and compute the estimated ratings, we used the *Cosine Similarity* function: our goal was then to identify the shrink term K that would maximise the score. Using a trial and error approach, the highest score, **0.00544** was obtained with:

$$K = 44 \tag{1}$$

2.2.1 Collaborative Filtering Algorithms

Our third approach was to implement *Collaborative Filtering* algorithms: user-based and item-based. As item-based collaborative filtering algorithms generally provide good recommendations, especially compared to the user-based ones, it was the first one to be implemented. As a similarity function, both *Cosine Similarity* and *Adjusted Cosine* were used (*Pearson Correlation* removes the items bias which we considered not to be too relevant). Since the first implementations, the *k-Nearest Neighbors* algorithm was used, where:

$$k = 8 \tag{2}$$

Additionally, *Global Effects* were computed on the dataset before the collaborative filtering algorithms effectly produced the recommendations.

The *Item-Based Collaborative Filtering* submissions resulted in very poor score results, pushing us to use the *User-Based Collaborative Filtering* instead. Surprisingly, still using the *Consine Similarity* as user bias is removed with the *Global Effects*, we obtained a better result, but still lower that the *Collaborative Filtering.* These results suggested us the use of *Hybrid* algorithms (see 2.2.2)

2.2.2 Matrix Factorization and Hybrid Algorithms

Our last group of algorithms involved *SVD* and *Hybrid* algorithms. Expecting notable improvements in the score implementing *SVD*, a relevant amount of time has been spent implementing it with different approaches.

Matrix Factorization: the goal was to approximate the URM as the product of two matrices by observing only the most important features.

- Funk's SVD: we started computing the baseline by unbiasing the URM and then iteratively minimize the prediction error of each rating, using the gradient descend, in order to learn the latent factors. Parameter used:

$$\lambda = 0.01 \;\; \gamma = 0.02 \;\; nFactors = 100 \tag{3}$$

- SVD++: MATLAB natively provides a $svds(URM, k)$ function in order to get the V' matrix

Hybrid Algorithms: we tried then to mix some of our algorithms in order to obtain better recommendations by pipelining *Global Effects* and merging in the last phase:

$$\widetilde{r}_{u,i} = \alpha \cdot \widetilde{r}_{A(u,i)} + \beta \cdot \widetilde{r}_{B(u,i)} \tag{4}$$

Keeping the *Content Based* as the A algorithm, we varied B with *User* and *Item-Based Collaborative Filtering.* The idea is to cope with the low number of ratings problem by computing *Content Based* recommendations as a step to add some poor users profiles with meaningful ratings (i.e rating predictions for items that nobody rated before). Unfortunately the noise generated with such technique resulted in a lower score: **0.00404**.

Additionally, as *SVD* resulted in very low scores, we decided to improve our existing algorithms by using *SVD* and *Latent Semantic Analysis* to reduce possible noise of our dataset, with little to no success.

2.3 Evaluation

We performed an off-line evaluation by partitioning our dataset with the *Hold-Out* technique:

- We took a subset of users with more than 5 positive ratings (where a positive rating is a rating greater than or equal to 8).

[1]http://dati.istat.it/

- The 80% of the users were considered as train users while the other 20% were test users.
- For each test user, 5 most unpopular items with positive rating were added to the test set, while the rest of the ratings were added to the train set.

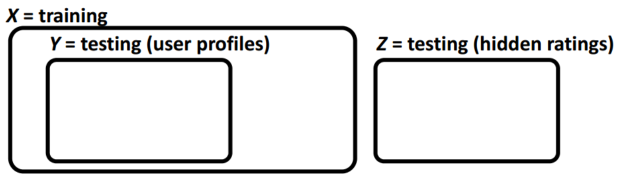

Figure 3: Hold-Out

We computed the *RMSE(Root Mean Square Error)* in order to measure the accuracy of the predicted ratings.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{|T|} \sum_{(u,i) \epsilon T} (r_{u,i} - \widetilde{r}_{u,i})^2} \qquad (5)$$

where T is the test set.

In order to keep the overfitting under control we made use of shrink terms and tried to use values that would reduce the RMSE and increasing our score on the public leader board as well.

3. CONCLUSIONS

At the very end we agreed that the content-based recommenders were the most effective due to the fact that they do not suffer from the first-rater problem (which affects collaborative recommenders); so we were able to provide each user a relatively good rating prediction to almost every movie in the dataset.