Проведено дослідження рекоменд ційних лгоритмів т виявлення перев г т недоліків. Вдоскон лено метод формув ння рекоменд цій н основі кол бор тивної фільтр ції як Content-Based Filtering (CBF), Collaborative Filtering (CF) т гібридних методів Machine Learning (ML). Опис но принципи проектув ння т функціон льні вимоги до рекоменд ційної системи у вигляді веб-дод тк для вибору необхідного для користув ч контенту н прикл ді фільмів. Основні дослідження зосереджені н вирішенні проблем холодного ст рту т м сит бов ності в методі кол бор тивної фільтр ції. Для ефективного вирішення цих проблем використ ні гібридні методи н вч ння. Здійснен пр ктичн ре ліз ція гібридної рекоменд ційної системи (ГРС) н д ння релев нтних рекоменд цій контенту н прикл ді фільмів з ур хув нням особистих потреб користув ч н основі розробленого гібридного методу. Удоскон лено лгоритм формув ння рекоменд цій контенту н основі кол бор тивної фільтр ції т Machine Learning для спільної фільтр ції пок зників подібності між користув ч ми бо між тов р ми. Гібридний лгоритм прийм є вхідну інформ цію у різному вигляді, норм лізує її т формує відповідні рекоменд ції н основі комбін ції методів CF т CBF. Machine Learning зд тне визн ч ти чинники, що вплив ють н підбір релев нтних фільмів, що сприяє поліпшенню н д ння рекоменд цій конкретному користув чу. Для вирішення цих з вд нь пропонується новий удоскон лений метод, н відміну від н явних систем рекоменд цій, в основі якого леж ть гібридні методи т Machine Learning. Д ні для Machine Learning розробленої ГРС взято із MovieLens. Про н лізов но методи формув ння рекоменд цій користув чеві, проведений огляд н явних рекоменд ційних систем. Експеримент льні результ ти пок зують, що пок зники роботи з пропонов ної ГРС н основі технології CF+CBF+ML кр щі, ніж у двох одиничних моделей, CF т CBF, т їх комбін ції як CF+CBF, CF+ML т CBF+ML. ГРС рекомендується використовув ти для збору д них про вподоб ння людей у виборі тов ру т н д ння релев нтних рекоменд цій

Ключові слов : комерційний контент, персон ліз ція, Machine Learning, SEO-технологія, метрики пошуку, електронн комерція, NLP

UDC 004.89
DOI: 10.15587/1729-4061.2019.175507

DESIGN OF A RECOMMENDATION SYSTEM BASED ON COLLABORATIVE FILTERING AND MACHINE LEARNING CONSIDERING PERSONAL NEEDS OF THE USER

V. Lytvyn
Doctor of Technical Sciences, Professor*
E-mail: vasyl.v.lytvyn@lpnu.ua
V. Vysotska
PhD, Associate Professor*
V. Shatskykh
Programmer
ANAT Company
Chervona Kalyna ave., 104, Lviv, Ukraine, 79049
I. Kohut
PhD, Associate Professor
Department of Computational Mathematics and Programming**
O. Petruchenko
PhD
Department of Engineering Mechanics
(Weapons and Equipment of Military Engineering Forces)
Hetman Petro Sahaidachnyi National Army Academy
Heroiv Maidanu str., 32, Lviv, Ukraine, 79026
L. Dzyubyk
PhD
Department of Technical Mechanics and Dynamics of Machines**
V. Bobrivetc
Lecturer
Department of Economic Expertise and Audit for Business***
V. Panasyuk
PhD, Associate Professor
Department of Accounting and Taxation of Entrepreneurship***
S. Sachenko
PhD
Department of Economic Assessment and Business Audit***
M. Komar
PhD
Department of Information and Computing Systems and Control***
*Department of Information Systems and Networks **
**Lviv Polytechnic National University
S. Bandery str., 12, Lviv, Ukraine, 79013
***Ternopil National Economic University
Lvivska str., 11, Ternopil, Ukraine, 46009

Received date 07.07.2019
Accepted date 31.07.2019
Published date 23.08.2019

1. Introduction

The amount of information around the world is growing at a rapid pace over a significant period of time [1]. Every day, people receive and filter the incoming flow of information arriving from various sources: work, social networking, email, online movie theaters, popular sources of information, etc. [2]. The advent of the Internet gave rise to a sharp

increase in the amount of such information; a large number of services have emerged aiming to provide users with everything required for a comfortable life [3]. In recent years, significant popularity has been enjoyed by the Internet services that offer products of all possible kinds (Internet shopping), information about anything (online magazines, movies, news, forums, books, serials, articles), etc. [4]. For the user, it has become extremely difficult to navigate the catalogs of movies and lists of serials, even equipped with built-in search and filtering, since it is very difficult to make a choice given such a large amount of information [5]. The global Internet network makes it possible to conveniently collect information about preferences by users of on-line resources [6]. The global Internet network has a huge number of search engines, catalogs, ratings, file-sharing services, etc. Moreover, this volume is constantly growing. The amount of content increases exponentially because each web-site/service in the process of its functioning generates even greater volumes of content [7]. The main, but not the only, task for recommendation services is to help the content user navigate search results. Historically, people have relied on their peers or experts' advice to support decision-making and obtain recommendations on products, news, entertainment, etc. [8]. Exponential growth of digital information over the past 25 years, especially in the Internet, has created a problem of information overload. This term is defined as "stress induced by receiving more information than necessary for making a decision, and by attempts to deal with it using outdated methods of time management". This issue limits the ability to view specifications and choose among many alternatives in the online market. On the other hand, information science and technology responded in an appropriate way by developing tools for filtering information in order to solve this problem [9]. Recommendation systems (RS) are such tools. RS emerged in the mid-1990s [10]. They are typically defined as software tools and methods that are used to provide suggestions to users and other stakeholders [1].

Recommendation systems are the software tools and methods that provide suggestions about items that are most likely to represent interest for a specific user [11]. The proposals relate to different processes of decision-making, such as what to buy, what music to listen to, or which movies or tv series to see [1].

Typically, recommendations about movies from different online services at existing reference systems are not necessarily relevant to users as they are often based on unproductive and obsolete algorithms [12]. There are many problems when constructing a forecasting system for recommending movies, so it is very interesting to see how different approaches to building reference systems address these problems and improve quality of recommendations [13]. The subject is relevant, because the Internet has not enough proper services to provide relevant recommendations about movies [14]. They are all closed commercial projects. Therefore, developers of similar projects need to re-invent the structure and architecture for their own recommendation systems. In addition, they must run their own analyses to verify the selection or improvement or development of new better methods and technologies to devise recommendations to end user.

Personalized approach to a Web-resource user greatly increases sales turnover. That is, those clients that cannot find the information required leave the Web-resource unsatisfied; they would probably never visit it again, given the extremely competitive Internet market. According to [2], up to 40 % of visitors typically use the search feature of a Web-resource, thereby demonstrating an intention to buy a product based on title or code. Therefore, obtaining the required personalized search results and their further analysis by means of artificial intelligence based on the collaborative filtering and Machine Learning is essential for the successful development of e-business. E-commerce turns to Machine Learning to improve recommendations for consumers as regular/potential visitors of a Web-resource. The Machine Learning methods significantly improve analysis of the results from a previous personalized search by a user after visiting a Web-resource. Generating a search topic rating makes it possible for a Web-resource to sort out results based on relevance or evaluative relevance. Such an estimate takes into consideration specific search terms, as well as the specific features in the respective user's profile (such as age, gender, previous orders, preferences, and previous search terms). Personalization algorithms make it possible to associate each user with the most probable list of goods/services according to the needs and interests. These algorithms also make it possible to predict customer wishes, even if s/he is not aware of them. In addition to the usual text entry of a categories and the tags based on images and the description of goods/services, it has become a more viable option to add automation processes and decision-making systems to identify wishes of a Web-resource user. Scientific advances concerning context recognition using Machine Learning networks now provide the technology for automated tagging the descriptions of goods at a Web-site for e-commerce.

2. Literature review and problem statement

Recommendation systems play an important role for popular Web sites such as YouTube, Netflix, Last.fm, Spotify, Facebook, LinkedIn, TripAdvisor, and IMDb [15]. Most media companies develop and deploy RS as part of the services provided to subscribers [16]. For example, Netflix, an online provider of streaming media on request, awarded a prize worth USD one million to the team, which, for the first time, managed to significantly improve the performance of their system of recommendations [1]. Paper [17] reports results of research into why providers might be interested in using this technology. It was shown that there are the following factors:

– increasing the number of sold goods;
– selling a larger variety of articles;
– improving user satisfaction;
– enhancing loyalty of users;
– a better understanding of what the user wants.

The first factor that one should consider when constructing RS is the area of application because it has a significant impact on the algorithmic approach that needs to be adopted [18]. There are resources that provide taxonomy on RS and categorize existing RS into specific domains of applications [1]. Based on these specific domains, one can define a more general domain classes for the most common systems in use [7]:

– entertainment – recommendations on movies, music, games, and IPTV [19];

– content – personalized newspapers, recommendations on documents, Web pages, software for e-learning, and e-mail filters [20];

– e-commerce – recommendations for buyers (books, cameras, PCs, etc.) [21];

– services – recommendations on travel services, experts for advice, houses to rent or dating services [22];

– social – recommendations of people and content in social networks such as tweets, Facebook feeds, updates for LinkedIn, etc. [23].

However, there remain the unresolved issues related to the choice of a particular algorithmic approach for a particular area of RS application [24]. The reason for this could include the objective and subjective difficulties related to:

– the lack of research and experiments regarding similar projects in these domains for a wide range of users (most projects are private, one-time, closed, commercial, whose results are not subject to publicity) [25];

– the fundamental impossibility to access both the architecture of such closed commercial projects and the methodology of development, analysis results, operation, etc. [26];

– high cost in terms of conducting own experiments involving both the third-party projects and creating several own multiple projects, which renders the respective research impractical, etc. [27].

An option to overcome such difficulties could only to try to conduct own experiments involving own developed projects [28]. However, there is one significant disadvantage – the lack of a comparison among the results from analogs [29]. Therefore, it is a relevant task to undertake a study to analyze product ratings offered by e-commerce in order to develop recommendations to end user based on personal needs and preferences.

Fig. 1 shows a general architecture of RS. In fact, ratings are the most popular type of data about transactions that RS collects [30]. These ratings can be explicitly or implicitly acquired [31]. In the explicit set of estimates, a user is asked to give an opinion about a subject based on the rating scale [32]. Another form of user's assessment consists of tags associated by the user with the elements that the system provides [33]. For example, at MovieLens (https://movielens.org/) the RS tags show the way the MovieLens users describe a movie, for instance: "drag" or "fantastic" [2]. Technically, RS employ multiple recommendation strategies based on editing algorithms such as Content-Based Filtering (CBF), Collaborative Filtering (CF), Demographic Filtering (DF), Knowledge-Based Filtering (KBF) [34]. There is also a class of approaches based on a combination of different methods – Hybrid Filtering (HF) [35].

Common steps for developing recommendations include (Fig. 2) [3]:

1) preliminary processing, which includes the search for similarities in data, defining a sample, data dimensionality, [36];

2) data analysis, which includes data categorization and clustering [37];

3) representation of results [38].

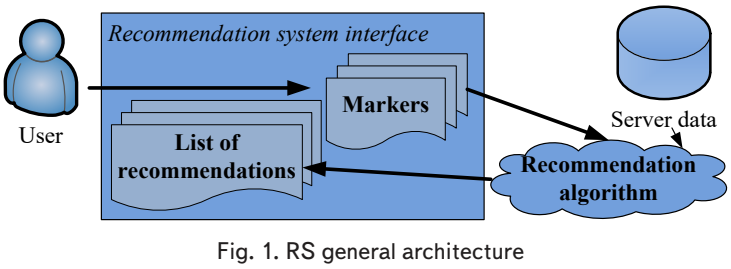


Fig. 1. RS general architecture

Today, RS are found everywhere, helping users search for different kinds of goods/services [39]. They are also sales assistants for enterprises, increasing their profits [40]. To figure out how to describe methods for providing recommendations, one must answer a series of questions [41]:

– what are the challenges faced by researchers in this field?

– which methods of data mining and Machine Learning are used in hybrid RS (HRS)?

– how are these methods combined, and which problems do they solve?

– which procedures and metrics are used for assessment?

– which RS characteristics are estimated and which metrics are used?

– which data sets are employed to train and test HRS?

– which areas are the most promising for future research?

The problems related to methods for developing recommendations within RS include [42]:

– Cold start is relevant to recommendations for new users or goods in the absence of the corresponding statistical information about them (a new user's preferences and ratings for a new product) [43]. Often used in this case is the probabilistic model to extract hidden attributes from elements [44]. By employing hidden functions, they generate accurate pseudo-ratings, even under a situation of cold start when there are few estimates, or not at all [45]. Another example of solving it is the combination of CF and CBF [46]. The issue related to the cold start is also resolved by using a mathematical technique for combining weighted outputs from different strategies on recommendations applying the Ordered Weighted Averaging (OWA) method [47].

– Data sparsity generally occurs when users evaluate a limited number of available elements, especially when a catalogue is long [48]. The result is a sparse matrix of users' rating with insufficient data to identify such users or goods, which adversely affects quality of recommendations [49]. Data sparsity prevails in CF RS that rely on a feedback from peers for giving recommendations [50]. Data sparsity in cross-domain recommendations is often tackled by using a factorization model of the triad relation user-item-domain [51]. In addition, they consider each rating by a user's element as the predicate of other missing ratings [52].

– Recommendation accuracy is the ability of RS to relevantly predict the advantages of an element for a specific user [53]. Increasing the accuracy of recommendations is always paid great attention to [54]. This is especially important for situations involving data sparsity [55].

– Scalability is a difficult-to-access characteristic that is associated with the number of users and products within RS [56]. When developing recommendations for several items for several hundred users, the system would probably not be able to offer hundreds of products to millions of people, if it was not designed for high scalability [57]. Developers of the system Hyred, in order to eliminate such a problem, combined a modified Pearson correlation for CF with the boundary distance CBF [58]. That is, they find the nearest and far neighbors of each user to reduce the data set [59]. Using a compressed data set improves scalability, reduces sparsity and the computation cost for the system [60].

– Diversity is a desirable characteristic, which has recently attracted particular attention [8]. It is important to produce a variety of recommendations, since it helps avoid a bias in recommendations [61]. This can be a list of

recommendations with similar elements (for example, all episodes from a very popular tv saga) [62]. The user who is not interested in one of them is probably not interested in both; thus, he gains no benefit using a recommendation [63].

– Other problems include the lack of personalization [64], confidentiality issues [65], noise reduction [66], the integration of data sources [67], lack of novelty [68] and the adaptability of user advantages [69].

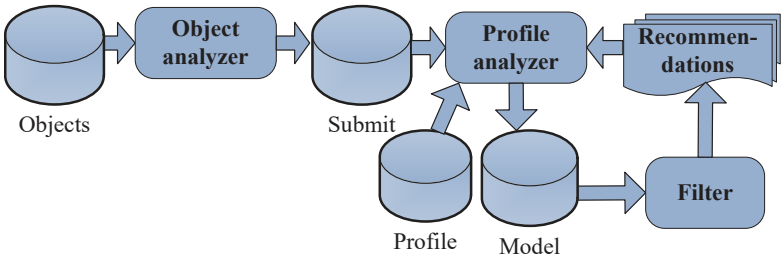


Fig. 2. Recommendation development path within CBF

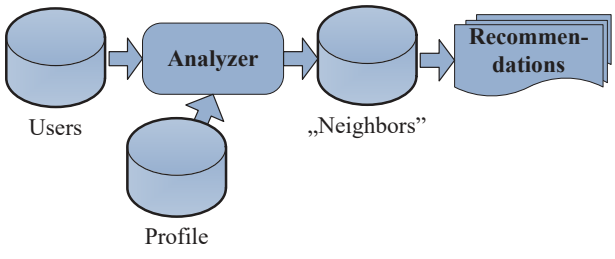


Fig. 3. Recommendation development path within CF

Solving at least part of the issues related to designing domain RS would greatly improve quality and efficiency of servicing the end user of e-commerce in today's world of advanced Internet technologies. However, the main problem is to choose the best algorithm to develop recommendations for domain RS of specific e-business. We shall analyze common and popular recommendation algorithms such as CBF, CF, KBF, DF, and HF [70].

CBF is based on the following assumption: once a product with certain attributes was adopted in the past, then such product could be attractive in the future [71]. It uses the elements' functions to compare an element to users' profiles and to provide recommendations (Fig. 2) [72]. The quality of recommendations is limited by the selected functions of the recommended elements. CBF suffers from the problem of a cold start (a new user or a new element), a "gray pencil" (users who do not fit any cluster of favorites) [73].

The basic CF assumption: similar preferences of people in the past are likely to be the same in the future [74]. One of the definitions is "cooperation between people to help each other perform filtering, by recording their responses to documents they read" [3]. It uses ratings or other forms of feedback generated by the user, to identify shared preferences between groups of users (Fig. 3). Then it generates recommendations based on similarities among users [2]. Similar to CBF, CF suffers from the problems of a cold start [75]. A vivid example of CBF is demonstrated by a Venn diagram for the system's users (Fig. 4), which shows links between the "sameness" and "difference" between two users of a single system. As one can see, the "differences" are an excellent source to generate recommendations [6]. The CBF algorithms are divided into such categories as Memory-based, Model-based, and Hybrid (Fig. 5) [76].

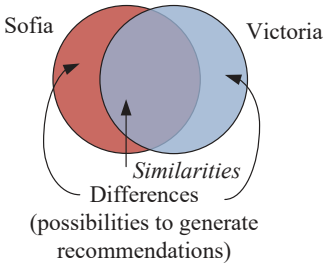


Fig. 4. CBF Venn diagram

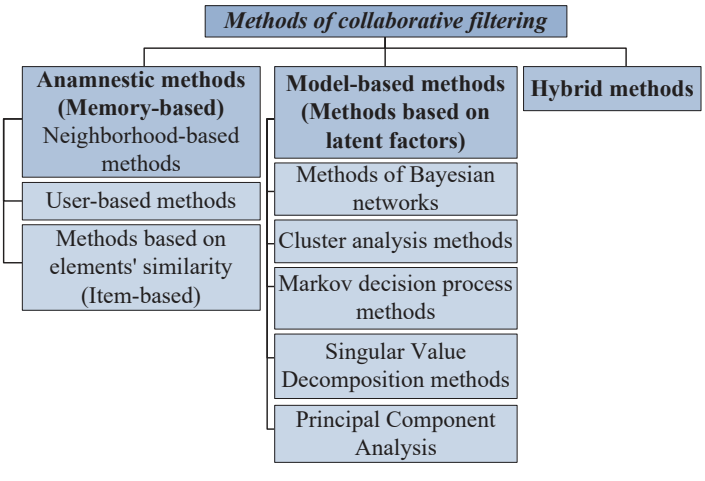


Fig. 5. CBF methods

KBF applies knowledge about users and goods to substantiate which goods meet the requirements of users, and generate appropriate recommendations [4]. A special type of KBF is the RS based on the constraints [72]. They can recommend technically-sophisticated products that are rarely purchased (for example, cars, houses) and detect important limitations for the user (for example, price) [5]. One cannot successfully use CF or CBF for these goods because there is a limited data on the interaction among a system's users (people rarely purchase houses) [73].

DF apply demographic data such as age, gender, education, etc. for defining the categories of users [74]. It does not suffer from the problem of a cold start because it does not use ratings [75]. However, it is difficult to collect sufficient amount of the required demographic information given the Internet privacy problems that limits the application of DF. It is combined with other methods to improve the accuracy of recommendations [76].

At a combination of CBF and CF, HF can be more effective in some cases [77]. HF is implemented in several ways: through separately defined forecasts based on content as well as shared. They are then combined by adding features based on content to the approach that is based on shared work (and vice versa) [78]. A few studies empirically compare performance of HF with pure methods of CBF and CF and demonstrate that HF can provide more accurate recommendations than pure approaches [79]. These methods can also be used to overcome some of the common problems related to RS such as a cold start and the issue on data sparsity. Netflix is a good example of using HRS [80]. The web-site provides advice by comparing the habits of viewing

and finding by such users (for example, shared filtering), as well as by offering movies that share characteristics with the movies highly appreciated by the user (content-based) [81].

Each of CBF, CF, KBF, DF methods has weaknesses [82]. That is why the application of HF solves the problems arising when using each of the methods CBF, CF, KBF, and separately DF. HRS is such that combines several methods to achieve certain synergies among them [84]. The term HRS is used here to describe any RS, which combines several methods of recommendations for a more effective and efficient result [85]. There is no reason why several different methods of the same type cannot be hybridized, for example, two different recommendations based on CBF and CF can work together [86]. There are several methods of hybridization [87]:

– weigh: numerical evaluation of various components of recommendations;

– switch: selecting one option among the components of recommendations;

– mixed: the simultaneous use of recommendations by various recommenders;

– combination of attributes: functions that obtained from various sources of knowledge are merged and represented in a unified algorithm of recommendation;

– increased number of functions: one recommendation procedure is used to compute the set of functions as part of the input data for the following technique;

– cascade: recommendations are assigned with a clear priority, in this case, recommendations with lower priority affect recommendations with higher priority;

– meta-level: one recommendation procedure is used; a model is generated as input information for the following technique.

And only a practical research for a specific domain HRS would make it possible to identify the best variant to develop relevant recommendations to the end user according to the personal needs and preferences.

3. The aim and objectives of the study

The aim of this study is to design a decision-making system to develop recommendations on content based on the collaborative filtering and Machine Learning taking into consideration the user's personal needs.

To accomplish the aim, the following tasks have been set:

– to devise general functional requirements to the architecture of a system to develop recommendations on content for distributing commercial content in the Internet space;

– to construct a method for the personalization of commercial content according to the user's needs;

– to develop software for distributing commercial content in the Internet space based on collaborative filtering and Machine Learning;

– to analyze results from experimental testing of the proposed method for the personalization of commercial content according to the user's needs.

4. General functional requirements to the architecture of a system to develop recommendations on content to distribute content

Aspiring to accomplish the aim, we shall describe the general purpose, aspects of the general purpose and sub-aspects of the general purpose of the systems, as well as criteria. To this end, we shall build the DSS objectives tree (Fig. 6).

Forming the most relevant recommendations to a user is the third aspect of the system's general purpose. We shall highlight a separate sub-aspect within it, the aim of which is to choose, among all relevant recommendations, 10 recommendations with the highest rating. To implement the system, we shall apply MAI (a method of analytical hierarchy) with 4 alternative variants (Tables 1–3):

– v_1 – combined type;

– v_2 – data-oriented;

– v_3 – knowledge-oriented;

– v_4 – model-oriented.

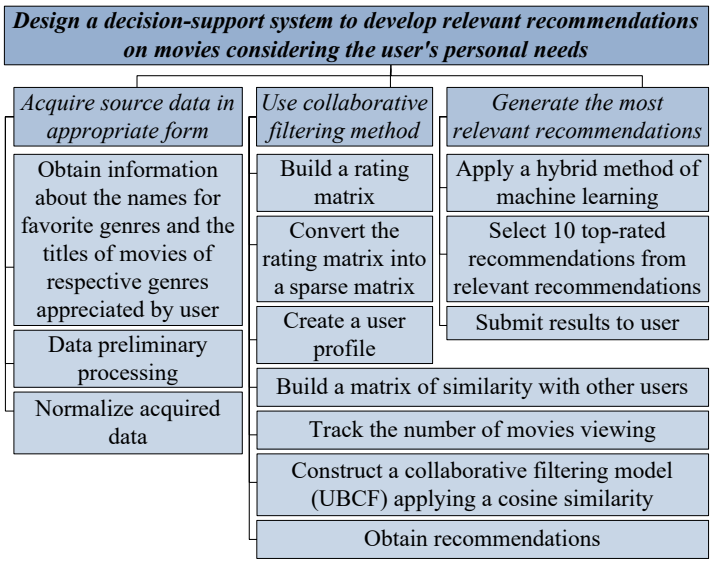


Fig. 6. Objectives tree to provide relevant recommendations

Table 1

Filled matrices of pairwise comparisons based on analysis of papers [1–13]

	Criterion 1 Recommendation accuracy				Criterion 2 Cold start elimination				Criterion 3 Scalability			
	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4	v_1	v_2	v_3	v_4
v_1	1	1	6	3	1	6	2	6	1	2	3	3
v_2	1	1	8	5	1/6	1	4	2	1/2	1	1	5
v_3	1/6	1/8	1	2	1/2	1/4	1	4	1/3	1	1	6
v_4	1/3	1/5	1/2	1	1/6	1/2	1/4	1	1/3	1/5	1/6	1
	Criterion 4 Data sparsity				Criterion 5 Increasing sales				Criterion 6 Diversity			
v_1	1	4	6	2	1	3	6	1	1	6	3	6
v_2	1/4	1	6	4	1/3	1	7	3	1/6	1	2	3
v_3	1/6	1/6	1	2	1/6	1/7	1	3	1/3	1/2	1	7
v_4	1/2	1/4	1/2	1	1	1/3	1/3	1	1/6	1/3	1/7	1
	Criterion 7 Recommendation relevance				Criterion 8 Increasing the number of regular users				Criterion 9 Noise elimination			
	1	3	7	3	1	2	3	2	1	2	6	2
v_2	1/3	1	2	3	1/2	1	9	1	1/2	1	6	3
v_3	1/7	1/2	1	6	1/3	1/9	1	6	1/6	1/6	1	2
v_4	1/3	1/3	1/6	1	1/2	1	1/6	1	1/2	1/3	1/2	1

Results from expert assessment based on analysis of papers [1–13] followed by the application of MAI allow us to draw a conclusion on the feasibility of choosing a variant for constructing a RC of the combined type (Table 3). As an example, let us take the HRS domain – entertainment, the subdomain – movies (Fig. 7). A user is involved in the processes of providing preferences for the genres of movies and favorite movies. The user must obtain relevant recommendations, as well as be able to view all available recommendations. A separate variant to use is the possibility to add a recommended movie to favorites.

The activity diagram demonstrates (Fig. 8) that there are three processes: "Provide preferences of movies' genres", "Update Preferences", and "View all recommendations". The system would also be able to perform other atomic activities such as deleting irrelevant advice, removing and adding users to the system, etc.

Table 2

Comparing the criteria for qualitative aspects based on analysis of papers [1–13]

Factor 1 Increasing user satisfaction				Factor 2 Adaptability of user advantages				Factor 3 Understanding the user's needs			
1	3	6	1	1	6	2	6	1	4	6	2
1/3	1	7	3	1/6	1	4	2	1/4	1	6	4
1/6	1/7	1	3	1/2	1/4	1	4	1/6	1/6	1	2
1	1/3	1/3	1	1/6	1/2	1/4	1	1/2	1/4	1/2	1

In the sequence diagram (Fig. 9), 9 messages interact with 3 objects: user, a user management service that generates relevant recommendations, and a data base of movies and ratings.

Table 3

Vector of priorities, calculated based on the method of analytical hierarchy

Alternatives	v_1	v_2	v_3	v_4	Weight values	Selection variants
v_1	1	2	6	2	0.42596	1
v_2	1/2	1	6	3	0.256486	2
v_3	1/6	1/6	1	2	0.252045	3
v_4	1/2	1/3	1/2	1	0.065509	4

It follows from the state transition diagram (Fig. 10) that in case relevant recommendations have already been developed the system immediately passes to the end-state. If recommendations are still to be generated, the system passes 4 stages: preliminary data processing, collaborative filtering, a hybrid method for generating recommendations, and demonstrating results to the user. It is the stage of the hybrid method application that provides, at the output form the system, the relevant recommendations of movies to the user, since the application of collaborative filtering only does not produce the desired result.

A class diagram (Fig. 11) consists of 6 entities such as collaborative filtering InvertedKNN, User, RecommenderSystem that provides recommendations, ratings and estimates MovieRating, database MovieDB, and a Wishlist. A deployment diagram for the developed system is shown in Fig. 12. It is evident that the following hardware is required: a user's browser, a server to manage users, a server for generating relevant recommendations, and a database server.

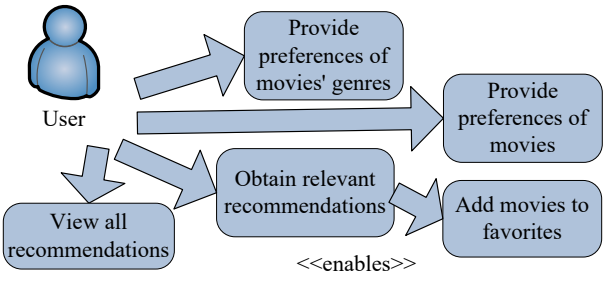


Fig. 7. Diagram of variants for using the combined type of RS

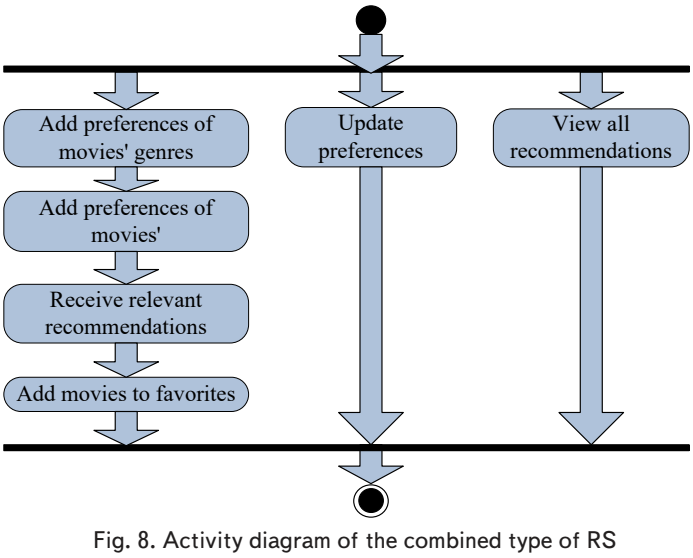


Fig. 8. Activity diagram of the combined type of RS

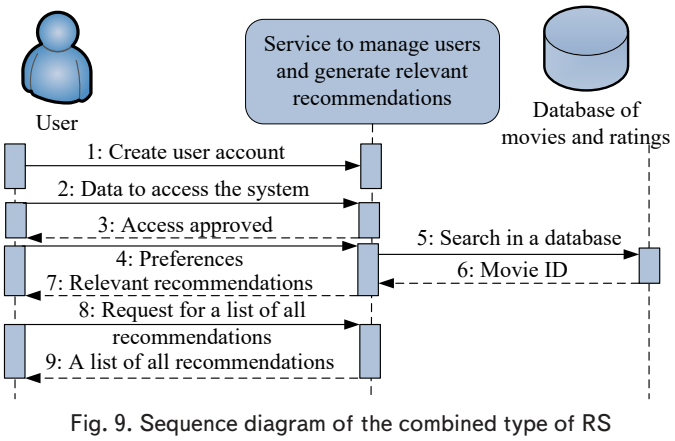


Fig. 9. Sequence diagram of the combined type of RS

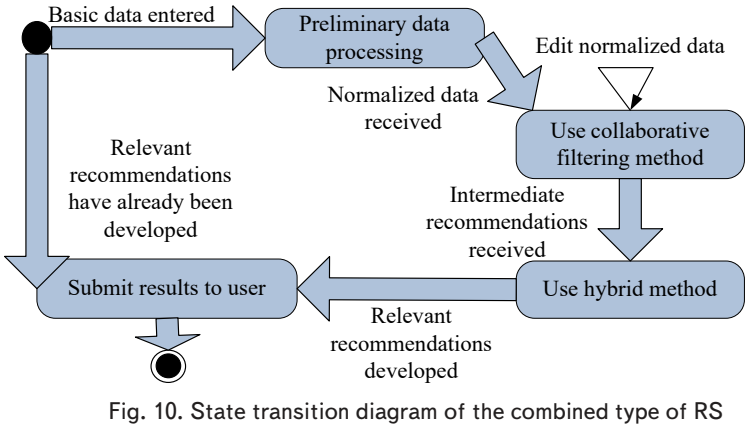


Fig. 10. State transition diagram of the combined type of RS

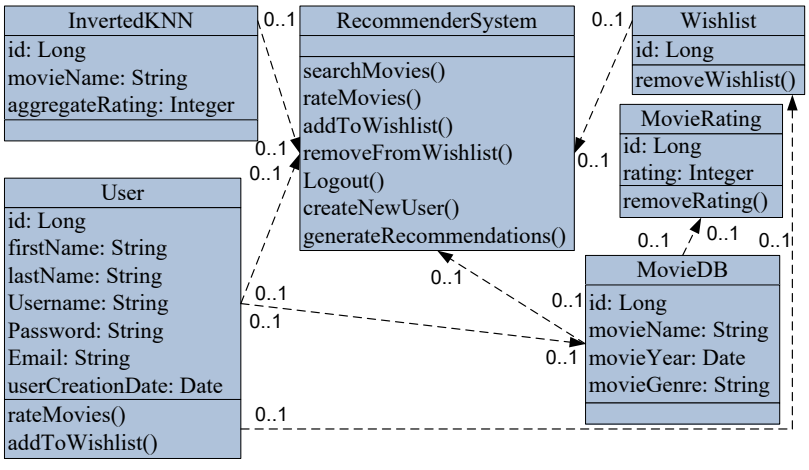


Fig. 11. Class diagram of the combined type of RC

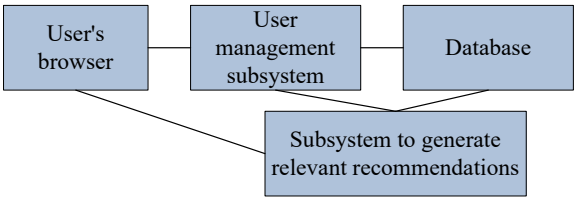


Fig. 12. Deployment diagram of the combined type of RS

For reasons of system security, it is not recommended using a single server for processing users' accounts and for generating recommendations. Since these are different jobs for the system, separate servers are required.

5. Content personalization method based on the user's needs

Existing systems of recommendations of movies, such as IMDb, Netflix, and Rotten Tomatoes, are effective in individual tracking of personal needs of users. These systems help select movies, providing effective recommendations that take into consideration the needs of a specific user, including the frequency of watching movies, a user's preferences in genres, actors, and time intervals. To ensure that the system can support decision-making to provide users with relevant recommendations of movies, it is necessary to combine existing recommendation algorithms based on personal needs and to build the hybrid ones, which is the purpose of the system design. As for the two types of problems on a cold start, the new user's issue is considered more challenging [10]. Accordingly, some studies suggested hybrid methods, which basically combine filtering based on content and the collaborative filtering for stricter restrictions by using algorithms [9]. Given the growth in the volume of information and data, there are various problems within the methods of filtering, including density and scalability. The newest trends in Machine Learning employ different processing layers, which helps training using complex contextual features. Machine Learning can determine those factors that influence the selection of relevant movies, which improves providing recommendations to a specific user [10]. To solve these tasks, a new improved method of learning has been proposed underlying which, in contrast to existing systems of recommendations, are the hybrid methods while it is based on Machine Learning. The main purpose of the system is to provide relevant recommendations on movies considering the personal needs of the user. The designed RS is recommended to use to collect data on the preferences of different people in the choice of movies and to provide relevant daily recommendations. In addition, the system can be easily adapted for recommending music, books, cryptocurrency, or even exchange markets. A hybrid algorithm could receive source information in different forms, normalize it, and provide appropriate recommendations. Given this, the system can be used both in everyday life for the selection of movies and in large corporations in workflows. The designed system is the RS of movies that is based on the hybrid algorithms for Machine Learning, including the user-based collaborative filtering algorithm. A Web application is intended to provide recommendations on the selection of movies. This is a cross-browser and multi-platform Web application. One can use it from any device with a Web browser and access to the Internet. NoSQL [11] is used to store data that helps integrate contextual information into two subsystems – a user profile subsystem and a content subsystem. Each technique focuses on different aspects using the methods of Machine Learning. Hybrid training methods are used for a personalized recommender and are based on feedback. There are recommendations that can be represented as an assessment of the responses from a new user, the choice of specific algorithms, a learning mechanism, and the evolution of performance [12]. The content of RS supports the recording of routine activities and provides services for recommendations. Processing the requirements to data is performed by transmitting the data sources in many aspects. There is an integration of data related to the user's context and recommendations means. Data processing is based on Machine Learning algorithms employing a trainer, which study reflexive functions, which, in turn, helps in predicting the relevance of recommendations for each user. The problem of a cold start occurs when the user's account does not exist within the RS and there is no any previous rating [9]. A conceptual model of the system is shown in Fig. 13.

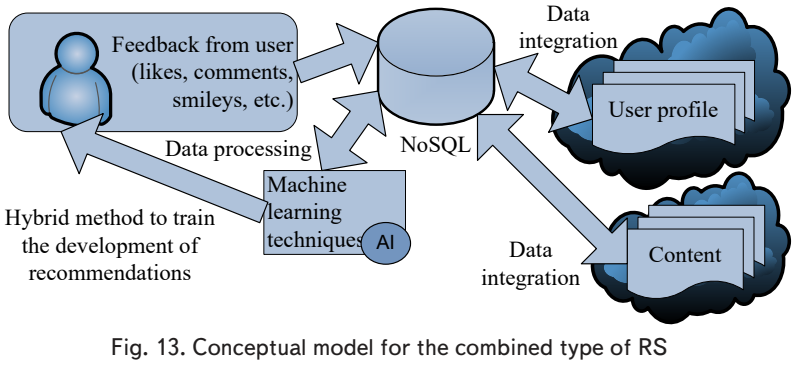


Fig. 13. Conceptual model for the combined type of RS

To address these problems, we propose a new training method, based on Machine Learning, for improved recommendations, based on the hybrid training methods to improve existing systems of recommendations using common filtering. Most research focuses on solving the problems on a cold start and scalability in the method of collaborative filtering. To effectively address these problems, we exam-

ined the hybrid training methods. HRS is designed to study the way the combined filtering improves solving important problems of a cold start, using the concept of the learning profile per cycle. The type of users' feedback improves the accuracy of recommendations and enhances efficiency of the system operation based on the methods of Machine Learning. As regards meeting the requirements, the methods of hybrid training integrate different algorithms, including:

– switch between filtering based on content and collaborative filtering in order to solve the problem on cold-starting the new users;

– detection of user's context within the dynamic filtering;

– integration of training methods based on the profile to display users' reviews.

The most important part of HRS is the users' profiles and the relevant content. Fig. 14 shows an HRS model with the identified components from Fig. 12 [32]:

– a user's browser – the component "User's experience and environment";

– a subsystem of user management – the component "Engineering of user's profile";

– a database – the component "Data storage".

– a subsystem for generating relevant recommendations – the component "Hybrid method to train recommending".

The flows of HRS service data are strictly regulated by the main stages in the system operation, steps 1–12. Preliminary processing of the user's data stream is performed at stages 1 and 3, respectively (substeps *A*–*D*). Data exchange between the main components of HRS is described and denoted by # [32].

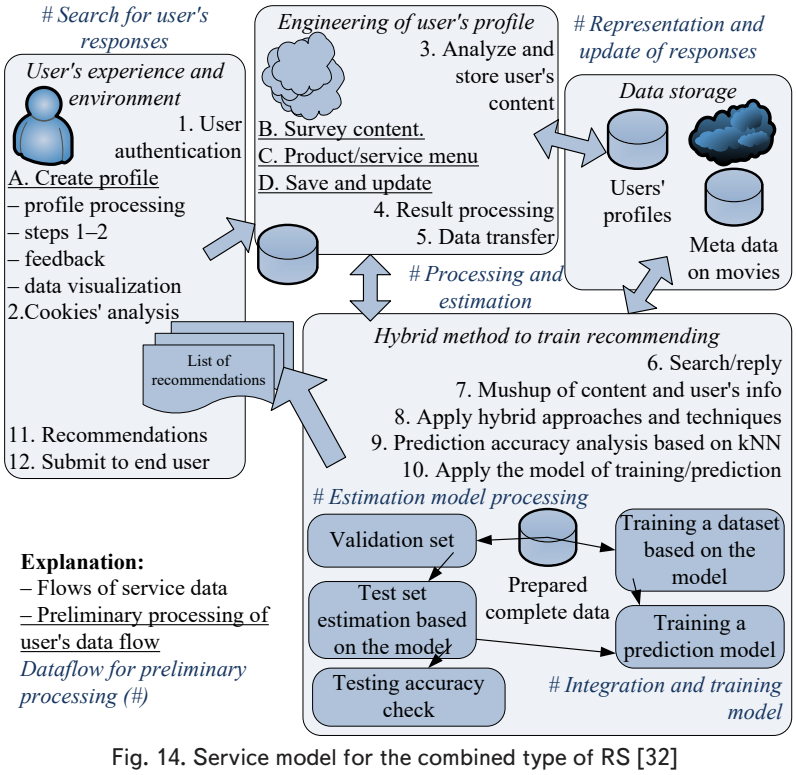


Fig. 14. Service model for the combined type of RS [32]

At the first stage, users respond to a simple survey to define their preferences and expectations for a movie genre. This information is saved in the history of the user profile to filter and develop an initial set of recommendations. Users use an adaptive visual interface to express personal preferences for movies based on the analysis of the browser cookies. The preferences defined are used to further re-evaluate submitted recommendations. The first subsystem of the system's model is the collection of data that includes the aggregate information, which is described at steps 1 and 2 with creating a profile at registration <A>. At the second stage, based on the processing of a flow of service data at steps <3–5> [32], the information is transmitted and stored as a preliminary processed data stream from the user, specified in the <B–D> algorithm for the user's profile. At the third stage – there is the application of hybrid training techniques at steps <6–10> necessary for the preliminary processing of the data flow of content, referred to in <#>, using Machine Learning (ML) for training and testing a dataset and assessing the learning model. This is needed to improve the accuracy of hybrid filtering and to save the results from filtering based on *k*-nearest neighbors for the classification algorithm. To construct a user's favorites list, we use the metadata on movies at steps <11–12> [32].

The proposed system's architecture is shown in Fig. 15. Training methods in the output functions are based on the algorithm of *k*-nearest neighbors for processing preferences in users' ratings and for ranking the relevant content [13]. At the next stage, the system provides users with relevant recommendations.

Fig. 16 shows the process of HRS functioning using the technique of mixed ML [32]. A user must first create a profile. Next, the user is offered a list of movies based on his estimates. At the next stage, the system creates a model of training data acquiring source data from observations. Further, this process includes the filtering of data and the selection of elements, preferred or assessed by the user. This same process is repeated to create a similar set of service content. This represents mixed ML, so the results of both iterations are merged. The process of hybrid ML continues to switch from one sample to another, to solve the problem on cold starting a new user. As presented in the diagram, the managed ML is applied. The source data for software are a database of movies and ratings. In order to develop a user-friendly Web application interface, source data were simplified to represent a choice among three genres and three movies in line with respective genres. Source data are a list of the ten movies most relevant to a user.

The born HRS resolves the classes of tasks on forecasting recommendations on movies, selecting those movies that would be relevant for each user based on his preferences.

The programming system employs the following methods to solve the tasks:

– User-based collaborative filtering.
– Element-based collaborative filtering.
– Intelligent data mining.
– Machine Learning.
– Hybrid methods.

The software offers relevant recommendations for movies based on the choice of three favorite genres and three favorite movies. By using this system, a user would much faster find new movies to watch. The system is at the cloud servers and works 24 hours. The administrative panel of the service Shiny records permanent logs of the Web application operation, which one can view at any

time. If the Web application devours too much traffic, this can cause it to slow, then the service reports this issue to the developer or administrator by using e-mail. For the stability of using the Web application, it is recommended to employ high-quality and fast Internet connection, such as Wi-Fi or 4G, and a device with sufficient RAM. The application could be enabled from any device with a Web browser and access to the Internet. Peripheral devices are not required to work with the system. In order to deploy the application at the Web servers Shiny, one must have the locally installed R programming language environment, such as RStudio IDE [28].

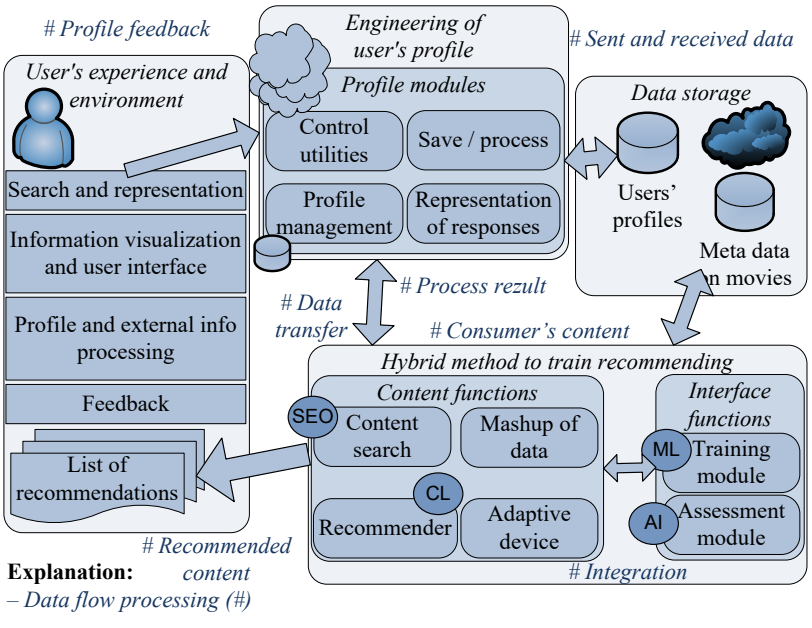


Fig. 15. Architecture of the combined type of RS [32]

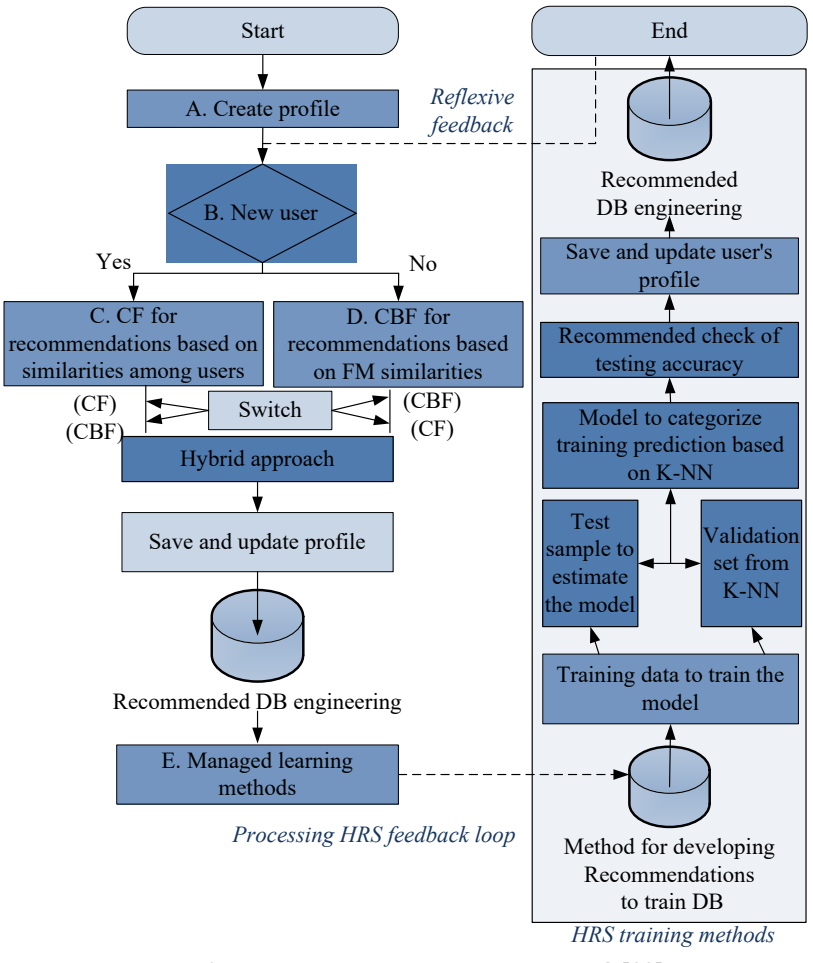


Fig. 16. Workflow of the combined type of RS [32]

6. Software for the distribution of commercial content in the Internet space based on collaborative filtering and Machine Learning

The result of implementing practical part of the current work is the designed system in the form of the Web application "VikToFilm" – a recommendation system for movies based on the hybrid algorithms of Machine Learning, including an algorithm for the User-based collaborative filtering. The backend is written in the R programming language, and the client part, that is the user interface, was designed using the frameworks Bootstrap and Shiny. Since this system is a Web based application, developed in the R programming language using the framework Shiny, the software necessary for its functioning is the Web server shinyapps [29] with our app deployed at it. The service Shiny operates in the cloud at servers, which are the property of RStudio Corporation. Each program is autonomous and works with any data downloaded along with the application, or data that are retrieved from the third-party data warehouses, such as databases or Web services. The main purpose of the system is to provide relevant recommendations on movies considering the personal needs of the user. The designed system is recommended to use for data collection on preferences by various people in choosing movies and providing daily recommendations. In addition, the system could be easily adapted for recommendations on music, books, cryptocurrency, or even exchange markets. The hybrid algorithm receives source information in a different form, normalizes it, and gives appropriate recommendations. Given this, the system could be used both in everyday life for the selection of movies and in large corporations in workflows. The main R-packages used in the development of the system are as follows: shiny version 1.3.2; shinydashboard version 0.7.1; knitr version 1.22; markdown version 0.9; proxy 0.4-17; recommenderlab 0.2-4; reshape2 1.4.3; ggplot2 3.1.1; ggvis 0.4.4; data.Table 1.12.2; DBI 1.0.0; openssl 1.3; htmltools 0.3.6; jsonlite 1.6; rsconnect 0.8.13. The application is built based on the Machine Learning hybrid algorithms, including the User-based collaborative filtering algorithm. The main idea of the CFR system (collaborative filtering) implies that if two users had the same interests in the past, for example, they were pleased with the same film, these users would have similar preferences in the future. If, for example, user *A* and user *B* have a similar history of watching movies and user *A* recently has recently watched a movie and praised it, while user *B* has not yet watched this movie, this movie would be recommended to user *B*. Data for processing were borrowed from *MovieLens* [30]. There, movies are rated based on a 5-point scale. Due to the impossibility to have access to powerful servers for computations, our project used the smallest available variant (*ml-latest-small.zip*), which at the time of downloading contained 100,000 ratings and 3,600 keywords (tags) for 9,000 movies. These data were created by 600 people over the period from January 1995 to September 2018 [22]. All selected users had already estimated no less than 20 movies. The developed HRS uses files *movies.csv* (Table 4) and *ratings.csv* (Table 5) from *MovieLens* to generate recommendations.

A brief description of data on movies according to the content of file *movies.csv*:

```
> summary(movies)
movieID          title              genres
Min. : 1         Length : 9125      Length : 9125
1st Qu. : 2850   Class : character  Class : character
Madian : 6290    Mode : character   Mode : character
Mean : 31123     3td Qu. : 56274    Max. : 164979
```

A brief description of data on movies ratings according to file *ratings.csv*:

```
> summary(ratings)
userID           movieId            rating           timestamp
Min. : 1         Min. : 1           Min. : 0.500     Min. : 7.897e+08
1st Qu. : 182    1st Qu. : 1028     1st Qu. : 3.000  1st Qu. : 9.658e+08
Madian : 367     Madian : 2406      Madian : 4.000   Madian : 1.110e+09
Mean : 347       Mean : 12549       Mean : 3.544     Mean : 1.130e+09
3td Qu. : 520    3td Qu. : 5418     3td Qu. : 4.000  3td Qu. : 1.296e+09
Max. : 671       Max. : 163949      Max. : 5.000     Max. : 1.477e+09
```

Table 4

Part of *movies.csv* with movies' identifiers, titles, and genres

1	movieId, title, genres
2	1, Toy Story (1995), Adventure|Animation|Children|Comedy|Fantasy
3	2, Jumanji (1995), Adventure|Children|Fantasy
4	3, Grumpier Old Men (1995), Comedy|Romance
5	4, Waiting to Exhale (1995), Comedy|Drama|Romance
6	5, Father of the Bride Part II (1995), Comedy
7	6, Heat (1995), Action|Crime|Thriller
8	7, Sabrina (1995), Comedy|Romance
9	8, Tom and Huck (1995), Adventure|Children
10	9, Sudden Death (1995), Action
11	10, Golden Eye (1995), Action|Adventure|Thriller
.....	

Before building a system of recommendations, it is necessary to preliminary process available data. First and foremost, we reorganized information about genres of movies such that it is possible for future users to search for movies they like within certain genres. In terms of a user-friendly interface, it is much easier for the user compared with choosing a movie from a list of all the available movies [23]. While designing the system, we constructed a matrix of appropriate genres for each movie. The next step is to build a search matrix, which would make it possible to search for any movie based any genre without much difficulty (Table 6). As one can see, each film can match ≥1 genre.

Table 5

Part of the content of file *ratings.csv* with unique users and movies IDs, movies ratings by users, and timestamps

1	userId, movieId, rating, timestamp
2	1, 31, 2.5, 1260759144
3	1, 1029, 3.0, 1260759179
4	1, 1061, 3.0, 1260759182
5	1, 1129, 2.0, 1260759185
6	1, 1172, 4.0, 1260759205
7	1, 1263, 2.0, 1260759151
8	1, 1287, 2.0, 1260759187
9	1, 1293, 2.0, 1260759148
10	1, 1339, 3.5, 1260759125
.....	

In order to apply data on ratings to construct a recommendation engine, we converted the matrix of rankings into a sparse matrix of the type *realRatingMatrix* using the library *recommenderlab*:

671×9066 rating matrix of class 'realRatingMatrix' with 100004 ratings.

The package *recommenderlab* contains some parameters for the recommendation algorithm:

```
> names(recommender_models)
[1] "ALS_realRatingMatrix" "ALS_implicit_realRatingMatrix" "IBCF_realRatingMatrix"
[4] "POPULAR_realRatingMatrix" "RANDOM_realRatingMatrix" "RERECOMMEND_realRatingMatrix"
[7] "SVD_realRatingMatrix" "SVDF_realRatingMatrix" "UBCF_realRatingMatrix"
> lapply(recommender_models, "[[", "description")
$ALS_realRatingMatrix
[1] "Recommender for explicit ratigs based on latent factors, calculated by alternating least squares algorithm."
$ALS_implicit_realRatingMatrix
[1] "Recommender for imlicit data based on latent factors, calculated by alternating least squares algorithm."
$IBCF_realRatingMatrix
[1] "Recommender based on element-based collaborative filtering."
$POPULAR_realRatingMatrix
[1] "Recommender based on element popularity."
RANDOM_realRatingMatrix
[1] "Produce random recommendations (real ratings)."
RERECOMMEND_realRatingMatrix
[1] "Re-recommends highly rated elements (real ratings)."
SVD_realRatingMatrix
[1] "Recommender based on SVD approximation with column-mean imputation."
SVDF_realRatingMatrix
[1] "Recommender based on Funk SVD with gradient descend."
UBCF_realRatingMatrix
[1] "Recommender based on user-based collaborative filterng."
```

Table 6

Part of a search matrix based on a movie genre

N	movieId	title	Action	Adventure	Animation	Children	Comedy	Crime	Documentary	...
1	1	Toy Story (1995)	0	1	1	1	1	0	0	...
2	2	Jumanji (1995)	0	1	0	1	0	0	0	...
3	3	Grumpier Old Men (1995)	0	0	0	0	1	0	0	...
4	4	Waiting to Exhale (1995)	0	0	0	0	1	0	0	...
5	5	Father of the Bride Part II (1995)	0	0	0	0	1	0	0	...
6	6	Heat (1995)	1	0	0	0	0	1	0	...
.....										

When designing the system, we decided to apply the models IBCF and UBCF [25], which have the following parameters:

```
> recommender_models$IBCF_realRatingMatrix$parameters
$k                      [1] 30
$method                 [1] "cosine"
$normalize              [1] "center"
$normalize_sim_matrix   [1] FALSE
$alpha                  [1] 0.5
$na_as_zero             [1] FALSE
> recommender_models$UBCF_realRatingMatrix$parameters
$method                 [1] "cosine"
$nn                     [1] 25
$sample                 [1] FALSE
$normalize              [1] "center"
```

7. Results of studying the implementation of the system for distributing commercial content in the Internet space according to the user's needs

The Web application "VikToFilm" is the recommendation system on movies based on the Machine Learning hybrid algorithms, including the User-based collaborative filtering algorithm. The user interface is designed using the framework Shiny (Fig. 17):

In the course of the system design, we used the following programming environments and languages:

– the R programming language, version 3.3.3;

– the RStudio IDE environment, version 1.2.1335;

– frameworks Shiny, version 1.3.2, HTML5/CSS3 and Bootstrap, version 4.3.1;

– libraries jQuery, version 3.3.1, and SockJS;

– operating system Windows 10;

– Web browser Google Chrome, version 74.0.3729.131.

The main purpose of the system is to provide relevant recommendations on movies considering the personal needs of the user. The designed system is recommended for use to collect data on preferences by various people in choosing movies and providing daily recommendations.

In addition, the system can be easily adapted for recommending music, books, cryptocurrency, or even exchange markets. The hybrid algorithm would receive source information in different forms, normalize it, and provide appropriate recommendations. Given this, the system could be used both in everyday life for the selection of movies and at large corporations in their workflows. Since this is a cross-browser and multiplatform Web application, then one can use it at any device that has a Web browser and access to the Internet (Fig. 18).

It is possible to access the application from any device with a Web browser and access the Internet. The input point to the Web application is its home page (https://viktorshatskykh.shinyapps.io/Source/). The software source data are a database of movies and ratings contained in the files *movies.csv* and *ratings.csv*. In order to develop a user-friendly Web application interface, the source data were simplified to a choice of three genres and three movies of respective genres. The source data are a list of the ten movies most relevant to a user. To confirm the feasibility of the designed Web application and to verify that the results from the system's operation match the set task, we used a control example (Fig. 19, 20). When a user visits the Web page of the application (https://viktorshatskykh.shinyapps.io/Source/) via his favorite Web browser (in this test, we used the Web browser Google

Chrome, version 74.0), he sees the home page of the Web application. Following the selection of three favorite genres and three favorite movies, the system generates relevant recommendations to him. These recommendations are 10 movies' titles and years when these movies were filmed and released to the screens worldwide.

Fig. 21 shows that the system does work and provides rather relevant recommendations on choosing movies. If a user wants to get different recommendations, it would suffice to change one of the three selected movies (there is a possibility to modify the choice at any stage of operating the system).

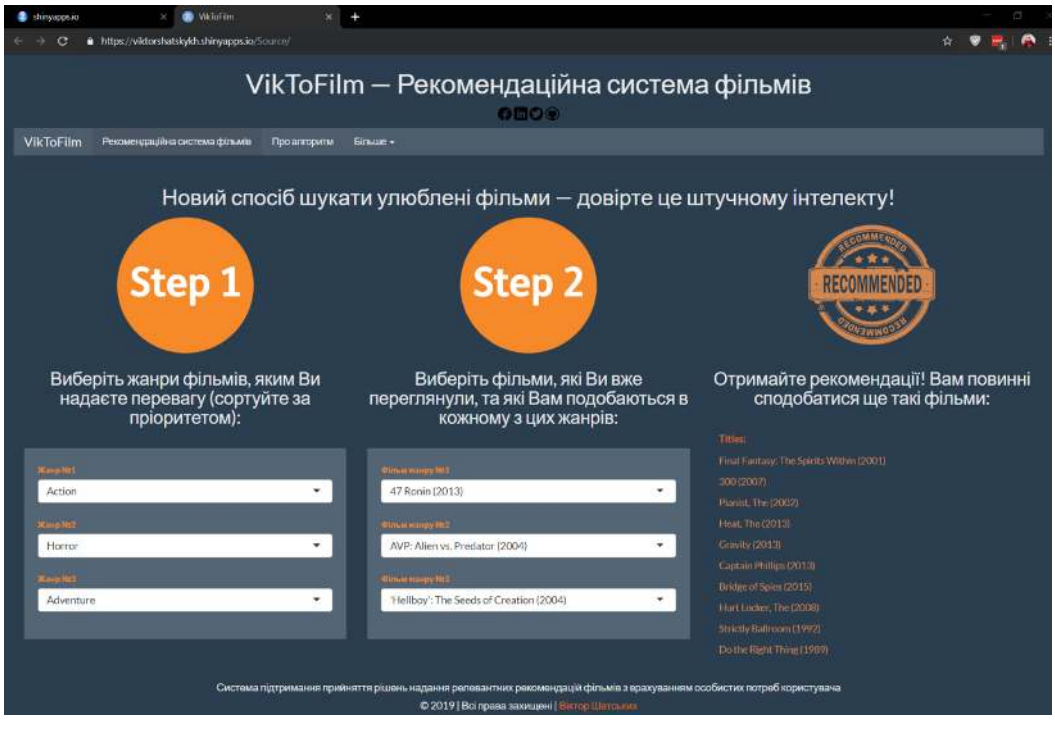
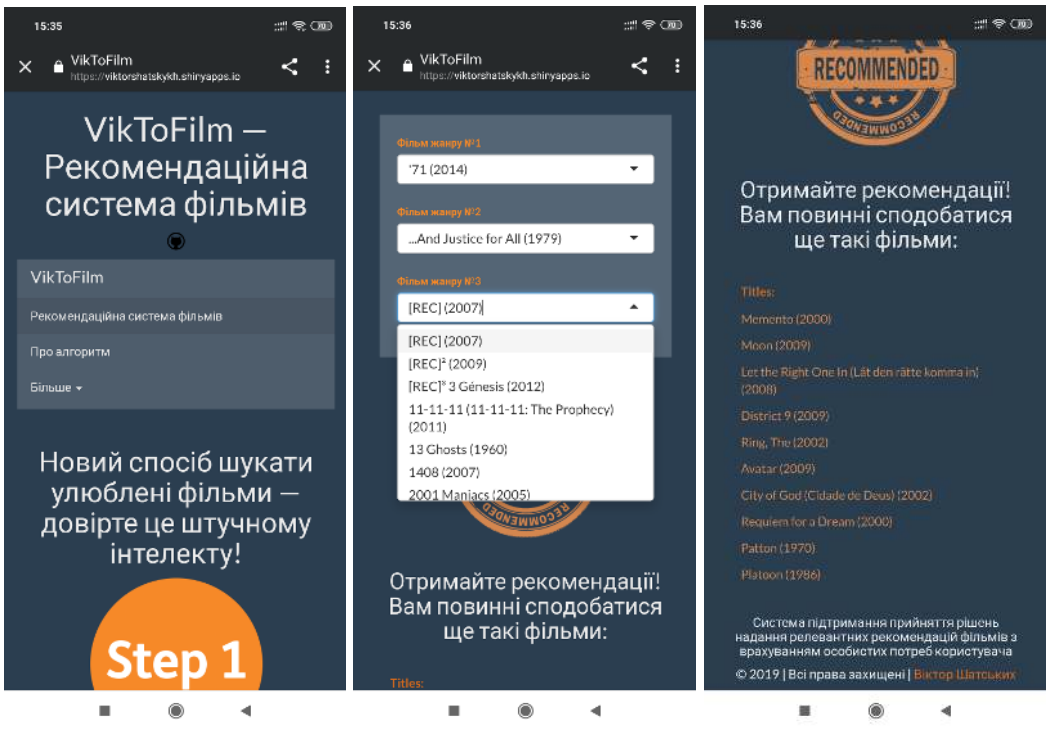


Fig. 17. Web application



a *b* *c*

Fig. 18. Android-based Web app: *a* — menu; *b* — genres; *c* — recommendations

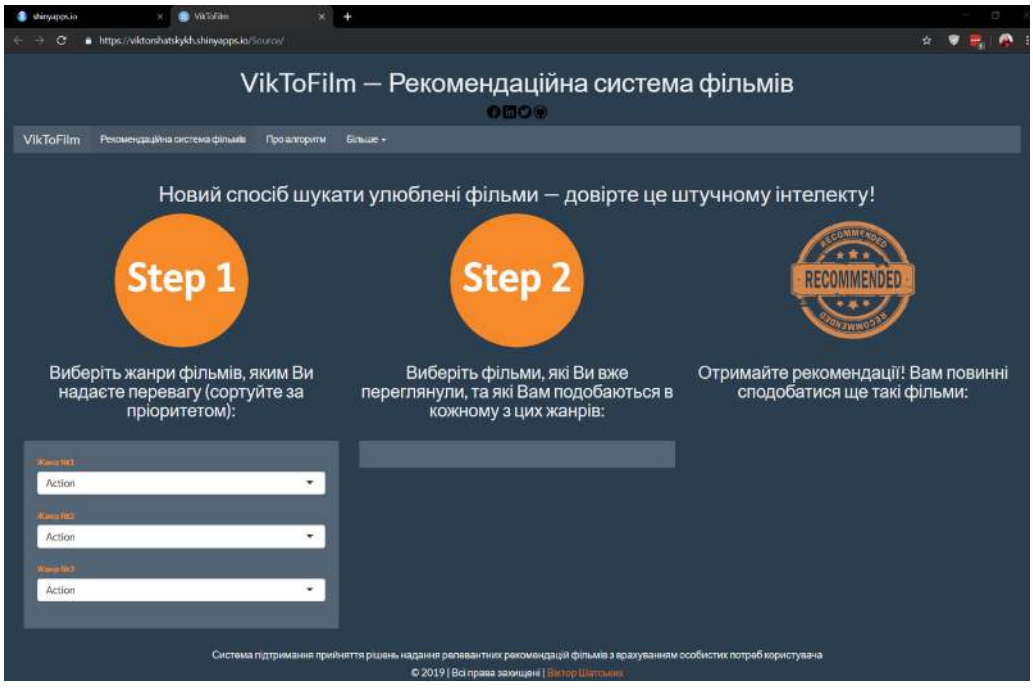


Fig. 19. Operation start

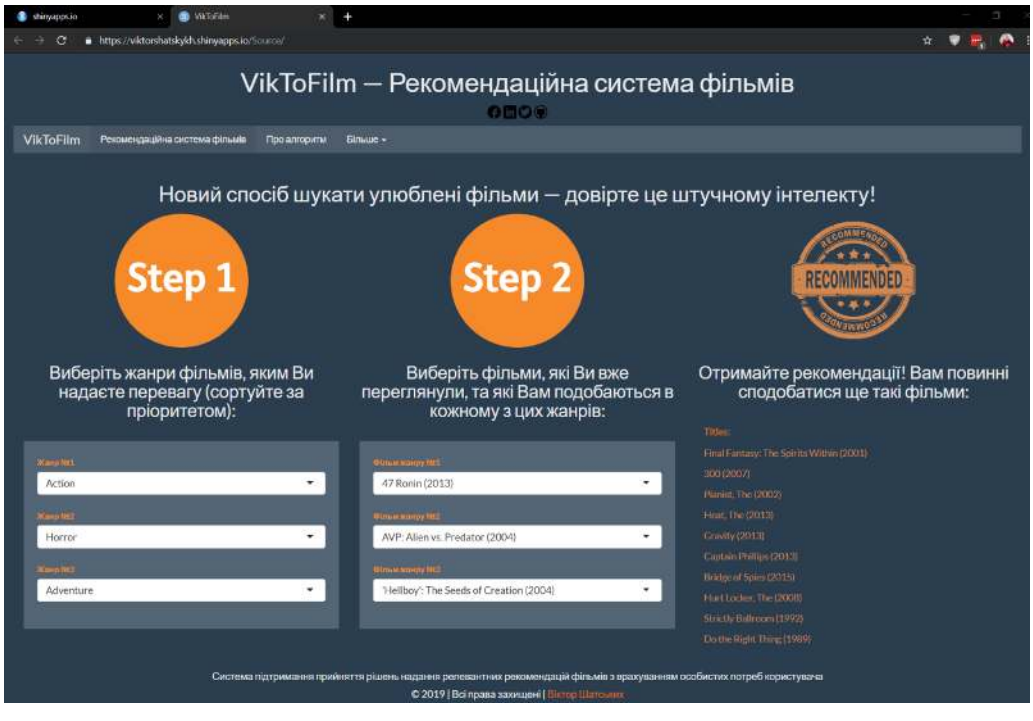


Fig. 20. Obtained recommendations

8. Discussion of results of studying the automated distribution of commercial content based on SEO technologies and Machine Learning

Algorithms for collaborative filtering are based on the degree of similarity among users or between elements. To this end, *recommenderlab* contains a similarity function. The following methods to compute similarities are employed:

– Cosine Similarity – cosine;
– Pearson correlation coefficient – pearson;
– Jaccard index – jaccard.

To find out to which extent the first four users/movies are similar to each other, we constructed and visualized a similarity matrix, which applies the cosine coefficient (Table 7, Fig. 21). In the matrix, each row and each column correspond to a user/movie, while each cell corresponds to the similarity between two users/movies. The darker the red cell, the more similar the users' two favorites are (the content of the two movies). It is worth noting that the diagonal is red: it contains a comparison of each user/movie to himself/itself.

Following the visualization of a similarity matrix of the first four movies, we examined the *weight of ratings*:

```
> vector_ratings <– as.vector(ratingmat@data)
> unique(vector_ratings) # which ratings' values are unique
[1] 0.0 3.0 4.0 5.0 2.0 3.5 1.0 2.5 4.5 1.5 0.5
> table_ratings <– table(vector_ratings) # the quantity of each rating
> table_ratings
vector_ratings
```

0	0.5	1	1.5	2	
5983282	1101	3326	1687	7271	
2.5	3	3.5	4	4.5	5
4449	20064	10538	28750	7723	15095

Table 7

Similarity matrices of users and movies based on a cosine coefficient

Similarity matrix of users				
	1	2	3	4
1	0.00000000	NA	NA	0.07448245
2	NA	0.000000	0.12429498	0.11882103
3	NA	0.124295	0.00000000	0.08163991
4	0.07448245	0.118821	0.08163991	0.00000000
Similarity matrix of movies				
	1	2	3	4
1	0.0000000	0.3945115	0.365159	0.1336141
2	0.3945115	0.0000000	0.2174915	0.1646513
3	0.3065159	0.2174915	0.0000000	0.1770118
4	0.1336141	0.1646513	0.1770118	0.0000000

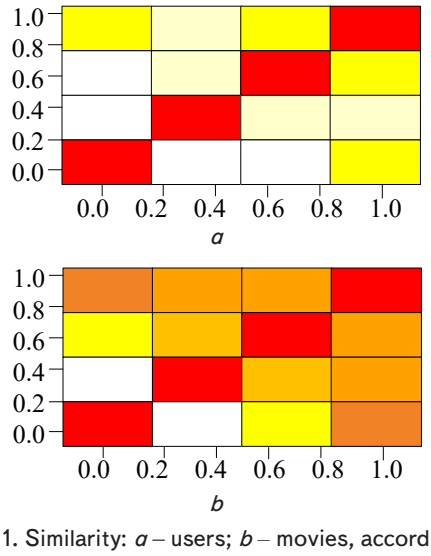


Fig. 21. Similarity: *a* – users; *b* – movies, according to MovieLens

In total, there are 11 unique values for estimates of the rating. Lower values denote lower ratings, and vice versa. If we disregard rating 0 (part of movies is not estimated by users), among the assessed movies, the most often rating is 4 (28750), and the least used rating is 0.5 (1101). According to documentation, if the rating is 0 it means the estimate is lacking, which is why we delete from the set of data before visualizing the results (Fig. 22). As shown by data, there are lower (less than 3) ratings, however, most movies are estimated by the score of 3 or higher. The most common is the rating of 4 points.

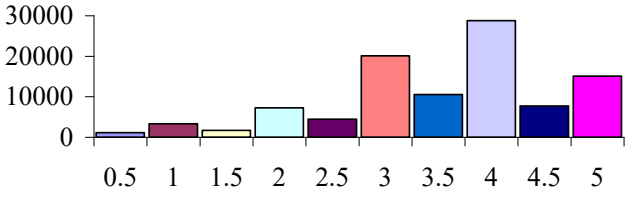


Fig. 22. Distribution of movies' ratings according to MovieLens (ml-latest-small. zip)

We shall consider which movies were watched by users the greatest number of times (Table 8) according to MovieLens (ml-latest-small.zip). The result obtained is due to the fact that the data were created by 600 people over the period from January 1995 to September 2018; they were not updated at the time of testing. That is why the most popular movies since 2015 have not yet received the corresponding ranking at MovieLens. The statistics are also limited by the small number of users and their regional base. However, even such a small sample demonstrates that such fundamental movies as Forrest Gump (1994), the Shawshank Redemption (1994), which change our society, ranked first among 9000 popular movies among a small audience of 600 people out of 100,000 ratings (about 170 per person) and 3,600 keywords. Probably, even in the updated sample, such movies, though not necessarily atop the list, would certainly be included in the list of the most popular movies for the Euro-American society. In other territories (Asia, Africa, etc.), users would probably choose different movies, while those listed in Table 8 might not enter the list of rated movies. The current article was not meant to include any demographic and regional statistical analysis of users' preferences.

Table 8

The most popular movies according to MovieLens (ml-latest-small.zip)

No.	movie	views	title
1	356	341	Forrest Gump (1994)
2	296	324	Pulp Fiction (1994)
3	318	311	Shawshank Redemption, The (1994)
4	593	304	Silence of the Lambs, The (1991)
5	260	291	Star Wars: Episode IV – A New Hope (1977)
6	480	274	Jurassic Park (1993)

Following the construction of rating of the largest number of views, we determined movies with the best rating and calculated the average score for each of them (Fig. 23). Fig. 23, *a* shows the distribution of the average rating of movies. The largest value is about 3, and there are several movies whose rating is equal to 1 or 5. It is possible that these movies were given an estimate by a few people only; hence, there is no need to take them into consideration. After removing the movies that gathered 50 views and below, we built a subset of relevant movies only. Fig. 23, *b* shows the distribution of respective relevant estimates. All ratings are between 2.16 and 4.55. As expected, the extrema were deleted. The greatest value for rating has changed to equal about 4.

We shall graphically represent the estimates matrix by constructing a heatmap of the matrix of rankings, whose color is ratings. Heatmap is a graphical representation of data, at which individual values contained in the matrix are represented in the form of colors. Each row of the matrix corresponds to a user, and each column – to a movie, therefore,

each cell at the intersection – to appropriate rating (Fig. 24). Since it shows a great number of users and elements, comprehending the chart including all 600 users and 9,000 movies might be difficult. Therefore, as an example, we constructed a chart at a larger scale that focuses on first data.

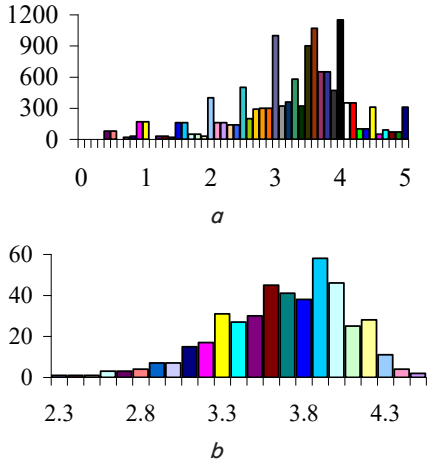


Fig. 23. Distribution according to MovieLens (ml-latest-small.zip): *a* – average rating for movies; *b* – mean relevant rating for movies

Some users viewed more movies than others. Thus, instead of showing any random users and movies, we choose the most relevant users and movies. In order to identify and select the most relevant users and movies, we shall perform the following steps (Fig. 25):

– Determine the minimum number of movies for each user.
– Determine the minimum number of users for each movie.
– Select users and movies that meet these criteria.

[1] "The minimum number of movies for each user":
99 % – 1131.4.
[1] "The minimum number of users for each movie":
99 % – 123.

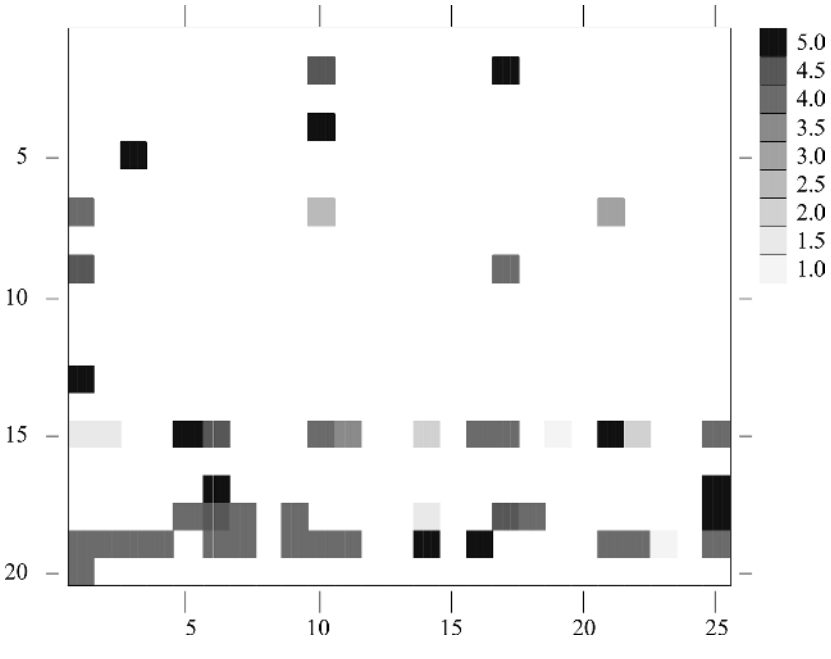


Fig. 24. Heatmap of ratings by first 20 users and of first 25 movies according to MovieLens (ml-latest-small.zip)

Those users should be taken into consideration who have watched the largest number of movies. Most of them have watched all the best movies. Some columns in a heatmap are darker than others, which means that these columns depict the highest rating of movies. Conversely, dark rows represent users that give the highest ratings. Therefore, it is appropriate to normalize the data.

The process of data preparation consists of the following steps [25]:

– select the relevant data;
– normalize the data;
– convert the data in a binary format.

To select the most relevant data, we determined the minimum number of users/movies that rated a movie/rated as 5.0:

421×444 rating matrix of class 'realRatingMatrix' with 37915 ratings.

Such a sample of the most relevant data contains 421 users and 444 movies (compared with previous 671 users and 9,066 movies in the overall dataset). By using the same approach applied above, we visualized top 2 % of users and movies in a new matrix of the most relevant data (Fig. 26). In the heatmap, some rows are darker than others since some users give higher ratings for all movies (Fig. 26).

Those users who give only high (or low) ratings for all the revised movies can ruin the results. To avoid this, the data were normalized such that the average score from each user corresponded to 0. For a quick check, we calculated the average score by each user, which is equal to 0 (data normalization):

```
> ratings_movies_norm <– normalize(ratings_movies)
> sum(rowMeans(ratings_movies_norm) > 0.00001)
[1] 0
```

Some models of recommendations are based on binary data, so it is useful to convert the data into a binary format, that is compile a table that contains only 0 and 1. Missing values or poor estimates can be considered to equal 0. In this case, it is advisable to:

– Build a matrix that contains a value of either 1, if a user estimated a movie, or 0 otherwise. In the latter case, the information about ratings is lost.

– Build a matrix that contains 1, if the score is higher or equal to a certain threshold value (for example, 3), and 0 otherwise. In this case, giving a poor rating to a movie means not estimating it at all.

The appropriateness of choosing any method depends on the context. By using two different approaches, we have constructed two matrices and visualized 5 % of each of the binary matrices (Fig. 27). In the first variant, we defined the matrix whose cells equal 1 if the film has been viewed (Fig. 27, *a*). In the second variant, we defined the matrix whose cells equal 1 if a cell has a rating above the threshold value (Fig. 27, *b*). The second heatmap includes more white

cells than the first one, which means that there are more movies with a missing or poor rating than those that have not been watched by users.

According to this approach, one searches, for a new user, the users similar to him. And the movies to which these users gave high ratings are being recommended to the new user [20].

For each new user, the following steps are performed:

1. Determining a degree of similarity between each user-neighbor and the new one. Similar to IBCF, the cosine coefficient or the Pearson correlation coefficient are applied.

2. Identifying the most similar users. Variants:

– Use top *k* users (*k* nearest neighbors).

– Use those users whose similarity exceeds a certain threshold value.

3. Estimation of movies according to the ranking compiled by the most similar users. The resulting rating is the average rating among similar users. The following approaches are used:

– Average rating.

– Average weighted rating that employs similarity as weights.

4. Obtaining the most relevant movies.

First, one should check the default settings in the UBCF model. Here, *nn* is the number of similar users [24]. The similarity function uses by default a cosine coefficient. Therefore, it is necessary to build a model using standard parameters and applying a training dataset:

```
$method
[1] "cosine"
$nn
[1] 25
$sample
[1] FALSE
$normalize
[1] "center"
Recommender of type 'UBCF' for 'realRatingMatrix'
Learned using 330 users.
330 x 444 rating matrix of class 'realRatingMatrix' with 29917 ratings.
Normalized using center on rows.
```

We have defined the ten best recommendations for each new user: Recommendations as 'topNList' with n=10 for 91 users. The matrix from Table 9 contains *movieId* for each recommended movie (rows) for first four users (columns) of the entire data set () blue color denotes the most relevant movies based on UBCF).

In addition, we computed how many times each movie is recommended and built a corresponding frequency histogram (Fig. 28).

Compared with IBCF, the distribution has a longer tail. This means that some movies are recommended much oftener than others [26]. A maximum is more than 30, compared with 10 for IBCF (Table 10).

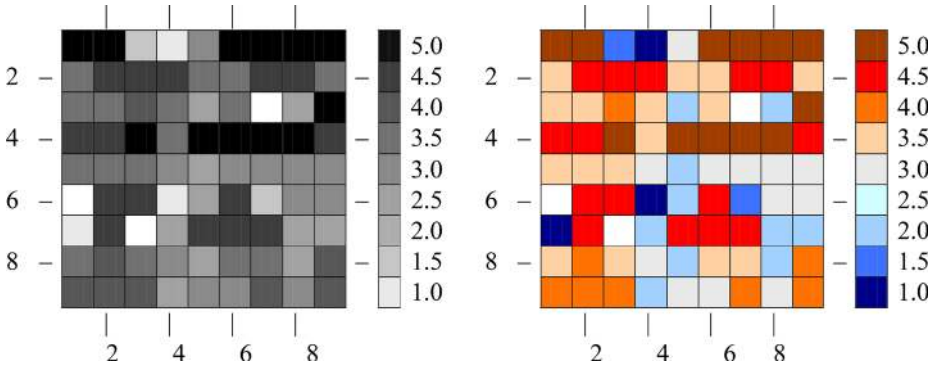


Fig. 25. Heatmap of the best users and movies according to MovieLens

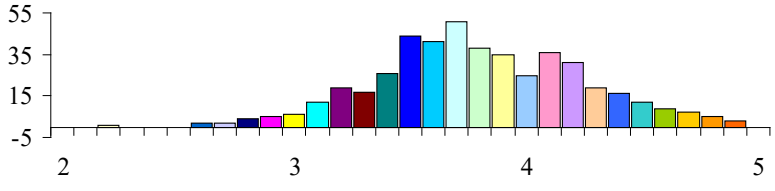


Fig. 26. Distribution of average rating for each active user in accordance with MovieLens (ml-latest-small.zip)

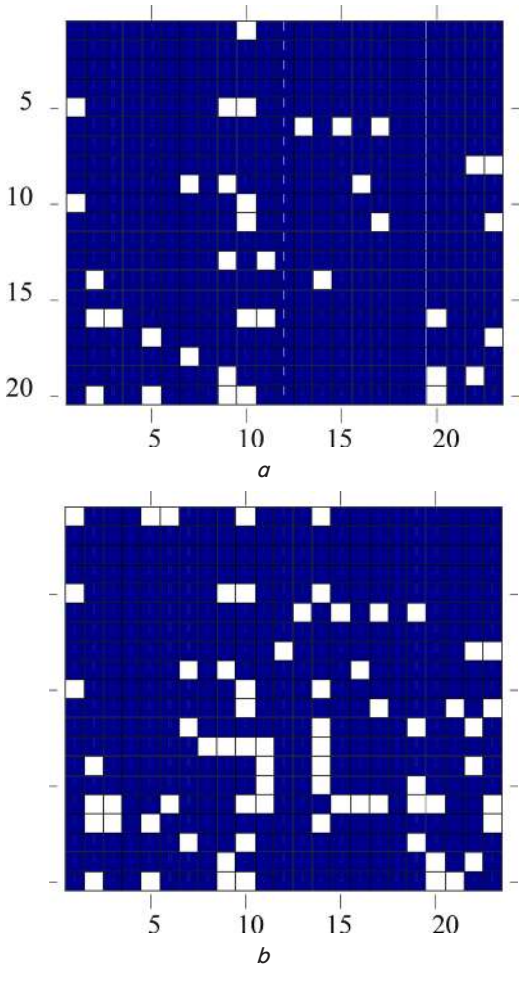


Fig. 27. Heatmap according to MovieLens (ml-latest-small.zip): *a* – 1, if a film has been watched; *b* – 1, the cell has a rating above the threshold value

UBCF must have access to the original data and store the entire set of information in memory, which often makes it difficult to work given a large matrix of rankings. In addi-

tion, building a similarity matrix requires much computing power, as well as time. However, the accuracy from UBCF is better than that from IBCF, and UBCF is a good choice if the dataset is not very large [21].

Table 9

Matrix with recommendations for first four users (UBCF)

	[,1]	[,2]	[,3]	[,4]
[1,]	1089	608	858	1090
[2,]	1206	318	260	595
[3,]	293	527	318	1080
[4,]	858	47	899	2804
[5,]	47	593	50	1278
[6,]	1732	2959	246	1674
[7,]	1213	50	111	2529
[8,]	6016	2571	750	596
[9,]	924	223	1198	661
[10,]	111	923	923	1285

Table 10

The most relevant movies (UBCF) according to MovieLens (ml-latest-small.zip)

	Movie title	Number of elements
50	Usual Suspects, The (1995)	39
858	Godfather, The (1972)	34
318	Shawshank Redemption, The (1994)	32
527	Schindler's List (1993)	32

Depending on analysis of the history of a particular user (the number of movies watched, his estimates for specific movies, the number of movies watched in a specific genre, the number of chosen genres, etc.), one can adjust training a system to develop recommendations to this user. One can also derive the criteria for the evaluation of methods for their further comparison. We shall take the most common parameter – the number of watched movies with their subsequent evaluation by the user. It is logical – the more movies a user has watched, the more estimates he gave to these movies – the better is the list of movies recommended to user that HRS could compile in the future. However, each of the above-considered methods has several disadvantages that affect the evaluation of the relevance of movies in a recommended list for a specific user. We shall take three groups of users *A*–*C* composed of the same number of people, who have watched movies and rated them respectively. Group *A* includes those that watched up to 100 movies and rated them respectively; group *B* – between 100 and 150 inclue sive, Group *C* – more than 150 movies (Fig. 29). Experimental results show that the performance indicators of the proposed HRS based on the technology of CF+CBF+ML outperform those in two individual models, CF and CBF, as well as their combinations CF+CBF, CF+ML, and CBF+ML.

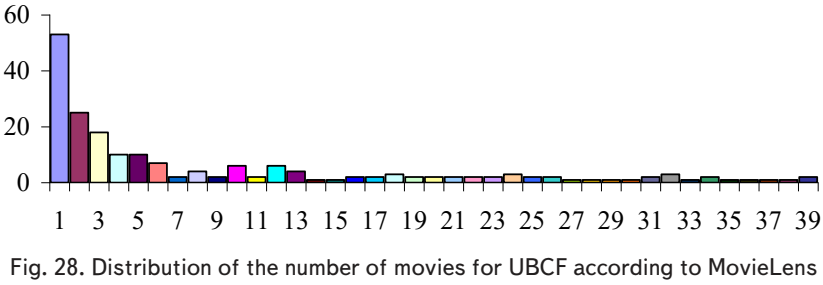


Fig. 28. Distribution of the number of movies for UBCF according to MovieLens

Much to our regret, it is impossible to compare our results to at least one of the known systems because those are commercial projects and no geek would give access to the system's operational statistics, nor to the results from applying recommendations and ML in these systems. However, we shall describe these systems from a user's viewpoint using the examples of the most common IMDb, Netflix, and Rotten Tomatoes.

1. IMDb develops personal recommendations that help users find new movies and tv shows. Recommended movies are shown at the section "Recommended For You" at the home page of IMDb [14]. IMDb accepts all movies and tv shows that a user rated or added to his watchlist, and then compares these data with the ratings by other users of IMDb. IMDb then can find movies and tv shows assigned to people with similar preferences. For each recommendation, a user can see a list of movies or tv shows on which the recommendation is based [14]. When a user rates a movie favorably or adds a movie to his watchlist, IMDb keeps track of this event as the movie that is of interest to the user. If IMDb has no recommendations to a user (either because the user exited the system, or because he did not estimate many elements), IMDb shows the user a list of elements that many people have seen so that the user could enter ratings to obtain recommendations. To improve the personalized recommendations, a user must find and assess more movies. New ratings exert a direct impact on recommendations. Upon assessing and adding titles to the watchlist, one needs to reload the page to update recommendations from IMDb [14].

2. Netflix splits viewers into more than two thousand groups based on preferences. More than 80 % of television programs that people watch at Netflix are provided through a system of recommendations. This means that most content that users have decided to watch at Netflix is the result of decisions taken by the recommendation system. Netflix employs machine learning and the algorithms that help overcome a previously perceived notion by users and suggest the content that they might have initially rejected. To this end, He looks at the anomalies of flow in content rather than relies on broad genres to make the predictions. That explains why, for example, one of the eight users who watch one of the shows, Marvel Netflix, happens to be completely new. Even though Netflix has over 100 million users around the world, given the multiple user profiles for each subscriber, that would total 250 million active profiles. What Netflix grabs from these profiles is the data on what people watch, observe, what they watched before, what they watched a year ago or recently, and at what time during the day. These data form the first part of the algorithm [15]. Then this information is combined with more data, aimed at understanding the content of a show. The second part of the algorithm is collected from dozens full-time and part-time employees who watch each show at Netflix and mark it with tags. Netflix takes all these tags and data on users' behavior, then uses sophisticated algorithms of machine learning, which define what is important. What must be the value for a rating if a user watched the movie yesterday? Should it count twice or ten times more than what users watched a year ago? How about a month ago? How about what they watched ten minutes and cancelled, or they watched it again in two nights? Machine learning helps Netflix deal with

it. Viewers are assigned to several groups of preferences – of which there are "a few thousand" – those that affect the recommendations popping up at the upper part of the screen, which genre lists are displayed, and the way each row is arranged for each particular user [15]. Tags that are used for the algorithms of machine learning are the same around the world. However, smaller subsets of tags are used more extensively, directly at the user interface, and differ depending on the country, language, and cultural context.

3. Rotten Tomatoes is one of the largest web-sites of responses and reviews on movies. They retain the cumulative rating of critics [16]. Rotten Tomatoes considers all reviews by critics in the entertainment industry. They estimate movies not based on a 1–10 scale, but 1–100, though only in two aspects – "Rotten" or "Fresh" ("Rotten" means it is not preferred, while "Fresh" means having been appreciated). So, if a movie is rated above 60 %, this means that the critics do not like it, but the audience must like it. When rated above 80 % by a certain number of "certified critics", a movie is considered "Fresh", worth recommending. The average rating is the average of all ratings (out of 10) that professional critics gave to the movie. Audience rating is the average score provided by users of Rotten Tomatoes (out of 5). The status of "Certified Fresh" is a special award given to the best movies and tv shows [16]. To qualify, movies or tv shows should meet the following requirements:

– Tomatometer steady score – 75 % or above;

– at least five reviews from Top Critics;

– movies with a wide release should have not less than 80 reviews;

– movies with a limited release should have not less than 40 reviews;

– only certain seasons of television shows are accepted: each of them must have not less than 20 reviews.

A movie or a tv show, which meet the requirements to "Certified Fresh", are not immediately given the award. Instead, they are automatically marked to be reviewed by the Rotten Tomatoes staff. Once the team can define that the score would hardly fall below minimum requirements, they could designate it to be "Certified Fresh." If the assessment based on Tomatometer drops below 70 %, then the movie or the tv show loses the status of being "Certified Fresh." Assessment of the audience marked by popcorn is the percentage of users who positively rated the movie or the tv show. If at least 60 % of the users rated a movie or a tv show to have 3.5 stars or higher, one sees a full bucket of popcorn indicating its popular status ("Fresh status") [16]. When less than 60 % of the users rated a movie or a tv show to have 3.5 stars or higher, the display shows a flipped bucket of popcorn, which indicates its low popular status ("Rotten status"). A plus sign is displayed for a tv show without any estimates from the audience ("Audience Scores"). The percentage that one sees is associated with this icon, this is the percentage of those users who have indicated that they want to watch these shows, as opposed to the users who have indicated that they do not want to. After reviewing in detail the problems related to methods for generating recommendations in recommendation systems, such as a cold start, data sparsity, accuracy, scalability and diversity, and by considering known means to solve these issues, one can draw a conclusion that it is necessary to use the hybrid systems of recommendations to make the modelled system efficient. The application of RS is still common for simple and low-cost products, such as movies, music, news, and books. Although there are systems that manage more complex types of elements, such as financial investment or travel, these categories of products are considered as typical cases. Much more research is needed into algorithms and user interfaces to construct coherent sequences of recommendations. In particular, one should simulate the impact on the user from several contextual conditions such as the way in which the already shown earlier recommendations could be shown, and the way this would affect the assessment of future recommendations by the user. Of course, fighting human preferences is an extremely complex problem, especially when in many cases a user can decide to use the system without knowing which genre he digs in. Each person is different in what brings him joy and the way it affects the choice of a movie to watch. In addition, people do change over time, so the algorithms must help anticipate these changes. To take into consideration the human factor, it is necessary to supplement the algorithmic approaches with ideas that could be derived from the research into such systems and their metadata.

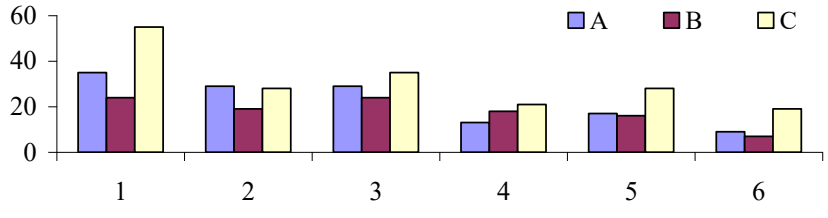


Fig. 29. Comparative characteristic (in percentage) of the growth in error in recommendations when using respective methods: 1 – CF, 2 – CBF, 3 – CF+CBF, 4 – CF+ML, 5 – CBF+ML, 6 – CF+CBF+ML

9. Conclusions

1. We have considered a task on designing an intelligent system for the commercial distribution of information products using the personalized approach to users based on the categories and tags of content that is interesting to the users. Application of the recommendation systems is still common for simple and low-cost products, such as movies, music, news, and books. Although there are systems that manage more complex types of elements, such as financial investment or travel, these categories of products are considered to be typical cases. Much more research is needed into algorithms and user interfaces to create coherent sequences of recommendations. Specifically, one should simulate the impact on the user from several contextual conditions such as the way the already shown recommendations could be shown again, and how this would affect the assessment of future recommendations by the user. Of course, fighting human preferences is an extremely complex problem, especially when in many cases a user decides to use the system without knowing which genre he digs in. Each person is different in what brings him joy and the way it affects the choice of a movie to watch. In addition, people change over time, so the algorithms must help anticipate these changes. To take into consideration the human factor, it is necessary to supplement the algorithmic approaches with ideas that could be derived from the research into such systems and their metadata. We have considered the methods of content filtering, collaborative filtering, as well as the hybrid methods. Analysis of the collaborative filtering algorithms has been performed. We have substantiated the choice of the Machine Learning hybrid algorithms, including

the User-based collaborative filtering algorithm to develop recommendations on movies for the system's users. The expediency to design a decision support system for providing relevant recommendations on movies considering the personal needs of the user has been justified.

2. We have constructed a method for the personalization of commercial content according to the user's needs based on collaborative filtering and Machine Learning. Given the growth in the volume of information and data, there are various problems related to the methods of filtering, including density and scalability. The Machine Learning data for the developed system were borrowed from *MovieLens* [30]. There, a movie is rated based on a *5-point scale*. Due to the impossibility to have access to powerful servers for computations, the current project employed the smallest available variant (ml-latest-small.zip), which at the time of downloading contained 100,000 ratings and 3,600 keyd words (tags) on 9,000 movies. These data were created by 600 people over the period from January 1995 to September 2018 [22]. All selected users had already rated not less than 20 movies. The born HRS uses the files *movies.csv* (Table 4) and *ratings.csv* (Table 5) from MovieLens to generate recommendations. However, Machine Learning was based on the English-language data due the current absence of similar statistics collected on the preferences of people in Ukraine. Despite this, the constructed method could be used in the future for any keywords (tags) in different languages. The parser embedded into the system does not depend on the language of search keyword by the users of the system. It only depends on the content of dictionaries and a database on movies at HRS. The newest trends in Machine Learning employ different layers of processing, which helps train them using complex contextual features. According to experts, it was found that Machine Learning is able to define those factors that influence the selection of relevant movies that improves the development of recommendations specific to the user. To solve these tasks, a new improved training method is proposed underlying which, in contrast to existing systems of recommendations, is the hybrid methods, based on Machine Learning. The main purpose of the system is to provide relevant recommendations on movies considering the personal needs of the user. The designed system is recommended for data collection on preferences by various people in choosing movies and providing respective daily recommendations. In addition, the system could be easily adapted for recommending music, books, cryptocurrency, or even exchange markets. The hybrid algorithm would receive source information in different forms, normalize it, and fire off appropriate recommendations. Given this, the system could be used both in everyday life for the selection of movies and in large corporations in their workflows. We have designed the structure for the decision support system to provide relevant recommendations on movies considering the personal needs of the user. A systems analysis of the proposed system has been carried out. We have designed the architecture of the system. A structural analysis has been performed and the software modules for the system of relevance of movies considering the user's needs have been designed. We have carried out analysis of the goal of the system operation, elucidating specific aspects and criteria of quality. We have grown the system's objectives tree and analyzed HRS using the systemic method of analytical hierarchy. Main variants to achieve the objective and available resources have been analyzed. In addition, we have built and described the UML diagrams. Moreover, we have clearly defined and described such issues as the purpose of the system, its purpose, location of application, substantiation of development, expected effect from its implementation; a conceptual model of the system has been constructed.

3. We have designed an intelligent system for the distrit bution of commercial content in the Internet space based on SEO technologies, neural networks, and Machine Learning. General functional requirements to a system for distributing commercial content in the Internet space have been stated. The developed system, based on modern methods of SEO technologies considering the metrics of performance evaluation of an information-search module of the system, makes it possible to select relevant content according to the user's personalized interests. The system includes classes and 6, which include the real commercial information products, which built the logical connection that happens intelligent presentation of content personalization based on the needs and interests of the user. In addition, based on the modern methods of Machine Learning, the designed system learns to refine search results for requested content according to a user's preferences and personalization. The developed decision support system to provide relevant recommendations on movies taking into consideration a user's personal needs is a free to access, and deployed at a free server. However, to successfully develop and improve it, and to be able to migrate the system to a more stable commercial server, it would be expedient to attract funds from investors, or regular donations from grateful users. Therefore, in my opinion, this system is viable in terms of its relevance and possible return on investment. This system is very useful for end users because it saves considerable time required for the selection of relevant movies for each. Therefore, it is in demand in terms of acceptance by consumers in the market of information technologies. Today's software market of recommendations on movies is shared by such giants as Netflix, Rotten Tomatoes, Movielens, and IMDb. However, they all are foreign and in English. The Ukraine's market has no analogs. To remedy the situation, it is advisable to create the system presented here. The long-term development of this market implies the creation and improvement of the Ukrainian-language systems to recommend movies. Since Ukraine has no established analogs of recommendation systems on movies, then there is a need to develop such software.

4. We have practically implemented a decision-support system to provide relevant recommendations on movies taking into consideration a user's personal needs based on the constructed algorithm. Performance of the decision-support system to provide relevant recommendations on movies taking into consideration a user's personal needs has been analyzed. We have improved an algorithm to form recommendations by the system, which is based on the Machine Learning hybrid algorithms, including the User-based collaborative filtering algorithm. The result of implementing practical part of the current work is the designed system in the form of "VikToMovie" Web application – a recommendation system on movies based on the Machine Learning hybrid algorithms, including the User-based collaborative filtering algorithm. The paper describes the software, general information, functional purpose, the logical structure, the technical means applied, launch and download, the input and output data on the system. Analysis of the control example of the system's operation is given. We have analyzed results from experimental testing of the proposed method for the personalization of commercial content according to the us-

er's needs. Operation of the decision-support system to provide relevant recommendations on movies taking into consideration a user's personal needs has been analyzed. The algorithm of the system's operation has been constructed based on the Machine Learning hybrid algorithms, including the User-based collaborative filtering algorithm. The result of implementing practical part of the current work is the designed system in the form of "VikToMovie" Web application – a recommendation system on movies based on the Machine Learning hybrid algorithms. The paper describes the software, general information, functional purpose, the logical structure, the technical means applied, launch and download, the input and output data for the system. Analysis of the control example of the software operation is given. We have completed the practical implementation of designing a decision-support system to provide relevant recommendations on movies taking into consideration a user's personal needs. The system is recommended for use to collect data on the preferences by different people in choosing movies and providing respective relevant recommendations. In the further research in this field, it is advisable to improve the technique for storing data on movies and replace the algorithm for ranking movies based on reviews by taking a hybrid algorithm based on taking into consideration the qualitative and quantitative characteristics of reviews.

References

1. Melville, P., Mooney, R., Nagarajan, R. (2016). Content-Boosted Collaborative Filtering for Improved Recommendations. National Conference on Artificial Intelligence: «AAAI-2002», 187–192.
2. Lytvyn, V., Vysotska, V., Demchuk, A., Demkiv, I., Ukhanska, O., Hladun, V. et. al. (2019). Design of the architecture of an intelligent system for distributing commercial content in the internet space based on SEO-technologies, neural networks, and Machine Learning. Eastern-European Journal of Enterprise Technologies, 2 (2 (98)), 15–34. doi: https://doi.org/10.15587/1729-4061.2019.164441
3. Jones, M. T. (2013). Recommender systems, Part 1. Introduction to approaches and algorithms. Available at: https://www.ibm.com/developerworks/opensource/library/os-recommender1
4. Su, X., Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, 2009, 1–19. doi: https://doi.org/10.1155/2009/421425
5. Burov, Y., Vysotska, V., Kravets, P. (2019). Ontological approach to plot analysis and modeling. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Systems (COLINS-2019). Volume I: Main Conference, 2362, 22–31.
6. Sarwar, B., Karypis, G., Konstan, J., Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. Proceedings of the Tenth International Conference on World Wide Web - WWW '01. doi: https://doi.org/10.1145/371920.372071
7. Schafer, J. B., Konstan, J., Riedi, J. (1999). Recommender systems in e-commerce. Proceedings of the 1st ACM Conference on Electronic Commerce - EC '99. doi: https://doi.org/10.1145/336992.337035
8. Gope, J., Jain, S. K. (2017). A survey on solving cold start problem in recommender systems. 2017 International Conference on Computing, Communication and Automation (ICCCA). doi: https://doi.org/10.1109/ccaa.2017.8229786
9. Ge, M., Delgado-Battenfeld, C., Jannach, D. (2010). Beyond accuracy: Evaluating recommender systems by coverage and serendipity. Proceedings of the fourth ACM conference on Recommender systems - RecSys '10, 257–260. doi: https://doi.org/10.1145/1864708.1864761
10. Bobadilla, J., Ortega, F., Hernando, A., Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. Knowledge-Based Systems, 26, 225–238. doi: https://doi.org/10.1016/j.knosys.2011.07.021
11. Nambiar, R., Bhardwaj, R., Sethi, A., Vargheese, R. (2013). A look at challenges and opportunities of Big Data analytics in healthcare. 2013 IEEE International Conference on Big Data. doi: https://doi.org/10.1109/bigdata.2013.6691753
12. Calero Valdez, A., Ziefle, M., Verbert, K. (2016). HCI for recommender systems: The past, the present and the future. RecSys '16 Proceedings of the 10th ACM Conference on Recommender System, 123–126. doi: https://doi.org/10.1145/2959100.2959158
13. Kotsiantis, S. B., Zaharakis, I., Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Frontiers in Artificial Intelligence and Applications, Volume 160: Emerging Artificial Intelligence Applications in Computer Engineering, 3–24.
14. Recommended For You FAQ. Available at: https://help.imdb.com/article/imdb/discover-watch/recommended-for-you-faq/GPZ2RSPB3CPVL86Z/
15. Netflix Prize. Available at: https://www.netflixprize.com/
16. About Rotten Tomatoes. Available at: https://www.rottentomatoes.com/about
17. Lytvyn, V., Vysotska, V., Rzheuskyi, A. (2019). Technology for the Psychological Portraits Formation of Social Networks Users for the IT Specialists Recruitment Based on Big Five, NLP and Big Data Analysis. Proceedings of the 1st International Workshop on Control, Optimisation and Analytical Processing of Social Networks (COAPSN-2019), 2392, 147–171.
18. Lytvyn, V., Vysotska, V., Rusyn, B., Pohreliuk, L., Berezin, P., Naum, O. (2019). Textual Content Categorizing Technology Development Based on Ontology. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 234–254.
19. Lytvyn, V., Kuchkovskiy, V., Vysotska, V., Markiv, O., Pabyrivskyy, V. (2018). Architecture of System for Content Integration and Formation Based on Cryptographic Consumer Needs. 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2018.8526669
20. Rubens, N., Elahi, M., Sugiyama, M., Kaplan, D. (2015). Active Learning in Recommender Systems. Recommender Systems Handbook, 809–846. doi: https://doi.org/10.1007/978-1-4899-7637-6_24
21. Ms. Ashwini A. Chirde, Ms. Urmila K. (2015). Combination of a Cluster-Based and Content-Based Collaborative Filtering Approach for Recommender System. International Journal on Recent and Innovation Trends in Computing and Communication, 3 (7), 4770–4774.

22. Harper, F. M., Konstan, J. A. (2015). The MovieLens Datasets. ACM Transactions on Interactive Intelligent Systems, 5 (4), 1–19. doi: https://doi.org/10.1145/2827872
23. Grolemund, G. (2015). Hands-On Programming with R: Write Your Own Functions and Simulations. Sebastopol, United States.
24. McLeod, D., Chen, A.-Y. (2009). Collaborative Filtering for Information Recommendation Systems. Research Reports.
25. Ricci, F., Rokach, L., Shapira, B. (Eds.) (2015). Recommender Systems Handbook. Springer. doi: https://doi.org/10.1007/978-1-4899-7637-6
26. Linden, G., Smith, B., York, J. (2003). Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, 7 (1), 76–80. doi: https://doi.org/10.1109/mic.2003.1167344
27. The Comprehensive R Archive Network. Available at: https://cran.r-project.org
28. RStudio. Available at: https://www.rstudio.com/products
29. Chapter 2 Getting Started. Available at: https://docs.rstudio.com/shinyapps.io/getting-started.html
30. MovieLens Latest Datasets. Available at: https://grouplens.org/datasets/movielens/latest
31. Sitecore Documentation: Access all the latest Sitecore documentation. Available at: https://doc.sitecore.com
32. Nouh, R., Lee, H.-H., Lee, W.-J., Lee, J.-D. (2019). A Smart Recommender Based on Hybrid Learning Methods for Personal Well-Being Services. Sensors, 19 (2), 431. doi: https://doi.org/10.3390/s19020431
33. Mobasher, B. (2007). Data Mining for Web Personalization. Lecture Notes in Computer Science, 90–135. doi: https://doi.org/10.1007/978-3-540-72079-9_3
34. Berko, A., Alieksieiev, V. (2018). A Method to Solve Uncertainty Problem for Big Data Sources. 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). doi: https://doi.org/10.1109/dsmp.2018.8478460
35. Xu, G., Zhang, Y., Li, L. (2010). Web Content Mining. Web Mining and Social Networking, 71–87. doi: https://doi.org/10.1007/978-1-4419-7735-9_4
36. Lytvyn, V., Vysotska, V., Pukach, P., Nytrebych, Z., Demkiv, I., Senyk, A. et. al. (2018). Analysis of the developed quantitative method for automatic attribution of scientific and technical text content written in Ukrainian. Eastern-European Journal of Enterprise Technologies, 6 (2 (96)), 19–31. doi: https://doi.org/10.15587/1729-4061.2018.149596
37. Gozhyj, A., Kalinina, I., Vysotska, V., Gozhyj, V. (2018). The Method of Web-Resources Management Under Conditions of Uncertainty Based on Fuzzy Logic. 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2018.8526761
38. Lytvyn, V., Vysotska, V., Dosyn, D., Burov, Y. (2018). Method for ontology content and structure optimization, provided by a weighted conceptual graph. Webology, 15 (2), 66–85.
39. Khomytska, I., Teslyuk, V. (2016). Specifics of phonostatistical structure of the scientific style in English style system. 2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2016.7589887
40. Khomytska, I., Teslyuk, V. (2016). The Method of Statistical Analysis of the Scientific, Colloquial, Belles-Lettres and Newspaper Styles on the Phonological Level. Advances in Intelligent Systems and Computing, 149–163. doi: https://doi.org/10.1007/978-3-319-45991-2_10
41. Nytrebych, Z. M., Malanchuk, O. M., Il'kiv, V. S., Pukach, P. Ya. (2017). Homogeneous problem with two-point conditions in time for some equations of mathematical physics. Azerbaijan Journal of Mathematics, 7 (2), 180–196.
42. Nytrebych, Z., Il'kiv, V., Pukach, P., Malanchuk, O. (2018). On nontrivial solutions of homogeneous Dirichlet problem for partial differential equations in a layer. Kragujevac Journal of Mathematics, 42 (2), 193–207. doi: https://doi.org/10.5937/kgjmath1802193n
43. Nytrebych, Z., Malanchuk, O., Il'kiv, V., Pukach, P. (2017). On the solvability of two-point in time problem for PDE. Italian Journal of Pure and Applied Mathematics, 38, 715–726.
44. Pukach, P. Ya., Kuzio, I. V., Nytrebych, Z. M., Ilkiv, V. S. (2017). Analytical methods for determining the effect of the dynamic process on the nonlinear flexural vibrations and the strength of compressed shaft. Naukovyi Visnyk Natsionalnoho Hirnychoho Universytetu, 5, 69–76.
45. Pukach, P. Y., Kuzio, I. V., Nytrebych, Z. M., Il'kiv, V. S. (2018). Asymptotic method for investigating resonant regimes of nonlinear bending vibrations of elastic shaft. Scientific Bulletin of National Mining University, 1, 68–73. doi: https://doi.org/10.29202/nvngu/2018-1/9
46. Nytrebych, Z., Ilkiv, V., Pukach, P., Malanchuk, O., Kohut, I., Senyk, A. (2019). Analytical method to study a mathematical model of wave processes under twopoint time conditions. Eastern-European Journal of Enterprise Technologies, 1 (7 (97)), 74–83. doi: https://doi.org/10.15587/1729-4061.2019.155148
47. Pukach, P., Il'kiv, V., Nytrebych, Z., Vovk, M., Pukach, P. (2017). On the Asymptotic Methods of the Mathematical Models of Strongly Nonlinear Physical Systems. Advances in Intelligent Systems and Computing, 421–433. doi: https://doi.org/10.1007/978-3-319-70581-1_30
48. Lavrenyuk, S. P., Pukach, P. Y. (2007). Mixed problem for a nonlinear hyperbolic equation in a domain unbounded with respect to space variables. Ukrainian Mathematical Journal, 59 (11), 1708–1718. doi: https://doi.org/10.1007/s11253-008-0020-0
49. Pukach, P. Y. (2016). Investigation of Bending Vibrations in Voigt–Kelvin Bars with Regard for Nonlinear Resistance Forces. Journal of Mathematical Sciences, 215 (1), 71–78. doi: https://doi.org/10.1007/s10958-016-2823-0

50. Pukach, P., Il'kiv, V., Nytrebych, Z., Vovk, M. (2017). On nonexistence of global in time solution for a mixed problem for a nonlinear evolution equation with memory generalizing the Voigt-Kelvin rheological model. Opuscula Mathematica, 37 (45), 735. doi: https://doi.org/10.7494/opmath.2017.37.5.735
51. Pukach, P. Y. (2012). On the unboundedness of a solution of the mixed problem for a nonlinear evolution equation at a finite time. Nonlinear Oscillations, 14 (3), 369–378. doi: https://doi.org/10.1007/s11072-012-0164-6
52. Pukach, P. Y. (2014). Qualitative Methods for the Investigation of a Mathematical Model of Nonlinear Vibrations of a Conveyer Belt. Journal of Mathematical Sciences, 198 (1), 31–38. doi: https://doi.org/10.1007/s10958-014-1770-x
53. Bezobrazov, S., Sachenko, A., Komar, M., Rubanau, V. (2016). The Methods of Artificial Intelligence for Malicious Applications Detection in Android OS. International Journal of Computing, 15 (3), 184–190.
54. Dunets, O., Wolff, C., Sachenko, A., Hladiy, G., Dobrotvor, I. (2017). Multi-agent system of IT project planning. 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS). doi: https://doi.org/10.1109/idaacs.2017.8095141
55. Lytvyn, V., Vysotska, V., Pukach, P., Nytrebych, Z., Demkiv, I., Kovalchuk, R., Huzyk, N. (2018). Development of the linguometric method for automatic identification of the author of text content based on statistical analysis of language diversity coefficients. Eastern-European Journal of Enterprise Technologies, 5 (2 (95)), 16–28. doi: https://doi.org/10.15587/1729-4061.2018.142451
56. Vysotska, V., Lytvyn, V., Burov, Y., Berezin, P., Emmerich, M., Basto Fernandes, V. (2019). Development of Information System for Textual Content Categorizing Based on Ontology. CEUR Workshop Proceedings, 53–70.
57. Vysotska, V., Lytvyn, V., Burov, Y., Gozhyj, A., Makara, S. (2018). The consolidated information web-resource about pharmacy networks in city. Proceedings of the 1st International Workshop on Informatics & Data-Driven Medicine (IDDM 2018), 2255, 239–255. Available at: http://ceur-ws.org/Vol-2255/paper22.pdf
58. Rusyn, B., Vysotska, V., Pohreliuk, L. (2018). Model and Architecture for Virtual Library Information System. 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2018.8526679
59. Lytvyn, V., Vysotska, V., Dosyn, D., Lozynska, O., Oborska, O. (2018). Methods of Building Intelligent Decision Support Systems Based on Adaptive Ontology. 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). doi: https://doi.org/10.1109/dsmp.2018.8478500
60. Lytvyn, V., Vysotska, V., Burov, Y., Bobyk, I., Ohirko, O. (2018). The Linguometric Approach for Co-authoring Author's Style Definition. 2018 IEEE 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS). doi: https://doi.org/10.1109/idaacs-sws.2018.8525741
61. Zdebskyi, P., Vysotska, V., Peleshchak, R., Peleshchak, I., Demchuk, A., Krylyshyn, M. (2019). An Application Development for Recognizing of View in Order to Control the Mouse Pointer. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 55–74.
62. Veres, O., Rusyn, B., Sachenko, A., Rishnyak, I. (2018). Choosing the method of finding similar images in the reverse search system. Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Systems. Volume I: Main Conference (COLINS 2018), 2136, 99–107.
63. Rashkevych, Y., Peleshko, D., Vynokurova, O., Izonin, I., Lotoshynska, N. (2017). Single-frame image super-resolution based on singular square matrix operator. 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). doi: https://doi.org/10.1109/ukrcon.2017.8100390
64. Vysotska, V., Lytvyn, V., Hrendus, M., Kubinska, S., Brodyak, O. (2018). Method of Textual Information Authorship Analysis Based on Stylometry. 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2018.8526608
65. Gozhyj, A., Chyrun, L., Kowalska-Styczen, A., Lozynska, O. (2018). Uniform Method of Operative Content Management in Web Systems. Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Systems. Volume I: Main Conference (COLINS 2018), 2136. P. 62–77. Available at: http://ceur-ws.org/Vol-2136/10000062.pdf
66. Vysotska, V., Burov, Y., Lytvyn, V., Demchuk, A. (2018). Defining Author's Style for Plagiarism Detection in Academic Environment. 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), 128–133. doi: https://doi.org/10.1109/dsmp.2018.8478574
67. Chyrun, L., Vysotska, V., Kis, I., Chyrun, L. (2018). Content Analysis Method for Cut Formation of Human Psychological State. 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). doi: https://doi.org/10.1109/dsmp.2018.8478619
68. Gozhyj, A., Vysotska, V., Yevseyeva, I., Kalinina, I., Gozhyj, V. (2018). Web Resources Management Method Based on Intelligent Technologies. Advances in Intelligent Systems and Computing III, 206–221. doi: https://doi.org/10.1007/978-3-030-01069-0_15
69. Chyrun, L., Kis, I., Vysotska, V., Chyrun, L. (2018). Content Monitoring Method for Cut Formation of Person Psychological State in Social Scoring. 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). doi: https://doi.org/10.1109/stc-csit.2018.8526624
70. Demchuk, A., Lytvyn, V., Vysotska, V., Dilai, M. (2019). Methods and Means of Web Content Personalization for Commercial Information Products Distribution. Lecture Notes in Computational Intelligence and Decision Making, 332–347. doi: https://doi.org/10.1007/978-3-030-26474-1_24

71. Lytvyn, V., Vysotska, V., Kuchkovskiy, V., Bobyk, I., Malanchuk, O., Ryshkovets, Y. et. al. (2019). Development of the system to integrate and generate content considering the cryptocurrent needs of users. Eastern-European Journal of Enterprise Technologies, 1 (2 (97)), 18–39. doi: https://doi.org/10.15587/1729-4061.2019.154709
72. Vysotska, V., Fernandes, V. B., Lytvyn, V., Emmerich, M., Hrendus, M. (2018). Method for Determining Linguometric Coefficient Dynamics of Ukrainian Text Content Authorship. Advances in Intelligent Systems and Computing III, 132–151. doi: https://doi.org/10.1007/978-3-030-01069-0_10
73. Kravets, P. (2010). The control agent with fuzzy logic. Perspective Technologies and Methods in MEMS Design, 40–41.
74. Kravets, P. (2007). The Game Method for Orthonormal Systems Construction. 2007 9th International Conference - The Experience of Designing and Applications of CAD Systems in Microelectronics. doi: https://doi.org/10.1109/cadsm.2007.4297555
75. Kravets, P. (2016). Game Model of Dragonfly Animat Self-Learning. Perspective Technologies and Methods in MEMS Design, 195–201.
76. Bazylyk, O., Taradaha, P., Nadobko, O., Chyrun, L., Shestakevych, T. (2012). The results of software complex OPTAN use for modeling and optimization of standard engineering processes of printed circuit boards manufacturing. 2012 11th International Conference on «Modern Problems of Radio Engineering, Telecommunications and Computer Science» (TCSET), 107–108.
77. Bondariev, A., Kiselychnyk, M., Nadobko, O., Nedostup, L., Chyrun, L., Shestakevych, T. (2012). The software complex development for modeling and optimizing of processes of radio-engineering equipment quality providing at the stage of manufacture. TCSET'2012, 159.
78. Teslyuk, V., Beregovskyi, V., Denysyuk, P., Teslyuk, T., Lozynskyi, A. (2018). Development and Implementation of the Technical Accident Prevention Subsystem for the Smart Home System. International Journal of Intelligent Systems and Applications, 10 (1), 1–8. doi: https://doi.org/10.5815/ijisa.2018.01.01
79. Basyuk, T. (2015). The main reasons of attendance falling of internet resource. 2015 Xth International Scientific and Technical Conference "Computer Sciences and Information Technologies" (CSIT). doi: https://doi.org/10.1109/stc-csit.2015.7325440
80. Chernukha, O., Bilushchak, Y. (2016). Mathematical modeling of random concentration field and its second moments in a semispace with erlangian disrtibution of layered inclusions. Task Quarterly, 20 (3), 295–334.
81. Chyrun, L., Kowalska-Styczen, A., Burov, Y., Berko, A., Vasevych, A., Pelekh, I., Ryshkovets, Y. (2019). Heterogeneous Data with Agreed Content Aggregation System Development. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 35–54.
82. Chyrun, L., Burov, Y., Rusyn, B., Pohreliuk, L., Oleshek, O. et. al. (2019). Web Resource Changes Monitoring System Development. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 255–273.
83. Vysotska, V., Burov, Y., Lytvyn, V., Oleshek, O. (2019). Automated Monitoring of Changes in Web Resources. Lecture Notes in Computational Intelligence and Decision Making, 348–363. doi: https://doi.org/10.1007/978-3-030-26474-1_25
84. Chyrun, L., Gozhyj, A., Yevseyeva, I., Dosyn, D., Tyhonov, V., Zakharchuk, M. (2019). Web Content Monitoring System Development. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Systems (COLINS-2019). Volume I: Main Conference, 2362, 126–142.
85. Rzheuskyi, A., Gozhyj, A., Stefanchuk, A., Oborska, O., Chyrun, L., Lozynska, O. et. al. (2019). Development of Mobile Application for Choreographic Productions Creation and Visualization. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 340–358.
86. Lytvynenko, V., Savina, N., Krejci, J., Voronenko, M., Yakobchuk, M., Kryvoruchko, O. (2019). Bayesian Networks' Development Based on Noisy-MAX Nodes for Modeling Investment Processes in Transport. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 1–10.
87. Lytvynenko, V., Lurie, I., Krejci, J., Voronenko, M., Savina, N., Taif, M. A. (2019). Two Step Density-Based Object-Inductive Clustering Algorithm. Workshop Proceedings of the 8th International Conference on "Mathematics. Information Technologies. Education", 2386, 117–135.