OpenStudy: Recommendations of the Following Ten Lectures After Viewing a Set of Three Given Lectures

Vladimir Nikulin

Department of Mathematics
University of Queensland
vnikulin.uq@gmail.com

Abstract. We propose to use two lectures (out of the given triplet of three lectures) in order to define a direction of prediction, which includes the set of predicted lectures accompanied by the corresponding frequencies. The relevance of the whole predicted set is calculated according to the remaining third lecture. Further improvements were achieved with homogeneous ensembles, based on the random sampling known, also, as bagging. The experimental results were obtained online during the VideoLectures.Net ECML/PKDD 2011 Discovery Challenge (Track N2).

1 Introduction

VideoLectures.Net is a free and open access multimedia repository of video lectures, mainly of research and educational character. The lectures are given by distinguished scholars and scientists at the most important and prominent events like conferences, summer schools, workshops and science promotional events from many fields of science. The website is aimed at promoting science, exchanging ideas and fostering knowledge sharing by providing high quality didactic contents not only to the scientific community but also to the general public. All lectures, accompanying documents, information and links are systematically selected and classified through the editorial process taking into account also users' comments[1].

The tasks of the challenge were focused on making recommendations for video lectures, based on historical data from the VideoLectures.Net website[2].

According to [1], Open Social Learning Systems open new prospects for millions of self-motivated learners to access online a high quality materials. It is estimated that there will be 100 million students qualified to enter universities over next decade. Universities have responded to this need with Open Education Resources: thousands of free, high quality online courses, developed by hundreds of faculty, used by millions worldwide. Unfortunately, online courseware does not offer a supporting learning experience or the engagement needed to keep students motivated.

However, students today care deeply about their education. Four out of five stress about their grades. To create a successful online learning experience for those students, there are two main issues that need to be addressed: 1) creation of online study materials (content), and 2) an engaging online interaction experience (community).

[1] http://tunedit.org/challenge/VLNetChallenge
[2] http://videolectures.net/

The core problem (and the main subject of this study) lies around the second issue: how to give the right direction to a generation of learners, who live on the Internet, in the wide space of the available research/educational resources.

Recommender systems 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 [2]. 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.

The most of the methods presented and discussed in [2] were motivated by the famous Netflix Cup. Those methods, where matrix factorization is the most common, cannot be applied in our case directly, because the structure of the data is different. In our case we are dealing not with a specific users, but with abstract users who had seen in the past a triplet as a set of three given lectures, where an exact sequence of lectures is not given. We shall consider in Section 4 one suitable modification (as a prospective direction) of the gradient-based matrix factorization (GMF) [6] as an example of stochastic gradient descent algorithm.

Traditional data mining techniques such as association rules were tried with good results at the early stages of the development of recommender systems [3]. Frequent item sets, discovered as part of association rule mining, represent the least restrictive type of navigational patterns, since they focus on the presence of items rather than the order in which they occur within user session [4]. Frequency-based methods are the primary tool in the following below Sections 3.1 - 3.6. Note, also, that Markov decision processes provide a more advanced model for recommender systems (in the case if the sequence of the states is given). According to the Markov Chain Model, we are dealing with a finite space of possible states, and, using a maximum-likelihood estimate (applied to the historical data) as a transition function, we can formulate a prediction [5].

Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class [7]. In Section 3.6 we consider method of random resampling: it is supposed that using the hundreds of predictors (base learners) based on the subset of the whole training set we shall reduce the random factors. According to the principles of homogeneous ensembling, the final predictor represents an average of base predictors. As a reference, we mention random forests as a well-known example of successful homogeneous ensemble. However, the construction of random forests is based on another method, which is linked to the features but not to the samples.

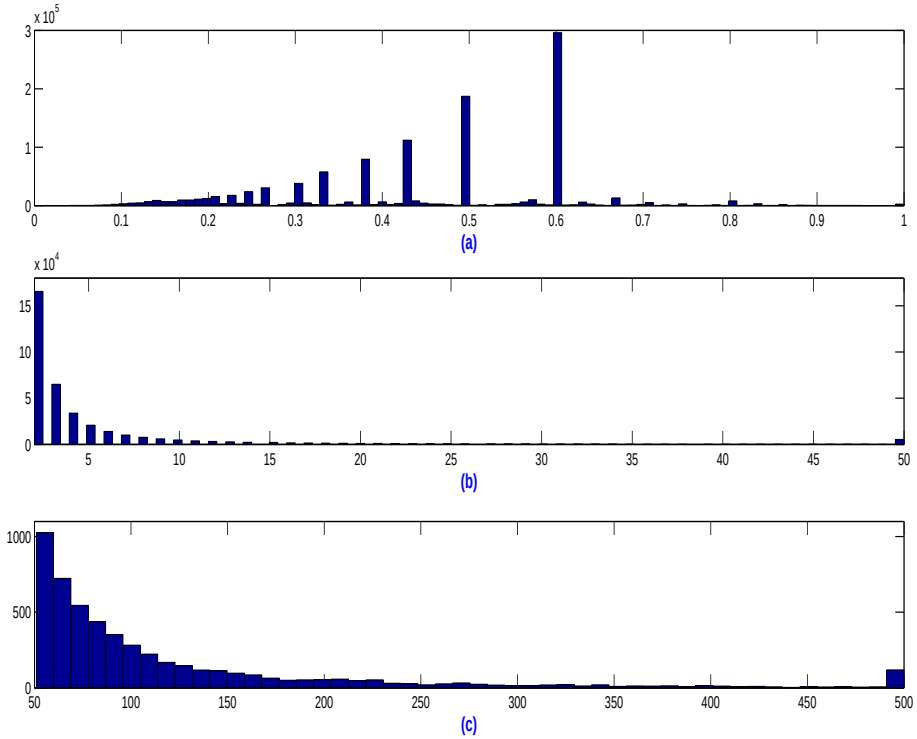

Fig. 1. Histograms of (a) triplet' frequencies; (b) pair' frequencies truncated to the level of 50; (c) pair' frequencies bigger than 50, and truncated to the level of 500, see for more details Section 2.

2 Data and some definitions

The training database includes two sets named 1) pairs P and 2) triplets T with two parts left and right, where the left part contains input triplets and the corresponding numbers of views, the right part contains output lectures and the corresponding numbers of views.

2.1 Pair data

Let us denote by I_P set of indexes corresponding to the pairs data. Any element of I_P represents a set (without order) of two lectures $I = \{a, b\}$, where $I \in I_P$. We shall understand under $P_I = P_{ab}$ number of times lectures a and b were viewed/seen together.

2.2 Triplet data

Let us denote by I_T set of indexes corresponding to the triplets data. The element with index $I \in T$ has two values (one for the left, and one for the right parts): $\tau_I = \{a, b, c\}$ is a triplet or a set of three lectures a, b and c. Under T_I we shall understand the number of times those three lectures were seen together. Further, L_I is a set of single lectures, which were seen after τ_I. Under $T_I(\ell), \ell \in L_I$, we shall understand the number of times lecture ℓ was seen after the triplet τ_I.

2.3 Graphical illustrations

Figure 1(a) illustrates histogram of empirical probabilities or frequencies

$$r_\tau(\ell) = \frac{T_I(\ell)}{T_I}, \ell \in L_I, I \in I_T, \tag{1}$$

where we replaced I by τ in the left side of (1), because there is unique correspondence between I and τ.

Figure 1(b-c) illustrates histogram of frequencies

$$P_I, I \in I_P,$$

where all the values in Figure 1(b) were truncated to 50, Figure 1(c) illustrates histogram of the values P_I bigger than 50, which were truncated to 500.

3 Methods

3.1 Predictions with couples

The task of the Challenge was to make predictions according to the test dataset V, which has the same structure as T (left part). In more details, the task was to make a recommendation of the ten the most appropriate lectures after viewing the given triplet.

Remark 1. As a particular and very important feature of this Challenge, we note the absence of the same triplets in both training T (left part) and test V sets. At the same time, we can report a very significant proportion of the same sets of *two* lectures (*couples*) in both training and test sets.

In total, we found $n_c = 34756$ couples, with number of record (which were extracted from the right part of T) per couple ranging from 1 to $m_c = 4020$. Note that any *triplet* may be considered as a set of three *couples*. There are $N_V = 60274$ triplets in the test set V, and 1) we did not find any related couples in the training set only in 116 cases, 2) we found one couple in 829 cases , 3) we found two couples in 4705 cases, and 4) we found all three couples in an absolute majority of 54624 cases.

Remark 2. Any lecture is identified by the index, where the biggest index is $n_L = 13251$. However, not all n_L lectures were used. We had assumed that the predicted lectures should be found in the right part of T, where we identified only $n_s = 5209$ different lectures.

Some preliminary definitions We shall explain how the system works in the terms of the n_s secondary indexes, because the transformation to the original n_L indexes is a trivial one. Our database was organised as follows. Squared matrix A with sizes $n_s \times n_s$ contains n_c different addresses of the matrix B with sizes $n_c \times m_c$.

Firstly, we shall find three couples $\alpha_{ij}, j = 1, \ldots, 3$, for any triplet $\tau_i, i = 1, \ldots, N_V$, in the test dataset V. Then, for any couple α_{ij} we shall find (according to the matrix A) the corresponding address $\beta(\alpha_{ij})$ (in the matrix B) and the number of records $n(\alpha_{ij})$, where $1 \leq \beta \leq n_c, 1 \leq n \leq m_c$.

Under the element of matrix B we shall understand predicted/recommended lecture ℓ and the corresponding frequency

$$\{\ell, r_\tau(\ell)\}, \tag{2}$$

where r_τ is defined in (1).

Remark 3. The main advantage of the above method is its speed: the algorithm will go through the whole test set V and will output required solution for the Track N2 within 5 min.

An update process Now, we shall describe the most critical step of the computational process. Any particular triplet τ from the test dataset is to be considered in an identical manner, so we omit index i in order to simplify notations.

Suppose that initially all the ratings are set to zero $s(\ell) = 0, \ell = 1, \ldots, n_s$, where $s(\ell)$ is the rating of the corresponding lecture ℓ which will be used for the final ranking as an output of this model.

This is the most important update formula

$$s(B_{\beta k}(1)) += B_{\beta k}(2), k = 1, \ldots, n(\alpha_j), j = 1, \ldots, 3, \tag{3}$$

where $B_{\beta k}(1)$ is the lecture index, and $B_{\beta k}(2)$ is the corresponding frequency defined in (2).

After computation of the vector s according to (3), we shall sort it in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest s (from the top to the bottom) are to be submitted as a solution.

Remark 4. In the case if the number of positive values in the vector s is smaller compared to the required 10, we shall generate remaining indexes at random assuming that the indexes are different compared to 1) the indexes of the input triplet τ_i plus 2) those indexes which were selected already.

The method, as described above produced the score on the Leaderboard **0.49568**, where the detailed definition (with numerical examples) of the Competition score is available from the web-site of the PKDD 2011 Contest.

3.2 Predictions with singles

Essentially, predictions with *singles* (single lectures) work similarly as predictions with couples. However, there are some differences, which could be treated as simplifications. We had found that the maximum number of the records corresponding to the single

lecture is $m_s = 77798$. Accordingly, the matrix $\widehat{B}$ (as a replacement to the matrix B in the previous Section 3.1) has sizes $n_s \times m_s$.

The model works in the following way: by definition, any triplet represents a set of three lectures $\ell_j, j = 1, \ldots, 3$. We shall find the number of records $1 \leq n(\ell) \leq m_s$, where $1 \leq \ell \leq n_s$.

An update process Again, initial ratings of lectures are set to zero: $s(\ell) = 0, \ell = 1, \ldots, n_s$.

This is the main update formula

$$s(\widehat{B}_{\ell_j k}(1)) += \widehat{B}_{\ell_j k}(2), k = 1, \ldots, n(\ell_j), j = 1, \ldots, 3. \tag{4}$$

After computation of the vector s according to (4), we shall sort it in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest s (from the top to the bottom) are to be submitted as a solution.

The method, as described above produced the score on the Leaderboard **0.33278**.

3.3 Predictions with pairs

Definition 1. *We shall call that the lectures a and b are P-linked if $P_{ab} \geq 1$. According to the symmetric matrix P, we define set $H(a)$ of all P-linked lectures to the lecture a.*

An update process As before, initial ratings of lectures are set to zero: $s(\ell) = 0, \ell = 1, \ldots, n_s$. Then, we shall apply an update formula

$$s(d) += P(\ell_j, d), d \in H(\ell_j), j = 1, \ldots, 3, \tag{5}$$

where an interpretation/definition of the lectures ℓ_j is the same as in (4).

After computation of the vector s according to (5), we shall sort it in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest s (from the top to the bottom) are to be submitted as a solution.

The method, as described above produced the score on the Leaderboard **0.12677**.

Remark 5. The solution, as described in this section, was recommended by the Organisers on the forum as "*simple pairs solution*".

Note, also, that during our numerous experiments we made a very interesting observation/discovery.

Remark 6. Statistics defined in (3-5) represent a sum of frequencies. It is very interesting to note that the results will be significantly poorer if we shall apply an average of frequencies as an alternative to the sums.

3.4 Predictions with weighted couples

According to the above three sections, predictions with the couples produced best results. We decided to go further and to take into account remaining third lectures ϕ and ψ in both training and test sets.

Motivation: in the case if remaining ("extra") lectures ϕ and ψ are closer (have bigger number of the joint views according to the pairs data), the predicted direction, corresponding to the related *couple*, must be given bigger weight.

As it was discussed in Remark 1, lectures ϕ and ψ are different by definition. In other words, the corresponding ("similar") triplets in the training and test datasets may be represented as

$$\alpha_j \cup \phi_j, \ \alpha_j \cup \psi_j,$$

where $\phi_j \neq \psi_j, j = 1, \ldots, 3$.

An update process As before, initial ratings of lectures are set to zero: $s(\ell) = 0, \ell = 1, \ldots, n_s$. Then, we can re-write (3) in this way

$$s(B_{\beta k}(1)) += w(P(\phi_j, \psi_j))B_{\beta k}(2), k = 1, \ldots, n(\alpha_j), j = 1, \ldots, 3, \tag{6}$$

where w is an increasing weight function. In our final submission we used very simple linear function: $w(x) = 0.01 \cdot x + 0.005$.

After computation of the vector s according to (6), we sort it in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest s (from the top to the bottom) are to be submitted as a solution.

The method with weighted couples, as described above, produced very significant improvement on the Leaderboard **0.58145**.

3.5 Predictions with weighted singles

This section may be regarded as an extension of Section 3.2. In some sense, prediction with weighted singles is similar to the prediction with weighted couples, Section 3.4. However, there are some differences. In the case of singles, we are defining direction of the prediction according to the single lectures. Accordingly, we have two other ("extra") lectures, which should be compared properly with two lectures in the corresponding triplet of the training data.

An update process Updates were conducted according to

$$s(\widehat{B}_{\ell_j k}(1)) += w(\phi_{1j}, \phi_{2j}; \psi_{1j}, \psi_{2j}) \cdot \widehat{B}_{\ell_j k}(2), k = 1, \ldots, n(\ell_j), j = 1, \ldots, 3, \tag{7}$$

where $w(\phi_{1j}, \phi_{2j}; \psi_{1j}, \psi_{2j})$

$$= 0.0005(P(\phi_{1j}, \psi_{1j})P(\phi_{2j}, \psi_{2j}) + P(\phi_{1j}, \psi_{2j})P(\phi_{2j}, \psi_{1j})) + 0.01.$$

The motivation behind the above formula is a very simple: we must ensure that any "extra" lecture from the test triplet is close to at least one "extra" lecture from the train triplet.

After computation of the vector s according to (7), we sort it in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest s (from the top to the bottom) are to be submitted as a solution.

The method, as described in this section, produced the score **0.4529** on the Leaderboard.

3.6 Resampling method (the final recommender)

In this section computation of the single ranking vector s was based on 75% of randomly selected samples. In an absolute majority of all 60274 test instances, the number of positive components of the vector s defined in (6) is greater than 100. So we shall consider this case only.

Let us denote vector of secondary ratings as z, which is set to zero at the beginning of the whole resampling process. We conducted 200 random samplings (global iterations). After any global iteration, only 100 top lectures (components of the vector z) received increments ranging from 1 to 100 votes (bigger for better performance). The method which we used within any global iteration (base learner) is described in Section 3.4.

After completion of all 200 global iterations, we sorted vector z in a decreasing order, and the arguments (indexes of the lectures) corresponding to the ten biggest z (from the top to the bottom) were submitted as a solution.

The final model with resampling, as described above, produced the following score on the Leaderboard **0.58727**. This solution was used as a final.

Table 1. Distances (8) between five solutions described in Sections 3.1 - 3.4 and 3.6.

N	Method	Score	1	2	3	4	5
1	couples	0.49568	0	0.2605	0.2137	0.6394	0.6517
2	singles	0.33278	0.2605	0	0.5832	0.4269	0.4327
3	pairs	0.12677	0.2137	0.5832	0	0.1565	0.1664
4	wgt-couples	0.58145	0.6394	0.4269	0.1565	0	0.91
5	resampling	0.58727	0.6517	0.4327	0.1664	0.91	0

3.7 Statistical comparison of different solutions

The distances in the above Table 1 were computer using simplified version of the PKDD 2011 Contest evaluation method. Any solution represents an integer matrix of $N_V \times 10$, $\mathcal{T} = 10N_V$ integer indexes in total. By comparing two matrices, we shall find the number of common indexes (intersection) in any row. The total number of all intersections will give us a numerator $\mathcal{R}$, and the required distance represents a ratio

$$\mathcal{D} = \frac{\mathcal{R}}{\mathcal{T}}. \tag{8}$$

3.8 Computation time

A Linux multiprocessor workstation with speed 3.2GHz and 16GB RAM was used for the most of the computations, which were conducted according to the specially developed codes in C. The computation of the final solution, as described in Section 3.6, was conducted overnight and took about 12 hours.

4 Gradient-based matrix factorization

The main idea behind an approach of this section is to factorize the transition matrix $\mathbf{X}$ of a Markov chain (or matrix of frequencies) [9].

By definition, $\mathbf{X}$ is a squared symmetrical matrix with size n_s, and any element of $\mathbf{X}$ reflects similarity between the corresponding lectures (the bigger value indicates the higher level of similarity).

Further, we shall consider factorization of the matrix $\mathbf{X}$

$$\mathbf{X} \sim \mathbf{G}\mathbf{G}', \tag{9}$$

where factor matrix $\mathbf{G}$ has sizes $n_s \times k$.

We shall compute matrix $\mathbf{X}$ according to an update formula, which is very similar to (4). Initially, all the values of the matrix $\mathbf{X}$ are set to zero. The update process will be conducted according to the formula

$$x(\widehat{B}_{\ell k}(1), \ell) += \widehat{B}_{\ell k}(2),\ \ x(\ell, \widehat{B}_{\ell k}(1)) = x(\widehat{B}_{\ell k}(1), \ell), \tag{10}$$

where $k = 1, \ldots, n(\ell), \ell = 1, \ldots, n_s$.

In this section, boldface capital letters denote matrices or vector-columns, while normal letters denote elements of matrices. Also, it will be convenient for us to use notation $x_{ab} = x(a, b)$.

We now describe the procedure for undertaking the matrix factorization (9). The matrix factorization represents a gradient-based optimisation process with the objective to minimise the following squared loss function:

$$\mathcal{L}(\mathbf{A}) = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} e_{ab}^2, \tag{11}$$

where $e_{ab} = x_{ab} - \sum_{f=1}^{k} g_{af} g_{fb}$.

The above target function (11) includes in total kn regulation parameters and may be unstable if we minimise it without taking into account the mutual dependence between elements of the factor matrix $\mathbf{G}$.

As a solution to the problem, we can go consequently through all the differences e_{ab}, minimising them as a function of the particular parameters which are involved in the definition of e_{ab}. Compared to usual gradient-based optimisation, in our optimisation model we are dealing with two sets of parameters, and we should mix uniformly updates of these parameters, because these parameters are dependent.

Algorithm 1: Gradient-based matrix factorization.

1. **Input**: $\mathbf{X}$ - similarity matrix.
2. Select M - number of global iterations; $k \geq 1$ - number of factors; $\lambda > 0$ - learning rate.
3. Initial similarity matrix $\mathbf{G}$ is generated randomly.
4. **Global** cycle: repeat M times the following steps 5 - 15:
5. **external**-cycle: for $a = 1$ to $n_s - 1$ repeat steps 6 - 15:
6. **internal**-cycle: for $b = a + 1$ to n_s repeat steps 7 - 15:
7. compute prediction $S = \sum_{f=1}^{k} g_{af} g_{fb}$;
8. compute error of prediction: $\Delta = x_{ab} - S$;
9. internal factors-cycle: for $f = 1$ to k repeat steps 10 - 15:
10. compute $\alpha = g_{af} g_{fb}$;
11. update $g_{af} \Leftarrow g_{af} + \lambda \cdot \Delta \cdot g_{fb}$ (see (12a));
12. $\Delta \Leftarrow \Delta + \alpha - g_{af} g_{fb}$;
13. compute $\alpha = g_{af} g_{fb}$;
14. update $g_{fb} \Leftarrow g_{fb} + \lambda \cdot \Delta \cdot g_{af}$ (see (12b));
15. $\Delta \Leftarrow \Delta + \alpha - g_{af} g_{fb}$;
16. **Output**: $\mathbf{G}$ - matrix of latent factors.

The following partial derivatives are necessary for Algorithm 1:

$$\frac{\partial e_{ab}^2}{\partial g_{af}} = -2 e_{ab} g_{fb}, \tag{12a}$$

$$\frac{\partial e_{ab}^2}{\partial g_{fb}} = -2 e_{ab} g_{af}, \tag{12b}$$

where $a = 1, \ldots, n_s - 1, b = a + 1, \ldots, n_s$.

Remark 7. The content within this section represents rather a direction for a prospective work. We are thinking that transformation by the logit function of the values of the matrix $\mathbf{X}$ will work better with Algorithm 1.

4.1 Ranking of the lectures with the matrix of latent factors

As an outcome, Algorithm 1 produces the matrix $\mathbf{G}$ of latent factors. Accordingly, we can characterize any lecture by the corresponding vector-row of k numerical values, and can compute a proximity measure with the given three lecture (triplet) from the test set V. The smaller value of the distance indicates higher preference.

5 Concluding remarks

We fully agree with [10] that the superiority of new algorithms should always be demonstrated on an independent validation data. In this sense, an importance of the data

mining contests is unquestionable. The rapid popularity growth of the data mining challenges demonstrates with confidence that it is the best known way to evaluate different models and systems.

In general terms, we are satisfied with our results. However, due to the lack of available time, we did not find an efficient way how to construct heterogeneous ensembles. For example, the outcomes of the methods of weighted singles and couples are very different. Nevertheless, both results are competitive, and it is very important to find out how to exploit the differences in order to produce more advanced solution.

Also, we did not explore in details gradient-based matrix factorization, which is presented in Section 4. We can expect that this method may lead (after proper adjustment and modification) to the solution, which will be different and competitive at the same time.

Acknowledgments

This work was supported by a grant from the Australian Research Council. Also, we are grateful to the Organisers of the PKDD 2011 data mining Contest for this stimulating opportunity. Many thanks to the reviewers for very helpful comments and advices, including some very important references.

References

1. Ram A.,Ai H.,Ram P. and Sahay S. (2011) Open Social Learning Communities. *In International Conference on Web Intelligence, Mining and Semantics*, Sogndal, Norway.
2. Takacs G., Pilaszy I., Nemeth B. and Tikk D. (2009) Scalable Collaborative Filtering Approaches for Large Recommender Systems. *Journal of Machine Learning Research*, **10**, 623-656.
3. Agrawal R. and Srikant R. (1994) Fast Algorithms for Mining Association Rules. *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, 32p.
4. Mobasher B., Dai H., Luo T. and Nakagawa M. (2002) Using Sequential and Non-Sequential Patterns in PredictiveWeb Usage Mining Tasks. *ICDM 2002*, 4p.
5. Shani G., Heckerman D. and Brafman R. (2005) An MDP-Based Recommender System. *Journal of Machine Learning Research*, **6**, 1265-1295.
6. Nikulin V., Huang T.-H., Ng S.-K., Rathnayake S. and McLachlan G.J. (2011) A Very Fast Algorithm for Matrix Factorisation. *Statistics and Probability Letters*, **81**, 773-782.
7. Breiman L. (1996) Bagging Predictors. *Machine Learning*, **24**, 123-140.
8. Breiman L. (2001) Random Forests. *Machine Learning*, **45**, 5-32.
9. Rendle S., Freudenthaler C. and Schmidt-Thieme L. (2010) Factorizing Personalized Markov Chains for Next-Basket Recommendation. *WWW 2010, North Carolina, USA*, 811-820.
10. Jelizarow M., Guillemot V., Tenenhaus A., Strimmer K. and Boulesteix A.-L. (2010) Over-optimism in bioinformatics: an illustration. *Bioinformatics*, **26**(16), 1990-1998.