Hybrid Recommender System for Prediction of the Yelp Users Preferences

Vladimir Nikulin

Department of Mathematical Methods in Economy,
Vyatka State University, Kirov, Russia
vnikulin.uq@gmail.com

Abstract. Recommender systems typically produce a list of recommendations in one of two ways - through collaborative or content-based filtering. Collaborative filtering approaches build a model from a user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users; then use that model to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties. These approaches are often combined, and called Hybrid Recommender Systems. In this paper we present hybrid recommender system, which was used online during ACM RecSys 2013 Contest, where we were awarded 2nd best prize. The contest was based on the real data, which were provided by Yelp - US internet based business recommender.

Keywords: collaborative filtering, content-based filtering, hybrid recommender systems.

1 Introduction

Recommender systems (RSs) attempt to profile user preferences over items, and model the relation between users and items. The task of recommender systems is to recommend items that fit a users tastes, in order to help the user in selecting/purchasing items from an overwhelming set of choices [1]. Such systems have great importance in applications such as e-commerce, subscription based services, information filtering, etc. Recommender systems providing personalized suggestions greatly increase the likelihood of a customer making a purchase compared to unpersonalized ones. Personalized recommendations are especially important in markets, where the variety of choices is large, the taste of the customer is important, and last but not least the price of the items is modest. Typical areas of such services are mostly related to art (esp. books, lectures, movies, music), fashion, food and restaurants, gaming and humor.

Recommender systems play an important role in such highly rated Internet sites as Amazon.com, YouTube, Netflix, Yahoo, Tripadvisor, Last.fm, and IMDb. Moreover many media companies are now developing and deploying RSs as part

P. Perner (Ed.): ICDM 2014, LNAI 8557, pp. 85–99, 2014.

of the services they provide to their subscribers. There are dedicated conferences and workshops related to the field. We refer specifically to ACM Recommender Systems (RecSys), established in 2007 and now the premier annual event in recommender technology research and applications [2]. At institutions of higher education around the world, undergraduate and graduate courses are now dedicated entirely to recommender systems; tutorials on RSs are very popular at computer science conferences; and recently a book introducing RSs techniques was published [3]. There have been several special issues in academic journals covering research and developments in the RS field. Among the journals that have dedicated issues to RS are: AI Communications (2008); IEEE Intelligent Systems (2007); International Journal of Electronic Commerce (2006); International Journal of Computer Science and Applications (2006); ACM Transactions on Computer-Human Interaction (2005); and ACM Transactions on Information Systems (2004).

Content-Based systems focus on properties of items. Similarity of items is determined by measuring the similarity in their properties.

Collaborative-Filtering systems focus on the relationship between users and items. Similarity of items is determined by the similarity of the ratings of those items by the users who have rated both items.

Suppose that user is new to the market ("cold start"), and we have no any corresponding review records. In this case, and in accordance with the content-based approach, we can use available properties for user and business in order to predict how user will rate the business under consideration.

In recommender systems, the cold start problem is often reduced by adopting a hybrid approach between content-based matching and collaborative filtering [4]. New items (which have not yet received any ratings from the community) would be assigned a rating automatically, based on the ratings assigned by the community to other similar items. Item similarity would be determined according to the items' content-based characteristics such as empirical probabilities or association rules. Note that traditional data mining techniques such as association rules were tried with good results at the early stages of the development of recommender systems [5].

Recent research has demonstrated that a hybrid approach, combining collaborative filtering and content-based filtering could be more effective in some cases. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model. Several studies empirically compare the performance of the hybrid with the pure collaborative and content-based methods demonstrate that the hybrid methods can provide more accurate recommendations than pure approaches. These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem.

It is not enough to create vectors describing items or businesses, we, also, have to create vectors with the same components that describe the users preferences.

We have the list of connections between users and items (it maybe list of review rates or list of purchases, for example). With this information, the best estimate, we can make regarding which items the user likes, is some aggregation of the profiles or properties of those items.

Note that the number of sufficiently frequent properties or categories maybe very large (in our case, see Section 2.2, it is 354). Accordingly, it will be too difficult to load and analyse the data in the standard form of the tables. One way to overcome this problem would be transfer all data (including locations of the businesses) into the sparse format. However, we decided to implement special novel transformation, which in described in Section 5. This transformation let us keep all the remaining blocks in a standard form. In order to calculate the final predictions, we used homogeneous ensembling [6], where any single GBM-based learner in R was based on about 10% of all data. In line with the main computations, we were able to validate our model with CV-passports, see Remark 8.

2 ACM RecSys 2013 Yelp Data

The Contest was started on 24th April 2013 and was ended on 31st August 2013 (129 total days). Initial period of the contest was ended on 24th August with 465 actively participated teams. The final solutions for the problem were submitted by 158 teams, and our solution was formally recognised as the 2nd best, see Table 4, where top ten teams and results are presented.

RecSys database[1] was given in JSON format, and includes two parts 1) training with 229907 known reviews (integers from 1 to 5 - assessments of the businesses by customers or users), where 1 stands for poor and 5 - for excellent; 2) testing with 36404 unknown reviews (to be predicted). The task was to minimise RMSE - the root mean squared error:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{s}_i - s_i)}{n}}, \tag{1}$$

where n is the total number of review ratings to be predicted, $\hat{s}_i$ is predicted value for review i, s_i is actual rating for review i.

Both datasets are divided into four groups (see Table 1): 1) business (or bus - to simplify notations), 2) checkins (statistical information about day of the week and time in hours when enquiry regarding related business was submitted), 3) users and 4) reviews.

All four groups are divided into subgroups: for training and testing. Note that train.bus $\cap$ test.bus $= \emptyset$; train.user $\cap$ test.user $= \emptyset$, where train.reviews are formulated in the terms of train.bus and train.users, test.reviews are formulated in the terms of all data: train.bus, test.bus, train.user and test.user. The coverage of the test.reviews by the train.bus and test.bus is full and complete, but in the significant proportion of all cases (2039 out of 36404) users are absolutely new.

[1] `http://www.kaggle.com`

Table 1. Some statistical characteristics of the RecSys 2013 Challenge data

name	train	test
business	11537	2797
checkin	8282	1796
user	43873	9522
review	229907	36404

That means, they can not be found in any of the available two sets: train.users or test.users. Such problem is known as "cold start"[7]-[8].

2.1 Additional Information

The data, which were described in the previous section, are sufficient in order to implement collaborative filtering. In this section we shall describe some additional information, which was provided by the organisers, and maybe used as a base for the content-based modelling.

The following information was given for train.bus: 1) city, 2) state, 3) latitude, 4) longitude, 5) $s(b)$ - average stars (star ratings or average reviews, rounded to half-stars), 6) $r(u)$ - review counts, 7) categories (see Table 2); and for train.users: 1) $r(u)$ - review counts, 2) $s(u)$ - average stars, 3) votes ('useful', 'funny', 'cool').

Remark 1. Note that stars field was removed from the test.bus; stars and votes fields were removed from test.users.

2.2 Business Categories

As an illustration, let us consider 13 popular businesses, as they maybe seen on the Yelp web-site: Arizona, Bank of America, Credit Union West, Kmart, Pizza-Hut, Jimmy John's, Starbucks, Panda Express, Homewood Suites (Hilton), Target, Safeway, McDonald's and Walmarts.

We counted $n_c = 354$ sufficiently frequent categories, which were used in train.business not less than 5 times. Other categories were ignored.

Remark 2. The categories data are sparse. This property is a very essential: not more than ten categories are used for the description of any business.

Further, we formed binary matrix of features $\mathcal{A}$ of $\{n_b \times n_c\}$, where $n_b = 11537$ - the size of train.bus set.

Stars (or average business ratings) were used as a target variables and categories were used as features (binary data, where one stands for present, and zero - for absent). Using "randomForest" function in R, we computed positive importance RF-ratings of different categories (bigger value means greater importance). The size of the training data in terms of reviews is a quite large. Accordingly, it will be very important to reduce dimensionality of the data, and we shall consider this topic in Section 5.

Table 2. Examples of categories for some businesses, where zero means empty space

Food	Ice Cream + Frozen Yogurt	0
Pet Services	Pet Boarding/Pet Sitting	Pets
Active Life	Yoga	Fitness + Instruction
Hobby Shops	Shopping	Toy Stores
Bars	Nightlife	0
Department Stores	Fashion	Shopping
Pizza	Restaurants	0
Mexican	Restaurants	0
Tanning	Beauty + Spas	0
Auto Repair	Automotive	0
Professional Services	0	0
Greek	Restaurants	0
Food	Donuts	Coffee + Tea
Hotels & Travel	Event Planning + Services	Hotels
Chinese	Restaurants	0
Local Services	Appliances + Repair	Home Services
Auto Repair	Automotive	0
Food	Coffee & Tea	0
Banks & Credit Unions	Financial Services	Mortgage Brokers
Department Stores	Fashion	Shopping
Hotels & Travel	Event Planning + Services	Hotels
American (New)	Restaurants	0
Breakfast & Brunch	Restaurants	0
Tapas/Small Plates	Restaurants	0
Health & Medical	Dentists	General Dentistry
Arts & Entertainment	American (Traditional)	Music Venues

3 Model N1: Content-Based Filtering with Average Stars

This model is the most straightforward, but complete component of the proposed recommender system.

Suppose, we would like to calculate prediction $\hat{s}(u, b)$ how user u will rate business b, where the pairs $\{s(u), r(u)\}$ and $\{s(b), r(b)\}$ are known (that means, both u and b are listed in the train.user and train.business). Then, we can implement the following formula

$$\hat{s}(u, b) = \exp\left(\frac{r(u) \cdot \log(s(u)) + r(b) \cdot \log(s(b))}{r(u) + r(b)}\right). \tag{2}$$

Otherwise, in the cases if $s(u)$ or $s(b)$ are known, we shall apply the following formulas

$$\hat{s}(u, b) = s(u); \tag{3a}$$

$$\hat{s}(u, b) = s(b). \tag{3b}$$

Table 3. List of the most important categories, where n is the number of occurrences in the train.bus set

Index	Category	n	Rating
73	Restaurants	4503	46.6809
211	Fast Food	386	34.6272
216	Apartments	83	30.4125
138	Real Estate	118	25.6183
28	Mobile Phones	70	24.2056
212	Hotels + Travel	379	17.0649
324	Specialty Food	181	15.8686
24	Beauty + Spas	764	14.3454
301	Hotels	284	13.0453
293	Active Life	525	11.0247
175	Food	1616	10.6413
93	Internet Service Providers	22	10.6279
81	Hair Salons	154	10.6242
160	Nail Salons	242	10.5391
247	Home Services	409	10.0099
232	American (Traditional)	480	9.7582
88	Shopping	1681	9.5534
268	Drugstores	125	9.1458
126	Event Planning + Services	453	9.0172
264	Chiropractors	39	8.5157
11	Car Wash	70	8.4186
29	Performing Arts	54	7.8904
244	Health + Medical	471	7.7370
343	Fitness + Instruction	204	7.3313
166	Professional Services	71	7.2573

3.1 Model N2 (Case of Star Averages, Extracted from the Review Data)

Using train.review data we can compute average stars for all involved businesses and users. In addition, we compute the numbers of reviews (support): $\phi(u)$ and $\phi(b)$. Further, we distinguish users and businesses according to the numbers of reviews.

Smoothing with Average Stars as Functions of the Numbers of Reviews. Clearly, average stars, which are based on small numbers of reviews are noisy and should be regularised. On the other hand, the numbers of users and businesses with small number of reviews are large (see Figure 1(a,c)), and the corresponding average stars z (see Figure 1(b,d)) maybe applied as regularises.

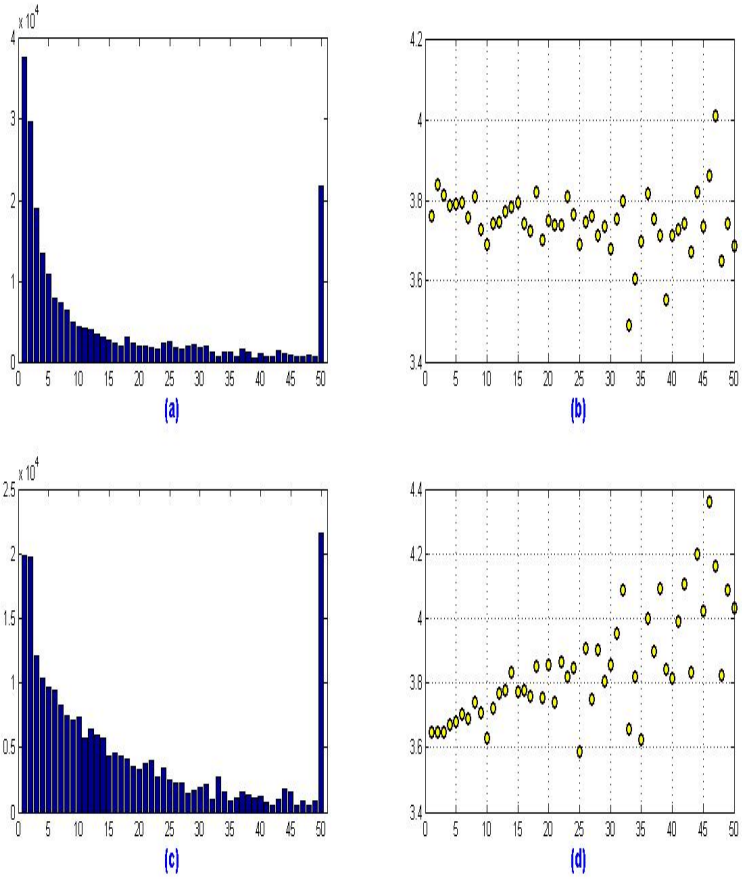

Fig. 1. (a) support for users, see Section 3.1; (b) average stars for users; (c) support for businesses; (d) average stars for businesses, where all horizontal axes represent number of occurrences in the training set

Remark 3. It is very interesting to note that users with bigger number of reviews appears to be more conservative (there is a slight decline in value of average stars $z(\phi(u))$, see Figure 1(b)). On the other hand, we can see that businesses which are more popular and attract greater attention (means with bigger number of reviews) are gaining higher average stars $z(\phi(b))$, see Figure 1(d) (smoothing parameters 3 and 6 were used for users and businesses to compute average stars).

The following smoothed averages were used as an input in (2)

$$\tilde{s}(u) = (1-\alpha)\hat{s}(u) + \alpha z(\phi(u)), \tag{4}$$

with smoothing parameter

$$\alpha = \frac{1}{(1+\phi(u))^{\gamma}},$$

where $\gamma = 0.25$ (computation of $\tilde{s}(b)$ is an identical).

Remark 4. The ACM RecSys 2013 Yelp lasted for more than 4 months. During that time there were two main opportunity to obtain a feedback: 1) external, through Kaggle LeaderBoard, and 2) internal, using homogeneous ensembling with CV-passports, see Section 4. Selection of the parameters was a consequence of both feedbacks, combined together.

Remark 5. The main objective of the smoothing (4) is to overcome "cold star" problem for both users and businesses.

Table 4. Final results of the ACM RecSys 2013 data mining contest: top ten teams, where "vsu" is our team (stands for Vyatka State University)

N	Team	Public	Private
1	BrickMover	1.19826	1.21251
2	**vsu**	1.20456	1.21552
3	Sergey	1.2112	1.22856
4	Merlion	1.2154	1.22724
5	Bryan Gregory	1.21923	1.23107
6	Gxav & Paul Duan	1.22078	1.23021
7	Biro Biro & Dmitry	1.2212	1.23118
8	YaTa	1.22518	1.23714
9	Xiong Cao & HIT	1.22589	1.23869
10	Li	1.22741	1.23699

3.2 Model N3: Transformation of the Available Properties between Users and Businesses

Business categories, which are available for both train and test parts of the data, represent key elements in the proposed recommender system. Using relations between users and businesses (which are represented by the reviews), where star ratings are not necessary, we transfer categories to users (compute sums of categories). Plus, we compute for users averages in the terms of latitudes and longitudes (locations of businesses). As a next step, we found expression of any category in the terms of votes and checkins (week-day and time). Further, we calculated expression of any user and business in the terms of votes and checkins, see Sections 5.1 and 5.2. It was noticed that $\hat{s}(user, business)$ - user's rating (named, also, as a star) of the business, maybe predicted with high quality in the case if we know average stars and numbers of reviews for both input parameters: user and business. In this particular project, we considered prediction of average stars (numbers of reviews are available) for users and businesses, using as an explanatory variables or features 1) locations (latitudes and longitudes), 2) numbers of reviews, 3) categories, 4) votes (processed by two different methods) and 5) checkins. Finally, we input all above data for users and businesses into

train.review and test.review sets, where we used predictions of the average stars for the test.users and test.businesses. As a predictor we applied GBM [9] and randomForest [10] functions in R.

Remark 6. The model, described in this section, represents a form of unsupervised learning. There are many cases when user from the test.review is not given in train.user or in test.user sets. However, we can obtain properties (in accordance with test.review), and use those properties as an input to the GBM predictor.

3.3 Computation of the Final Solution: Hybrid Model as an Ensemble of the Models NN 1, 2 and 3

We can ensemble the outcomes of the models N1 and N2 according to the formula (2), where the numbers of reviews equal to $r(u) + r(b)$. Note that $r(u)$ and $r(b)$ are different for the models N1 and N2. As a consequence, we shall produce ens_{12}.

Let us denote by $q_i = r_i(u) + r_i(b), i = 1, 2$, where $i = 1$ corresponds to the model N1, and $i = 2$ corresponds to the model N2.

Then, the final solution is calculated according to the formula

$$ens_{\text{final}} = \theta ens_{12} + (1 - \theta) ens_{\text{gbm}}, \tag{5}$$

where

$$\theta = \left(\frac{10}{q_1 + q_2 + 1} \right)^{\lambda}, \lambda = 0.2,$$

subject to the condition that $\theta \leq 1$, ens_{gbm} - heterogeneous ensemble as an outcome of the model N3.

Remark 7. In order to produce ens_{gbm} we used about 30 of different homogeneous ensembles, where each of which was based on the different configuration of the database. In particular, we conducted experiments with date (considering test.reviews as a future), gender (according to the name) and city.

Remark 8. In line with computation of the homogeneous ensembles as it was described in [6] (see, also, Section 4), we were able compute the corresponding CV-passports. The following results we observed in the terms of CV-passports and training error: 1) review.stars predictions $\{0.957, 0.8738\}$, 2) user.stars predictions $\{0.94, 0.852\}$, and 3) business.stars predictions $\{0.7838, 0.7049\}$. As an output, our model produces not only test.predictions, but, also, train.predictions, which maybe used for smoothing (similar to (4)).

In fact, we considered, also, model N4 - matrix factorisation via stochastic gradient descent [11], but it did not produce any significant improvement.

Table 5. Categorical data in a novel format (see Section 5), where any row corresponds to the particular business, "v11" - number of different categories. Columns from 1 to 10 are sequenced in a decreasing order according to the importance.

v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11
175	305	305	305	305	305	305	305	305	305	2
278	299	43	43	43	43	43	43	43	43	3
293	343	312	312	312	312	312	312	312	312	3
0	0	0	0	0	0	0	0	0	0	0
167	53	53	53	53	53	53	53	53	53	2
88	213	352	352	352	352	352	352	352	352	3
73	341	341	341	341	341	341	341	341	341	2
73	182	182	182	182	182	182	182	182	182	2
24	111	111	111	111	111	111	111	111	111	2
242	303	303	303	303	303	303	303	303	303	2
166	166	166	166	166	166	166	166	166	166	1
73	260	260	260	260	260	260	260	260	260	2
175	329	47	47	47	47	47	47	47	47	3
126	212	301	301	301	301	301	301	301	301	3
73	98	98	98	98	98	98	98	98	98	2
247	152	192	258	258	258	258	258	258	258	4
242	303	303	303	303	303	303	303	303	303	2
175	329	329	329	329	329	329	329	329	329	2
247	138	65	308	30	30	30	30	30	30	5
88	213	352	352	352	352	352	352	352	352	3
126	212	301	301	301	301	301	301	301	301	3
73	85	85	85	85	85	85	85	85	85	2
73	156	156	156	156	156	156	156	156	156	2
73	73	73	73	73	73	73	73	73	73	1
244	210	70	58	58	58	58	58	58	58	4
73	167	53	232	202	236	236	236	236	236	6
88	213	137	224	224	224	224	224	224	224	4
247	138	216	216	216	216	216	216	216	216	3
126	212	301	301	301	301	301	301	301	301	3
73	286	107	1	1	1	1	1	1	1	4
247	138	30	30	30	30	30	30	30	30	3

Using an External Data from the Yelp Web-Site. We note that some business star averages maybe easily collected manually from the Yelp web-site[2]. It is a fairly standard and publicly available procedure, which require no any special knowledge or skills. After that those averages maybe used in the model N1 of the proposed method. That means, the corresponding businesses will be transferred from test.business to train.business. Consequently, we shall observe some improvement (about 0.004) of the score in the terms of the root mean squared error. We do believe that some limited data warehousing (work with and collection of the real data) as an essential element of the Contest is highly

[2] `http://www.yelp.com/phoenix`

Table 6. Original checkin data (15 features), see Section 5.2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	5	22	24	7	26	0	16	33	8	0	30	0	0	20
0	9	45	36	0	58	0	30	64	0	0	38	0	0	2
2	21	32	42	3	48	0	51	40	6	0	34	0	0	5
0	3	41	23	1	44	0	30	80	2	0	48	0	0	12
4	2	20	14	6	10	0	18	24	9	0	42	0	0	40
0	6	49	30	4	37	0	24	72	4	0	44	0	0	13
0	2	22	19	4	21	0	20	36	5	0	43	0	0	17
0	3	24	18	4	24	0	19	39	4	0	38	0	0	16
0	4	23	17	2	34	0	22	47	2	0	32	0	0	6
0	8	28	22	1	46	0	12	50	1	0	18	0	0	2
0	1	15	7	0	18	0	8	26	0	0	19	0	0	3
0	2	27	15	2	32	0	18	48	2	0	33	0	0	11
1	27	38	42	2	76	0	14	50	3	0	23	0	0	9
4	17	34	35	9	42	0	20	46	9	0	43	0	0	26
0	2	25	20	4	24	0	20	38	3	0	39	0	0	13

desirable in some cases, as it may help participants to understand the data better. For example, the winner of the PAKDD 2010 data mining Contest[3] used external data, and it was highly rewarded. Generally, it is a good idea to encourage some initiative. In fact, it is far from easy to find out what sort of data to collect, where to find those data, and how to use new data in the model.

4 Calculation of the CV-Passports as a Validation Trajectories against All Training Data

In accordance with the principles of homogeneous ensembling, it appears to be natural to consider calculation of the decision function as an average of the large number of the single learners (or base classifiers), where any single learner is based on the randomly selected subsamples of observations and features.

Let us describe the proposed method in more details. Suppose that we are using about 75% of the available data for training. Then, remaining 25% of the data maybe used for the validation control. In line with the principles of cross-validation, we shall test stability of the validation results by considering a sequence of the random splittings.

Definition 1. *We are proposing to accumulate the validation results against all training samples in line with construction of the homogeneous ensemble. We shall call averaged validation trajectory as a CV-passport of the corresponding homogeneous ensemble.*

[3] `http://sede.neurotech.com.br/PAKDD2010/`

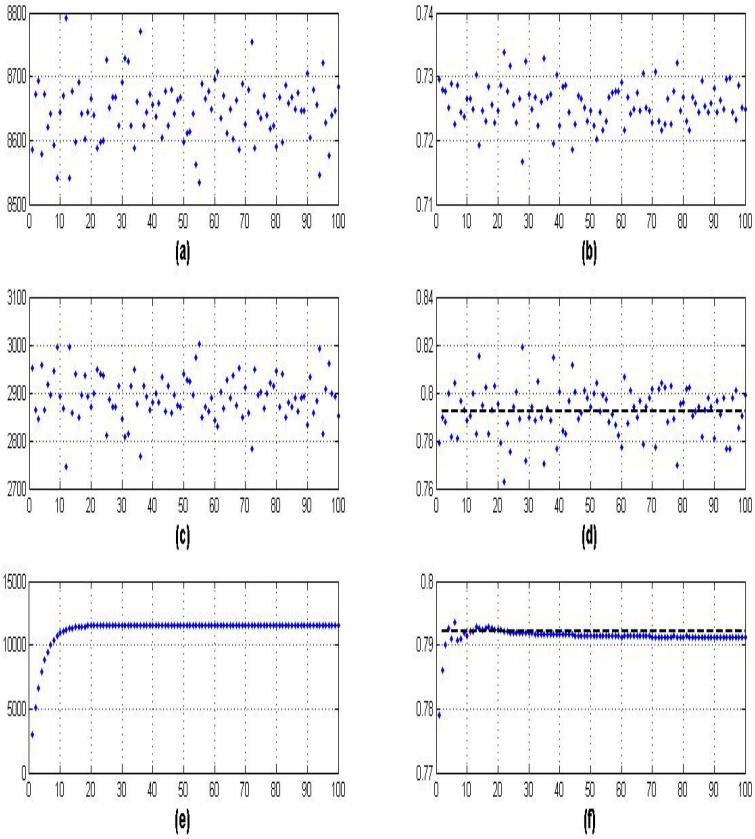

Fig. 2. Homogeneous ensembling applied to estimation of the star ratings of the test businesses, where horizontal axis determines index of random splitting. Left column shows numbers of samples used for (a) training, (c) testing and (e) calculation of CV-passport. Right column shows the corresponding errors in terms of RMSE (1).

Figure 2(d) illustrates the process of standard cross validation, where dashed black line corresponds to the average after 100 random splittings. It is interesting to note that all training samples will be filled after about 25 splittings, see Figure 2(e). We note, also, that evaluation results with CV-passports are slightly better compared to the the standard CV, see Figure 2(f).

5 Dimensional Reduction (Novel Format for Categorical Variables)

This section represents one particular component of the proposed method as a main novelty of the proposed recommendation system. We replaced ones in the

Table 7. Dimensional reduction: data of Table 6 after transformation (only 5 features)

9	12	6	4	3
9	6	3	12	4
8	6	4	9	12
9	12	6	3	8
12	15	9	3	8
9	3	12	6	4
12	9	3	6	8
9	12	6	3	8
9	6	12	3	8
9	6	3	4	12
9	12	6	3	8
9	12	6	3	8
6	9	4	3	2
9	12	6	4	3
12	9	3	6	8

matrix $\mathcal{A}$ by the corresponding positive RF-rating, zeros were kept intact. After that, we sorted any row in a decreasing order, and replaced positive values by the corresponding indexes (see Table 5). By definition, there are 11 columns in the secondary matrix of indexes $\mathcal{B}$, where first column contain indexes of the most important categories, second column contain indexes of the less important categories if available and so on. The maximum number of the different indexes is 10 (see Remark 2), and this number is given in 11th column. In the case if we don't know any categories for this business, all values in the corresponding row of the matrix $\mathcal{B}$ are set to zero.

5.1 Transfer of the Votes Data to all Users and Businesses

We have "votes = (useful, funny, cool)" information for train.users, and the task is to transfer this knowledge to the remaining data. As it was described in Section 3.2, we can find expression of all users in the terms of categories. Then, we can compute matrix $\mathcal{C}$ of $[categories, votes]$, using data from train.users. After that, we can explain all users and businesses in the terms of votes, and compute related secondary features: for example, normalised vectors of votes or sums of votes, $\log(sums)$ and so on.

5.2 Transfer of the Checkin Data to All Users and Businesses

The idea to transform checkin data (hour, day of the week, number of counts) to all users and businesses is about the same as in the case of votes, see above Section 5.1. Note that checkin information is not available for all businesses in the both train and test sets.

Firstly, we applied some smoothing. We transferred 7 days of the week to only 3 different categorical values: $\{1, 2, 2, 3, 3, 3, 1\}$. Also, we split 24 hours into

5 sub-intervals: $\{0:5, 6:9, 10:15, 16:19, 20:23\}$. The product of 3 and 5 will give us 15 new features. Then, we computed matrix $\mathcal{D}$ of $[n_c, 15]$, $n_c = 354$ categories and 15 checkins, using data from train and test businesses taking into account the numbers of checkins. After that, we can explain all users and businesses in the terms of checkins, see Table 6. Further, we applied dimensional reduction from 15 to 5 features according to the method similar to Section 5. Figure 2 illustrates the process of homogeneous ensembling for the estimation of the star ratings of the test businesses, applied to the database containing two main blocks: with 11 features as described in Section 5 and with 5 features as described in this section. Besides, we included 1) count numbers (for star ratings) and 2) geographical location with latitude and longitude.

6 Concluding Remarks

The proposed system represents an ensemble of three models, where the first one is content-based, the second one maybe classified as collaborative filtering, and the last one is a hybrid. We have found that the third model is the most influential, and produced very significant improvement. This model is based on the following two facts: 1) all businesses in review data maybe found in train.bus or test.bus; 2) all businesses are categorised. Further, we can transfer categories (plus some additional information) to all the users in review data. Consequently, we shall produce standard regression model with rectangular matrix of explanatory variable. Finally, it is proposed to solve a high-dimensionality problem using novel method as described in Section 5. Note that this method is general, and maybe applied for the variety of similar tasks.

References

[1] Takacs, G., Pilaszy, I., Nemeth, B., Tikk, D.: Scalable collaborative filtering approaches for large recommender systems. Journal of Machine Learning Research 10, 623–656 (2009)

[2] Ricci, F., Rokach, L., Shapira, B., Kantor, P. (eds.): Recommender systems handbook. Springer (2011)

[3] Jannach, D., Zanker, M., Felfernig, A., Friedrich, G. (eds.): Recommender systems an introduction. Cambridge University Press (2010)

[4] Schein, A., Popescul, A., Ungar, L., Pennock, D.: Methods and metrics for cold-start recommendations. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002), pp. 253–260. ACM, New York City (2002)

[5] Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. 20th Int. Conf. Very Large Data Bases, 32 p. (1994)

[6] Nikulin, V., Bakharia, A., Huang, T.-H.: On the evaluation of the homogeneous ensembles with CV-passports. In: Li, J., Cao, L., Wang, C., Tan, K.C., Liu, B., Pei, J., Tseng, V.S. (eds.) PAKDD 2013 Workshops. LNCS, vol. 7867, pp. 109–120. Springer, Heidelberg (2013)

[7] Spyromitros-Xioufis, E., Stachtiari, E., Tsoumakas, G., Vlahavas, I.: A hybrid approach for cold-start recommendations of videolectures. In: Smuc, T., Antulov-Fantulin, N., Morzy, M. (eds.) ECML/PKDD Workshop and Conference Proceedings, Discovery Challenge, pp. 29–40 (2011)
[8] Iaquinta, L., Semeraro, G.: Lightweight approach to the cold start problem in the video lecture recommendation. In: Smuc, T., Antulov-Fantulin, N., Morzy, M. (eds.) ECML/PKDD Workshop and Conference Proceedings, Discovery Challenge, pp. 83–94 (2011)
[9] Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Annals of Statistics 28, 337–374 (2000)
[10] Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)
[11] Nikulin, V., Huang, T.-H.: Unsupervised dimensionality reduction via gradient-based matrix factorization with two learning rates and their automatic updates. Journal of Machine Learning Research, Workshop and Conference Proceedings 27, 181–195 (2012)