A Semi-Supervised Hybrid System to Enhance the Recommendation of Channels in Terms of Campaign ROI

Julie Séguéla
Cedric-CNAM & Multiposting.fr
292, rue Saint-Martin
75141, Paris, France
jseguela@multiposting.fr

Gilbert Saporta
Cedric-CNAM
292, rue Saint-Martin
75141, Paris, France
gilbert.saporta@cnam.fr

ABSTRACT

In domains such as Marketing, Advertising or even Human Resources (sourcing), decision-makers have to choose the most suitable channels according to their objectives when starting a campaign. In this paper, three recommender systems providing channel ("user") ranking for a given campaign ("item") are introduced. This work refers exclusively to the new item problem, which is still a challenging topic in the literature. The first two systems are standard content-based recommendation approaches, with different rating estimation techniques (model-based vs heuristic-based). To overcome the lacks of previous approaches, we introduce a new hybrid system using a supervised similarity based on PLS components. Algorithms are compared in a case study: purpose is to predict the ranking of job boards (job search web sites) in terms of ROI (return on investment) per job posting. In this application, the semi-supervised hybrid system outperforms standard approaches.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Indexing methods; I.2.6 [**Learning**]: Knowledge acquisition

General Terms

Algorithms, Experimentation

Keywords

Recommender systems, feature extraction, PLS, channels

1. INTRODUCTION

With the expansion of internet to advertise, the number of potential channels (and targets) is exponentially growing. Today, a great challenge common to several research domains is the development of intelligent tools to support (or to replace) users in their choices. In this context, we are introducing algorithms which aim at providing a ranking of channels according to the expected ROI, in order to help decision-makers to identify the best channels. To do that, we resort to an innovative application of recommender system, in which items are assumed to be campaigns, users are assumed to be channels, and ratings are assumed to be observed returns of campaigns on channels. The type of recommender studied is very specific: since each campaign is different, we don't have any explicit preference on items of interest. Consequently, traditional collaborative methods are excluded: we have to face to item cold-start problem, which is still a challenging topic in the literature. In order to simplify and to compare the effectiveness of algorithms on high dimensional data, items are only described by textual data. Background is presented in section 2. Algorithms and performance indicators are described in section 3. Section 4 is dedicated to a case study: identify the most effective sourcing channels to support recruiters in their choice.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.
Copyright 2011 ACM 978-1-4503-0717-8/11/10 ...$10.00.

2. BACKGROUND

2.1 Feature extraction

As previously mentioned, campaigns are characterized by a text. To extract a set of features from the text we will use several common techniques in Information Retrieval and compare them. First, we use the well-known Vector Space Model in order to obtain a vector of term frequencies for each campaign. To improve text representation we will use TF-IDF weighting, which decreases weights of keywords appearing in many documents and not useful to distinguish a relevant document from an irrelevant one. After weighting, documents will be indexed by Latent Semantic Analysis, a dimensionality reduction technique through a singular value decomposition of term-document matrices [4].

2.2 Recommender systems

Recommender systems are usually classified into three main categories: content-based recommendations, collaborative recommendations, and hybrid approaches [1]. The first approach proposed is a content-based algorithm which recommendation technique is based on a model (PLS regression, see 2.3). The second approach uses a heuristic-based recommendation technique (computing of a feature-based similarity measure) in a content-based approach. In addition to these two basic approaches, we introduce a hybrid recommender (output produced by a content-based model is used as an input in a collaborative approach) based on a heuristic technique to estimate ratings. Hybridization strategy used is the same as in [5] or [2], also called "collaboration through content".

2.3 PLS Regression

Partial Least Squares Regression (PLS-R), introduced by [9], is a technique that generalizes and combines features from principal component analysis and multiple regression. This method is used when the number of predictors is large, and also large compared to the number of observations. The higher the number of predictors is, the higher the risk of important correlations between variables is. In case of high correlations between predictors, standard regression methods fail in estimating coefficients. As an alternative, PLS-R provides robust components (linear combinations of predictors), independent and highly correlated with the dependent variable. In this work, we have to cope with a large number of predictors (number of features extracted from campaign descriptions) and high correlations, so PLS-R seems to be a very suitable technique.

3. RECOMMENDATION APPROACHES

3.1 Extraction of PLS components

One part of PLS-R algorithm consists in computing non-correlated components, highly correlated with the dependent variable. The computing of components is based on NIPALS algorithm, first introduced by [8] for principal component analysis. Considering $(x_1, ..., x_p)$ a set of features, potentially highly correlated, and a dependent variable r, the first component is: $t_1 = w_{11}x_1 + \cdots + w_{1p}x_p$, where $w_{1j} = \frac{cov(x_j,r)}{\sqrt{\sum_{j=1}^{p} cov^2(x_j,r)}}$. Residual r_1 is obtained by regressing r on t_1. The second component t_2 is a linear combination of x_{1j}, residuals coming from the regressions of x_j on t_1: $t_2 = w_{21}x_{11} + \cdots + w_{2p}x_{1p}$, where $w_{2j} = \frac{cov(x_{1j},r_1)}{\sqrt{\sum_{j=1}^{p} cov^2(x_{1j},r_1)}}$. The regression of r on t_1 and t_2 gives the residual r_2. This iterative process continues until reaching H components, where H is determined by cross validation [3].

3.2 Recommender algorithms

Let $r_{u,i}$ and $p_{u,i}$ be respectively actual and predicted rating of user u for item i.

3.2.1 Model-based recommender: PLS-R

The first approach is content-based, and ratings are estimated by a PLS regression model learned on each user. After computing of PLS components (see 3.1), a linear regression is applied: $r = c_1t_1 + \cdots + c_Ht_H + r_{H+1}$, where c_h are the regression coefficients. PLS components and predictions $p_{u,i}$ are computed by *10-fold* cross-validation.

3.2.2 Heuristic feature-based recommender

The second approach is also content-based but ratings are predicted by an heuristic technique. The similarity of the new item with respect to all past items is computed. It is a feature-based similarity since no rating is known for this new item. This approach assumes that similar items regarding to their features should have a similar rating for a given user. Four similarity measures between items i_1 and i_2 are tested. Since we are working on textual data, cosine similarity measure will be used. As a particular case, we will test constant similarity measure. The following similarity functions are based on Euclidean distance: d_{i_1,i_2}(the lower the Euclidean distance, the greater the similarity). We use the same method as [6] for generating a third similarity measure: $sim(i_1, i_2) = \max_{i_k \in K}(d_{i_1,i_k}) - d_{i_1,i_2}$, where K is the set of $|K|$ nearest neighbors of i_1 according to d_{i_1,i_2}. Finally, the fourth similarity function is inverse proportional to Euclidean distance: $sim(i_1, i_2) = 1/d_{i_1,i_2}$.

Ratings are then estimated thanks to an aggregating function [1], computed on item neighborhood. Expected rating of user u for item i_1 is given by:

$$p_{u,i_1} = \frac{\sum_{i_{k'} \in K} sim(i_1, i_{k'}) \times r_{u,i_{k'}}}{\sum_{i_{k'} \in K} sim(i_1, i_{k'})} \quad (1)$$

where K is the set of $|K|$ nearest neighbors of i_1 with respect to the correspondent similarity measure. For constant similarity, which means computing average rating on i_1 neighborhood, nearest neighbors are determined by Euclidean distance.

3.2.3 Semi-supervised hybrid recommender

While recommender introduced in section 3.2.2 is based on a "naive" similarity, this approach is based on a supervised similarity measure which allows to identify nearest neighbors with respect to relevant features. Instead of computing Euclidean distance on item features, it is computed on PLS components which are extracted so as to explain as much as possible actual ratings. Similarity measures are computed as defined in section 3.2.2. Cosine similarity is not used because in practice, the number of components kept is very low (1 to 12). Similarities are then taken as inputs in a collaborative approach by respecting (1) formula.

These three approaches will be respectively named $S1$, $S2$, and $S3$ in the rest of the paper. The third algorithm is built so as to overcome weaknesses of the first two approaches. PLS-R implies a linear constraint between components and the dependent variable, whereas other approaches are non-linear. In addition, the risk of overfitting is high (because of the high number of input variables) while it is low or absent in other approaches. We appreciate the ability of $S1$ and $S3$ to provide interpretation tools of variable impact (at the PLS modeling step), absent in $S2$.

3.3 System evaluation

Recommender systems are systematically compared with the performance of the "average" recommender (AR). The latter provides channel recommendations based on their average rating (computed on all past campaigns) and does not take campaign features into account.

3.3.1 Mean Absolute Error

Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE) are commonly used to assess the capability of a system to provide good rating estimations. In our context, users have big differences in the number of rated items. This can bias our estimator because of the high variability between average ratings of users. Consequently, the mean of users MAE is used to compare system performances on the prediction task: $\overline{MAE} = \frac{1}{|U|}\sum_{u \in U} \frac{\sum_{i \in D_u} |p_{u,i} - r_{u,i}|}{|D_u|}$, where U is the set of users and D_u the set of items previously rated by user u.

3.3.2 Ranking error and return loss

Although providing an accurate prediction is important, professionals want to choose channels with the best ROI

under their budget constraint. The estimated return on investment of campaign i on channel u is: $ROI_{u,i} = \frac{p_{u,i}}{c_u}$, given that the cost of a channel c_u is independent from the campaign. Since the cost is fixed for a given channel, ROI is deducted in an obvious way from the estimated return $p_{u,i}$. In order to simplify and without loss of generality towards performance evaluation, we consider a fixed cost c for all channels. Consequently, classifying estimated returns in decreasing order is sufficient to give us the ranking of channels. In order to assess system performances on a domain-driven way, we are interested in return loss generated by an error in channel ranking. In this case study, we focus on the first rank channel because it is by definition that with the most important return. Two measures of performance are considered: the ability of system to find the first rank channel, and the reduction of return loss when using an alternative recommender S instead of using average recommender AR. Let a be the number of campaigns for which first rank channel is correctly identified, and $a+b$ the number of campaigns, the first evaluation measure for recommender S is: $Precisions_S = \frac{a}{a+b}$. Let r_{R1} be the return associated with actual first rank channel and $r_{P1,S}$ the return associated with first rank channel predicted by system S. Per se, $r_{R1} \geq r_{P1,S}$, equality being true when the first channel is correctly predicted. The second indicator is defined by: $Loss\ reduction_S = \frac{\sum(r_{R1}-r_{P1,AR})-\sum(r_{R1}-r_{P1,S})}{\sum(r_{R1}-r_{P1,AR})}$.

4. EXPERIMENTS: JOB BOARD RECOMMENDATION FOR JOB POSTINGS

The increasing number of online job boards has made crucial the introduction of decision-making tools adapted to recruiter needs. On a job board, a job posting ROI can be measured by the number of applications received per currency unit spent (or symmetrically the cost per application received). We consider that each job board has the same cost and focus on the number of applications received on channels. The purpose of application is to provide a tool which automatically recommends the best job board with respect to the expected return. To our knowledge, this kind of data has never been studied in the recommender system literature. Data are provided by *Multiposting.fr*, an online job posting distributor, and concern generalist and specialized job boards.

4.1 Dataset description

This kind of dataset is quite unusual in recommender system literature: only few users (31 job boards), very small dataset (about 30 500 ratings), and high rating variability inside and between users. Return on a job board is not limited contrary to usual recommenders where ratings take values between 0 and 5, which explains that our collaborative recommendation approaches are item-based. The number of items is reasonable with about 7 730 jobs posted on 4 different job boards in average. We are studying job boards with 100 postings and more.

4.2 Feature extraction

The first step is the preprocessing of job offer texts to obtain "bag-of-words" representation: lemmatization and tagging, filtering according to grammatical category, filtering words occurring in less than five postings. Before preprocessing, a posting is in average 165 tokens long, and the total size of vocabulary is about 20 000 distinct words. After preprocessing, a posting is in average 92 tokens long, and the vocabulary size is reduced to 5 000 distinct words. During experiments, several text representation techniques are tested: term frequency (TF), TF-IDF weighting, LSI (with TF-IDF weighting). Since $S2$ is a non supervised system, the number of dimensions kept with LSI can impact the recommender performance: experiments lead us to keep 50 dimensions (highest *loss reduction*).

Table 1: PLS-R (S1) results

Feature representation	$\overline{MAE}$	Precision (%)	Loss reduction (%)
AR	10.18	62.8	.
TF	7.80	63.3	27.7
TF-IDF	7.64	62.7	23.3
LSI	7.98	61.8	27.8

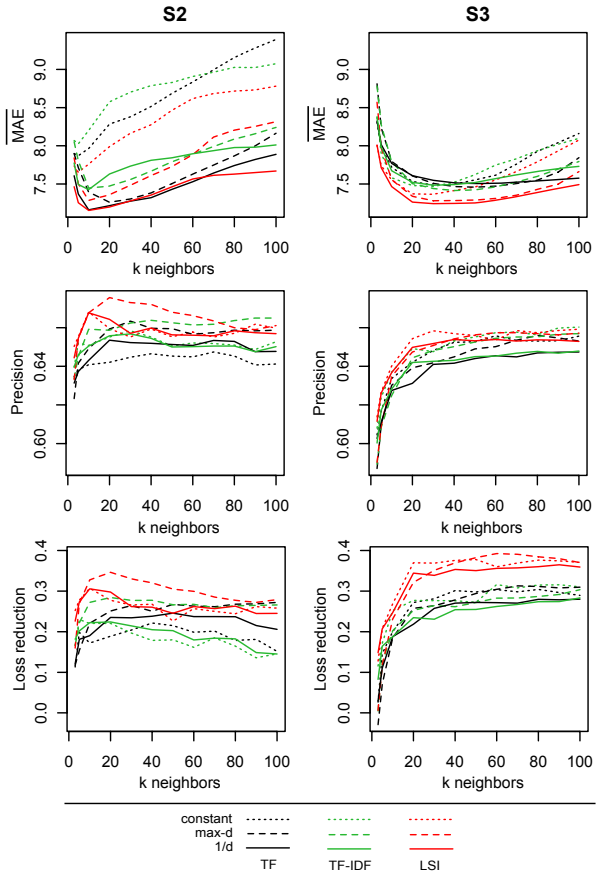

Figure 1: S2 and S3 performance according to feature representation method and similarity measure.

4.3 Results

Loss reduction will be used to identify the best recommenders while other indicators, $\overline{MAE}$ and *Precision*, have an illustrative purpose. Table 1 shows $S1$ results according to model input data, and average recommender (AR) results. $S1$ globally outperforms average recommender and differences associated with different feature representation techniques are very slight. Results of experiments for $S2$ and $S3$ systems are presented in Figure 1. Results obtained

Table 2: Best recommender for each approach

Approach	$\overline{MAE}$	Precision (%)	Loss reduction (%)
AR	10.18	62.8	.
S1	7.98	61.8	27.8
S2($k = 20$)	7.36	**67.6**	34.6
S3($k = 60$)	**7.31**	65.7	**39.2**

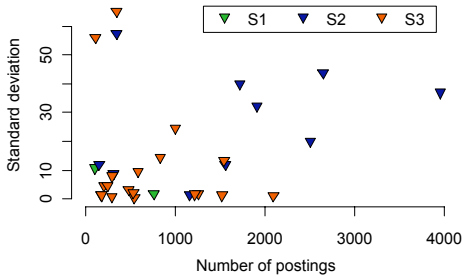

Figure 2: The best system for each job board according to the number of postings and return variability.

with cosine similarity are not represented because they are very close to those obtained with average on neighborhood (constant similarity). Whatever the similarity measure, for both $S2$ and $S3$ approaches, LSI representation allows to obtain the best performance indicators (except for $\overline{MAE}$ in $S2$ approach where TF representation performs equally). Assuming that LSI provides the most relevant features to explain the job board return, we now focus on loss reduction criterion and comment the impact of similarity measure. While $max - d$ is better than other similarities in approach $S2$, the three measures are fairly equivalent in approach $S3$. The highest loss reduction is reached with $max - d$ similarity in both approaches, so we choose it for a general comparison of recommender performances (Table 2). The three approaches outperform average recommender. $S2$ allows to reach a higher loss reduction than $S1$, but $S3$ is the best recommender with regard to this performance indicator.

In the non supervised approach ($S2$), the optimal number of neighbors is very low: 20 or less. Especially with constant similarity (average prediction), $\overline{MAE}$ increases drastically with the number of neighbors. That measure gives too much importance to distant neighbors (contrary to $1/d$ weighting). In the semi-supervised approach ($S3$), indicators are more stable from 20 to 100 neighbors. PLS components operate as an information smoothing method.

4.4 Discussion

In Figure 2, each point represents a job board and color designates the winner recommender (that which has provided the lowest MAE on the channel). $S1$ is the best recommender on only 2 job boards, $S2$ on 10 job boards and $S3$ is winning on 19 job boards. While $S3$ seems to be efficient on job boards with few postings, on the contrary, $S2$ seems to perform better on job boards with a high number of postings (the highest the number of postings, the highest the probability to find a very similar offer on the job board). Since we have kept a selection of features (and not all features) as input in PLS modeling, some information has been lost in the process. But when job boards have few postings, generalization power of PLS provides better results.

5. CONCLUSION AND FUTURE WORK

We have introduced a semi-supervised hybrid recommender suitable in the case of new item problem. This system outperforms content-based and non-supervised approaches, by combining the understanding of campaign features impact on channel ROI and the usage of collaborative knowledge. The evaluation of system performances on the basis of channel returns ensures relevant results in practice for similar applications.

Campaigns are described by texts but we could add other information to the vector of descriptors. In this latter case, $S3$ would be a suitable recommendation approach (fitting of predictor weights according to their power of explanation) contrary to $S2$. Weights of variables could be even more controlled with techniques such as multiblock PLS [7]. In the future, we will test other similarity functions (e.g. gaussian) under a nonparametric theoretical framework.

6. ACKNOWLEDGMENTS

We would like to thank Stéphane Le Viet and Gautier Machelon (*Multiposting.fr* company) for supporting this work. We also thank Guillaume Bandet (*Multiposting.fr* company) for his many helpful comments on this paper.

7. REFERENCES

[1] 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, 2005.

[2] M. Balabanovic. An adaptive web page recommendation service. In *Agents 97: Proceedings of the 1st International Conference on Autonomous Agents*, pages 378–385, 1997.

[3] A. Höskuldsson. PLS regression methods. *Journal of Chemometrics*, 2:211–228, 1988.

[4] D. I. Martin and M. W. Berry. Mathematical foundations behind latent semantic analysis. In T. K. Landauer, D. S. McNamara, S. Dennis, and W. Kintsch editors, *Handbook of Latent Semantic Analysis*, pages 35–55. Lawrence Erlbaum Associates, 2007.

[5] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5/6):393–408, 1999.

[6] U. Shardanand. *Social Information Filtering for Music Recommendation*. MIT EECS M. Eng. thesis, 1994.

[7] L. E. Wangen and B. R. Kowalski. A multiblock partial least squares algorithm for investigating complex chemical systems. *Journal of Chemometrics*, 3(1):3–20, 1989.

[8] H. Wold. Estimation of principal components and related models by iterative least squares. In P. R. Krishnaiaah editor, *Multivariate Analysis*, pages 391–420. New York: Academic Press, 1966.

[9] S. Wold, H. Martens, and H. Wold. The multivariate calibration problem in chemistry solved by the pls method. In *Proceedings of the Conference on Matrix Pencils*, pages 286–293, 1983.