УДК 004.048 **doi:** https://doi.org/10.20998/2522-9052.2021.4.08

Svitlana Krepych, Iryna Spivak

West Ukrainian National University, Ternopil, Ukraine

IMPROVEMENT OF SVD ALGORITHM TO INCREASE THE EFFICIENCY OF RECOMMENDATION SYSTEMS

Abstract. Many existing websites use recommendation systems for their users. They generate various offers for them, for example, similar products or recommend the people registered on this site with similar interests. Such referral mechanisms process vast amounts of information to identify potential user preferences. Recommendation systems are programs that try to determine what users want to find, what might interest them, and recommend it to them. These mechanisms have improved the interaction between the user and the site. Instead of static information, they provide dynamic information that changes: recommendations are generated separately for each user, based on his previous activity on this web resource. Information from other visitors may also be taken into account. The methods of collecting information provided by the Internet have greatly simplified the use of human thought through collaborative filtering. But, on the other hand, the large amount of information complicates the implementation of this possibility. For example, the behavior of some people is quite clearly amenable to modeling, while others behave completely unpredictably. And it is the latter that affect the shift of the results of the recommendation system and reduce its effectiveness. An analysis of Internet resources has shown that most of the recommendation systems do not provide recommendations to users, and the part that does, for example, offers products to the user, selects recommendations manually. Therefore, the task of developing methods for automated generation of recommendations for a limited set of input data is quite relevant. The problems of data sparseness, new user problem, scalability of the widely used SVD algorithm for the development of such recommendation systems are proposed to be eliminated by improving this algorithm by the method of the nearest k-neighbors. This method will allow you to easily segment and cluster system data, which will save system resources.

Keywords: recommendation system; SVD algorithm; k-nearest neighbors' method; data sparseness; scalability; clustering.

Introduction

The problem of choice has always existed. Today is no exception. However, nowadays this problem is especially acute, because in many cases the wrong choice leads to a loss of profit. To solve this problem, recommendation systems have been created [1]. By analyzing the similarities of users and their ratings, the recommendation system can offer similar objects (books, music, videos, etc.).

However, recommendation systems have a number of disadvantages, in particular:

- sparse data - recommendation systems operate with a huge amount of data, and most users do not participate in the evaluation of objects and, accordingly, do not evaluate them;

- the problem of a new user - with the advent of new users or objects of evaluation there is a problem of similarity due to lack of information. Yes, new users can't get feedback until they rate certain items themselves;

- scalability - the next problem is related to the growing number of users in the system.

There are algorithms that lack some of the above disadvantages, but the quality of the recommendations of such algorithms is quite low. To address these and other shortcomings of recommendation systems, it is proposed to use the SVD algorithm. It is simple, incredibly flexible and, importantly, shows good quality recommendations.

Statement of task

The general principle of operation of recommendation systems is presented in Fig. 1. Methods of collecting information by recommendation systems are divided into two types: explicit data collection and implicit. When explicitly collecting data, the user provides all the necessary information for further processing. This can include both standard personal data and evaluative judgments of content from various fields. If a visitor refuses to provide information about himself, the following method becomes relevant - implicit data collection. There is a kind of tracking of a person, during which the user's actions are recorded by a special program that constantly collects the necessary information for further analysis and application [2].

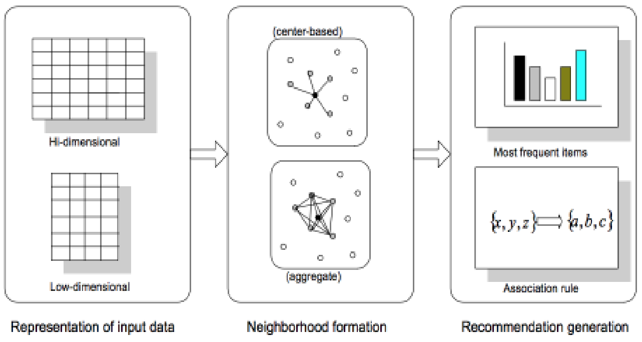


Fig. 1. The principle of operation of recommendation systems

To solve the problem with a new user and a high sparse matrix, it is recommended to use a regulator. Regularization solves the problem of retraining. That is, when the new model works well with test sample data, but behaves unexpectedly with data samples that did not participate in the training.

To solve the problem of scalability, it is proposed to use the method of nearest k-neighbors. This method allows us to easily segment and cluster system data. This saves resources, because the system will use only the data of those users who are with it in the same cluster. An analysis of Internet resources revealed that most of them don't provide recommendations to users. The part of the resources that make this, for example, offers the user products, goods, etc., selects recommendations manually, the process is not automated [3]. Therefore, the task of developing methods and software to create recommendations interesting materials for user based on a limited set of input data is actual.

Research result

Suppose we have a matrix R, that consists of ratings r (in our case, the number of ratings) that users i have assigned to products a. It turns out a matrix $R=(r_{ia})^{N.M}$, in which the ratings known to us are written down. As a rule, one user will not be able to appreciate a significant proportion of products. Therefore, it is unlikely that there will be many products that are ready to appreciate by a significant part of users [4]. This means that the matrix R is very sparse. We apply to it the so-called singular decomposition in the form:

$$R=UDV^T\,, \qquad (1)$$

where the matrix U and V – orthogonal, and D – diagonal. R – large size matrix $N\times M$, but of small rank f, that is, it can be decomposed into the product of the matrix $N\times f$ and matrix $f\times M$, thereby drastically reducing the number of parameters from $N\times M$ to $(N+M)\times f$. The main property of the SVD method is that it gives the optimal approximation if in the matrix D just leave the straight f the first diagonal elements, and the rest to put equal 0, that is

$$R = UDV^T =$$

$$=U\begin{pmatrix}\sigma_1 & 0 & ... & 0\\ 0 & \sigma_2 & ... & 0\\ ... & ... & ... & ...\\ 0 & 0 & ... & \sigma_k\end{pmatrix}V^T\approx U\begin{bmatrix}\sigma_1 & ... & 0\\ ... & ... & ...\\ 0 & ... & \sigma_f\end{bmatrix}V^T\,. \qquad (2)$$

In the diagonal matrix D elements are sorted by the size: $\sigma_1\ge\sigma_2\ge...\ge\sigma_k$, to zeroed the last elements, it means to zeroed the smallest elements. And f is selected based on the size of the singular values of the matrix, the same diagonal elements of the matrix D. It is desirable to discard as many elements as possible. In the case of recommendation systems, it turns out that we represent each user with the vector of f factors U_i, and each product is a vector with f factors V_j. Next, to predict the users i rating of the product j, we take their scalar product. We have a task: according to the known evaluations of known products to predict how well each product will be appreciated by the new user.

Introduce the so-called basic predictors $b_{i,a}$, which consist of basic predictors of individual users b_i i and individual products b_a, as well as just the overall component of the average rating on the base μ:

$$b_{i,a}=\mu+b_i+b_a\,, \qquad (3)$$

where μ – of the average rating on the base; b_i – basic predictors of individual users; b_a the average rating of each product a.

To determine only the basic predictors, you need to find the following μ, b_i and b_a, for which $b_{i,a}$ the best approximate the available ratings. After obtaining the basic predictors, the residues will be comparable with each other and based on them obtained reasonable values for the factors:

$$\hat{r}_{ia}=\mu+b_i+b_a+v_a^T u_i\,, \qquad (4)$$

where v_a – vector of factors representing the product a; u_i – vector of factors representing the user i.

Now we can return to the original problem and formulate it precisely: we need to find predictors that minimize the next function:

$$L(\mu,b_i,b_a,v_a,u_i)=\sum_{(i,a)\in D}(r_{ia}-\hat{r}_{ia})^2\to\min\,. \qquad (5)$$

Function $L(\mu,b_i,b_a,v_a,u_i)$ can be minimized based on the gradient descent method [5]. However, over time it will be very costly, because we do not know the location of the vectors we need, and we still need to minimize the error.

In order to reduce the runtime of the SVD algorithm and minimize the magnitude of the error, we use the k-nearest neighbor (kNN) algorithm for large amounts of data. This will allow us to use it with a very sparse matrix and with large amounts of data, as well as save resources, because the system will use the data only of those users who are with the current user in the same group. We briefly describe the essence of this method.

Method kNN – is one of the simplest methods of classification. Due to its simplicity and scalability, it is extremely effective for classification. The task of classification in machine learning means the task of assigning an object to one of the predefined classes on the basis of its formalized features. Each of the objects in this problem is represented as a vector in N-measuring space, each dimension of which is a description of one of the features of the object [6].

To classify each of the objects of the test sample, we must perform the following operations:

- calculate the distance to each of the objects of the training sample;
- select k objects of the training sample, the distance to which is minimal;
- the class of the object to be classified, – s the class that most often occurs among k the nearest neighbors.

So there is a set of objects for each of which a class is specified.

Now we need to divide this set into two parts: the training sample and the test sample. This breakdown describes the code shown in Listing 1.

Not only the Euclidean distance can be used to determine the distance between objects. Manhattan distance, cosine measure, Pearson's correlation criterion and others are also widely used.

```
   def splitTrainTest (data, testPercentage):
       trainDataArray = []
       testDataArray  = []
       for row in data:
           if random.random() < testPercentage:
               testDataArray.append(row)
           else:
               trainDataArray.append(row)
       return trainDataArray, testDataArray
```

Listing 1

Now, having a training sample, you can implement the classification algorithm (Listing 2).

```
   def classifyKNN (trainDataArray, testDataArray,
k, amountOfClasses):
       def dist (a, b):
           return  math.sqrt((a[0]  -  b[0])**2  +
(a[1] - b[1])**2)
       testLabels = []
       for testPoint in testDataArray:
           testDistance   =   [   [dist(testPoint,
trainDataArray[i][0]),  trainDataArray[i][1]]  for  i
in range(len(trainDataArray))]
           stat     =     [0      for      i      in
range(amountOfClasses)]
           for d in sorted(testDistance)[0:k]:
               stat[d[1]] += 1
           testLabels.append( sorted(zip(stat,
range(amountOfClasses)), reverse=True)[0][1] )
       return testLabels
```

Listing 2

In the Fig. 2 an example of the operation of the classification algorithm for 40 elements when $k = 3$.

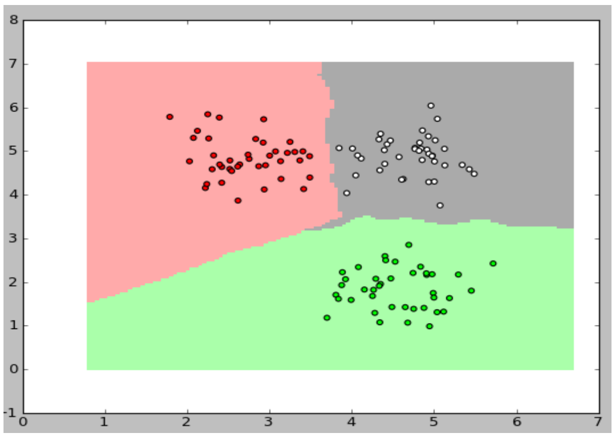


Fig. 2. The work of the kNN classifier algorithm

For the correct operation of the method with the new data, the class was modified to the form in Listing 3. Schematically, the operation of the advanced method is represented by a UML state diagram in the Fig. 3. The verification of the effectiveness of the application of the improved method to increase the efficiency of the recommendation systems was carried out on the basis of the calculation of the average absolute error:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|r_{ia} - \hat{r}_{ia}\right|, \qquad (6)$$

where n – number of users.

```
   class NearestNeighborsStrategy():
      def _set_similarity(self, data_model,
similarity, dist, neirhood_size):
         if not isinstance(self.similarity,
UserSimilarity) \
            or not distance ==
self.similarity.distance:
   neirhood_count = neirhood_count if not
neirhood_count else neirhood_count + 1
          self.similarity =
UserSimilarity(data_model, distance,
neirhood_count)
      def user_neighborhood(self, user, data_model,
similarity='user_similarity', dist=None,
neirhood_count=False, **params):
         minimal_similarity =
params.get('minimal_similarity', 0.0)
         sampling_rate = params.get('sampling_rate',
1.0)
         data_model = self._sampling(data_model,
sampling_rate)
   if dist is None:
         dist = euclidean_distances_method
         if similarity == 'user_similarity':
             self._set_similarity(data_model,
similarity, dist)
         else:
             raise ValueError('wrong similarity
method')
         neighborhood = [tmp_user_id for tmp_user_id,
score in self.similarity[user]
   if not np.isnan(score) and score >=
minimal_similarity and user != tmp_user_id]
         return neighborhood
```

Listing 3

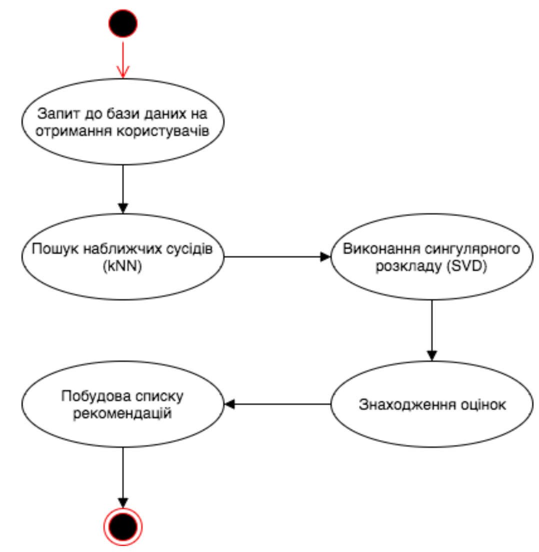

Запит до бази даних на отримання користувачів
Пошук наближчих сусідів (kNN)
Виконання сингулярного розкладу (SVD)
Знаходження оцінок
Побудова списку рекомендацій

Fig. 3. Schematic work of the advanced method

Fig. 4 shows a comparison of the usual recommendation method and its improvement based on the kNN method.

The software system of book recommendations for users was developed based on the improved method [7]. The principle of operation of the system is as follows: after the user has successfully logged in, a list of books available in the system is available to him (Fig. 5). Then the user can give their ratings to books, add new ones if the book is not available on the site. Through ratings, reviews and reviews, the system can predict the ratings that the user could give to other books (Fig. 6).

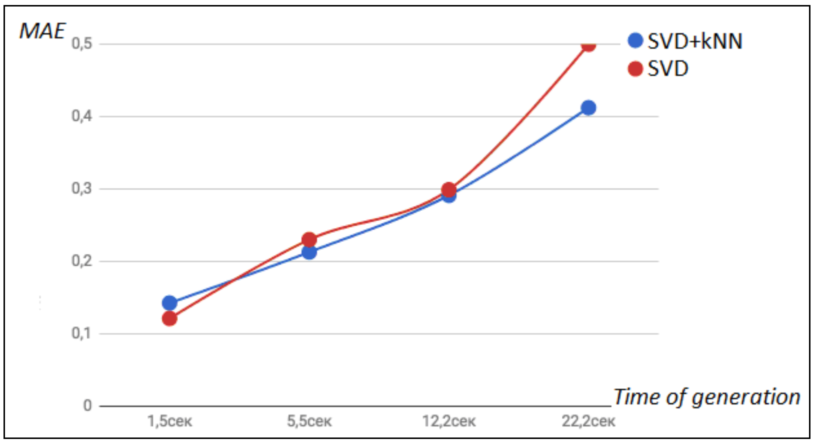


Fig. 4. The dependence of the increase in errors
in the system relative to the time of generation of user query results

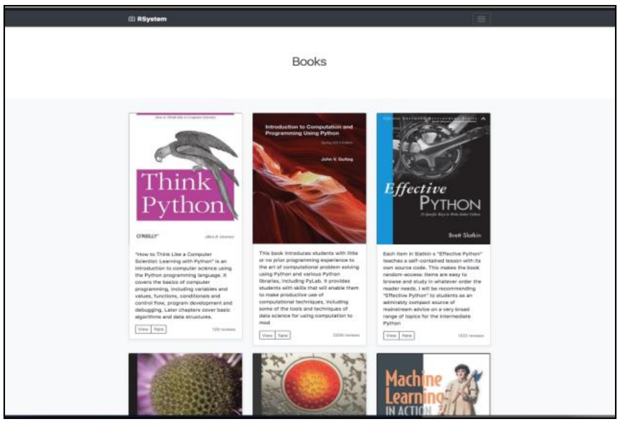


Fig. 5. Display page of all books

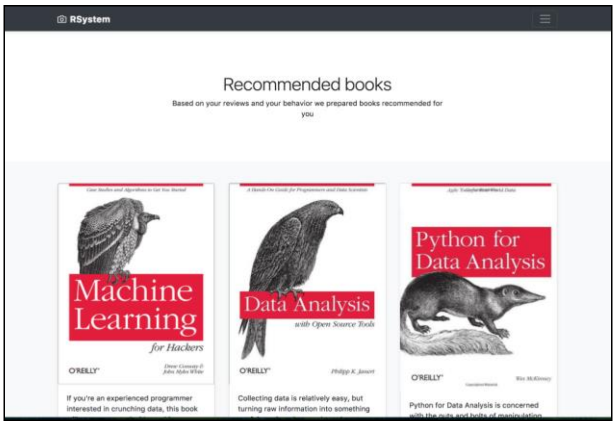
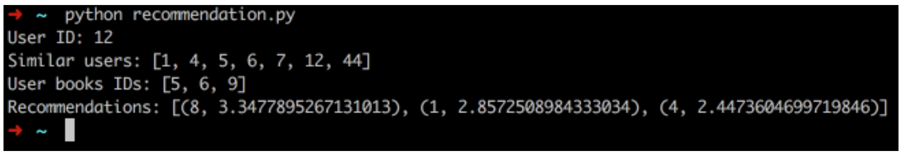

RSystem
Recommended books
Based on your reviews and your behavior we prepared books recommended for you
Machine Learning
Data Analysis
Python for Data Analysis

Fig. 6. Book recommendations page

Fig. 7 shows the result of the implementation of the recommendation module.



```
➜  ~ python recommendation.py
User ID: 12
Similar users: [1, 4, 5, 6, 7, 12, 44]
User books IDs: [5, 6, 9]
Recommendations: [(8, 3.3477895267131013), (1, 2.8572508984333034), (4, 2.4473604699719846)]
➜  ~ 
```

Fig. 7. The result of the recommendation module

The image shows that the system recommends the user to view books with ID 8, 1 and 4 with their recommendation ratios.

Conclusions

The problem of increase of efficiency of work of recommendation systems is considered in the work. It is shown that the most used algorithm for providing recommendations in such systems is the SVD algorithm. However, this algorithm has a number of disadvantages, including sparse data, the problem of a new user and the problem of scalability. To solve these problems, it was proposed to improve the SVD algorithm by using the method of nearest neighbors, which saves a lot of time and computing resources, because the system to provide recommendations uses data only from users who are in the same cluster. The quality of both methods was tested on the example of book recommendations to the user and a comparative graph of the error of their application.

References

1. Roisner, M. (2020), *How recommender systems work*, available at: https://habrahabr.ru/company/yandex/blog/241455/
2. Goncharov, M. (2020), Data Mining: Recommender Systems, available at: http://kek.ksu.ru/EOS/WM/50_132-670.pdf.
3. Koroleva, D. (2020), *Analysis of recommendation systems learning algorithms*, available at: http://engjournal.ru/articles/816/816.pdf.
4. (2021), *Singular value decomposition*, available at: http://www.machinelearning.ru/wiki/index.php?title=%D0%A1%D0%B8%D0%BD%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%BE%D0%B5_%D1%80%D0%B0%D0%B7%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5
5. (2021), *Gradient descent: everything you need to know*, available at: https://neurohive.io/ru/osnovy-data-science/gradient-descent/
6. (2021), *The method of k-nearest neighbors*, available at: http://om.univ.kiev.ua/users_upload/15/upload/file/pr_lecture_03.pdf

7. Krepych, S.Ya., Dzhulii, M.V. (2017), “Software system to support the selection of books on fuzzy criteria”, *Modern computer information technologies: Proceedings of the All-Ukrainian conference with inte. participation ASIT’2017*, Ternopil: TNEU, p. 144-145.

Received (Надійшла) 29.06.2021
Accepted for publication (Прийнята до друку) 20.10.2021

ВІДОМОСТІ ПРО АВТОРІВ/ ABOUT THE AUTHORS

Крепич Світлана Ярославівна – кандидат технічних наук, доцент, доцент кафедри комп’ютерних наук, Західноукраїнський національний університет, Тернопіль, Україна;
Svitlana Krepych – Candidate of Technical Sciences, Associate Professor, Associate Professor of Computer Science Department, Western Ukrainian National University, Ternopil, Ukraine:
e-mail: msya220189@gmail.com; ORCID ID: https://orcid.org/0000-0001-7700-8367

Співак Ірина Ярославівна - кандидат технічних наук, доцент, доцент кафедри комп’ютерних наук, Західноукраїнський національний університет, Тернопіль, Україна;
Iryna Spivak – Candidate of Technical Sciences, Associate Professor, Associate Professor of Computer Science Department, Western Ukrainian National University, Ternopil, Ukraine:
e-mail: spivak.iruna@gmail.com: ORCID ID; https://orcid.org/0000-0003-4831-0780

Удосконалення алгоритму SVD для підвищення ефективності рекомендаційних систем

С. Я. Крепич, І. Я. Співак

Анотація. Велика кількість на сьогодні існуючих веб-сайтів використовують рекомендаційні системи для своїх користувачів. Вони генерують їм різні пропозиції, наприклад, подібні товари або рекомендують людей, зареєстрованих на цьому сайті, зі схожими інтересами. Такі рекомендаційні механізми обробляють величезні обсяги інформації для позначення потенційних переваг користувачів. Рекомендаційні системи - це програми, які намагаються визначити, що хочуть знайти користувачі, що може їх зацікавити і рекомендують їм це. Ці механізми вдосконалили взаємодію між користувачем і сайтом. Взамін статичної інформації вони надають динамічну інформацію, яка змінюється: рекомендації генеруються окремо для кожного користувача, ґрунтуючись на його попередній активності на даному веб-ресурсі. Також може враховуватися інформація, що надходить від інших відвідувачів. Методи збору інформації, що надаються Інтернетом, значно спростили використання людської думки за допомогою коллаборативної фільтрації. Але, з іншого боку, великий обсяг інформації ускладнює втілення цієї можливості. Наприклад, поведінка одних людей досить ясно піддається моделюванню, в той час як інші поводяться абсолютно непередбачувано. І саме другі впливають на зміщення результатів рекомендаційної системи і зниження її ефективності. Аналіз інтернет-ресурсів показав, що більшість рекомендаційних систем не надає користувачам рекомендацій, а та частина, яка це робить, наприклад пропонує користувачеві продукти, здійснює підбір рекомендацій вручну. Отже задача розробки методів автоматизованого створення рекомендацій за обмеженим набором вхідних даних є досить актуальною. Проблеми роботи (розрідженість даних, проблема нового користувача, масштабованість) широко використовуваного алгоритму SVD для розробки таких рекомендаційних систем пропонується усунути шляхом удосконалення даного алгоритму методом найближчих k-сусідів. Даний метод дозволить легко сегментувати і кластеризувати дані системи, що зекономить ресурси системи.

Ключові слова: рекомендаційна система; алгоритм SVD; метод k-найближчих сусідів; розрідженість даних; масштабованість; кластеризація.

Усовершенствование алгоритма SVD для повышения эффективности рекомендательных систем

С. Я. Крепыч, И. Я. Спивак

Аннотация. Большое количество на сегодня существующих веб-сайтов используют рекомендательные системы для своих пользователей. Они генерируют им различные предложения, например, подобные товары или рекомендуют людей, зарегистрированных на этом сайте, по похожим интересам. Такие рекомендательные механизмы обрабатывают огромные объемы информации для обозначения потенциальных преимуществ пользователей. Рекомендательные системы - это программы, которые пытаются определить, что хотят найти пользователи, что может их заинтересовать и рекомендуют им это. Эти механизмы усовершенствовали взаимодействие между пользователем и сайтом. Взамен статической информации они предоставляют динамическую информацию, которая меняется: рекомендации генерируются отдельно для каждого пользователя, основываясь на его предыдущей активности на данном веб-ресурсе. Также может учитываться информация, поступающая от других посетителей. Методы сбора информации, предоставляемых интернетом, значительно упростили использование человеческой мысли с помощью коллаборативной фильтрации. Но, с другой стороны, большой объем информации затрудняет воплощение этой возможности. Например, поведение одних людей достаточно ясно подвергается моделированию, в то время как другие ведут себя абсолютно непредсказуемо. И именно вторые влияют на смещение результатов рекомендательной системы и снижение ее эффективности. Анализ интернет-ресурсов показал, что большинство рекомендательных систем не предоставляет пользователям рекомендаций, а та часть, которая это делает, например, предлагает пользователю продукты, осуществляет подбор рекомендаций вручную. Итак, задача разработки методов автоматизированного создания рекомендаций по ограниченным наборам входных данных является весьма актуальной. Проблемы работы (разреженность данных, проблема нового пользователя, масштабируемость) широко используемого алгоритма SVD для разработки таких рекомендательных систем предлагается устранить путем усовершенствования данного алгоритма методом ближайших k-соседей. Данный метод позволит легко сегментировать и кластеризовать данные системы, что сэкономит ресурсы системы.

Ключевые слова: рекомендательная система; алгоритм SVD; метод k-ближайших соседей; разреженность данных; масштабированость; кластеризация.