|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | | |
|  | Name of the student: | | Yash Sankpal | | | | Roll No. | | 8695 | |  |
| Practical Number: | | 8 | | | | Date of Practical: | |  | |
| Relevant CO’s | | At the end of the course students will be able to apply appropriate algorithms for extracting knowledge from given dataset. | | | | | | | |
| Sign here to indicate that you have read all the relevant material provided Sign:  before attempting this practical  Practical grading using Rubrics | | | | | | | | | |
| Indicator | Very Poor | | Poor | Average | Good | | Excellent | |  |
| Timeline  (2) | More than a  session late  (0) | | NA | NA | NA | | Early or on  time (2) | |
| Code de-  sign (2) | N/A | | Very poor  code design  with no  comments and indenta-  tion(0.5) | Poor code  design with  very comments and  indentation  (1) | Design with good coding standards  (1.5) | | Accurate design  with better coding  satndards (2) | |
| Performance  (4) | Unable to  perform the experiment  (0) | | Able to  partially  perform the experiment  (1) | Able to  perform the experiment for certain use cases (2) | Able to  perform the experiment considering  most of the  use cases (3) | | Able to  perform the experiment considering all use cases  (4) | |
| Postlab (2) | Incorrect answer(0) | | N/A | Partially cor-  rect answer  (1) | N/A | | Fully correct answer (2) | |
| |  |  | | --- | --- | | Total Marks (10) | Sign of instructor with date | |  |  | | | | | | | | | |

|  |
| --- |
| Practical  Course title: Big Data Analytics  Course term: 2021-2022  Instructor name: Saurabh Kulkarni  Problem Statement: To demonstrate use of recommendation system for movie rating prediction  Theory: |

|  |
| --- |
| Code:  Write R code for recommendation system for given input code for the problem: |
| library(lsa)  critics = read.csv("Movieratings.csv")  #calculate the euclidian distance  #EUD = dist(critics[,2:7])  #cosine similarity calculation  x = critics[,2:7]  x[is.na(x)] = 0  user\_sim = cosine(as.matrix(t(x))) #user similarity  #Recommending items  #for Toby  #create weightge matrix  weight\_mat = user\_sim[,7]\*critics[,2:7]  rec\_itm\_for\_user = function(userNo)  {  #calcualte column wise sum  col\_sums= list()  rat\_user = critics[userNo,2:ncol(critics)]  x=1  tot = list()  z=1  for(i in 1:ncol(rat\_user)){  if(is.na(rat\_user[1,i]))  {    col\_sums[x] = sum(weight\_mat[,i],na.rm=TRUE)  x=x+1    temp = as.data.frame(weight\_mat[,i])    sum\_temp=0    for(j in 1:nrow(temp)){  if(!is.na(temp[j,1])){  sum\_temp = sum\_temp+user\_sim[j,ncol(rat\_user)]  }    }  tot[z] = sum\_temp  z=z+1  }      }  z=NULL  z=1  for(i in 1:ncol(rat\_user)){  if(is.na(rat\_user[1,i]))  {  rat\_user[1,i] = col\_sums[[z]]/tot[[z]]  z=z+1  }    }  return(rat\_user)  }  #to get N recommendations:  rec\_itm\_for\_user(6) #first person recommendations |
|  |

PostLab:

Explain Content based recommendation systems

Answer for postlab question

Another common approach when designing recommender systems is **content-based filtering**. Content-based filtering methods are based on a description of the item and a profile of the user's preferences.[[44]](https://en.wikipedia.org/wiki/Recommender_system#cite_note-Aggarwal16Book-44)[[45]](https://en.wikipedia.org/wiki/Recommender_system#cite_note-45) These methods are best suited to situations where there is known data on an item (name, location, description, etc.), but not on the user. Content-based recommenders treat recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on an item's features.

In this system, keywords are used to describe the items, and a [user profile](https://en.wikipedia.org/wiki/User_profile) is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items similar to those that a user liked in the past or is examining in the present. It does not rely on a user sign-in mechanism to generate this often temporary profile. In particular, various candidate items are compared with items previously rated by the user, and the best-matching items are recommended. This approach has its roots in [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval) and [information filtering](https://en.wikipedia.org/wiki/Information_filtering) research.

To create a [user profile](https://en.wikipedia.org/wiki/User_profile), the system mostly focuses on two types of information:

1. A model of the user's preference.

2. A history of the user's interaction with the recommender system.

Basically, these methods use an item profile (i.e., a set of discrete attributes and features) characterizing the item within the system. To abstract the features of the items in the system, an item presentation algorithm is applied. A widely used algorithm is the [tf–idf](https://en.wikipedia.org/wiki/Tf%E2%80%93idf" \o "Tf–idf) representation (also called vector space representation).[[46]](https://en.wikipedia.org/wiki/Recommender_system#cite_note-46) The system creates a content-based profile of users based on a weighted vector of item features. The weights denote the importance of each feature to the user and can be computed from individually rated content vectors using a variety of techniques. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as [Bayesian Classifiers](https://en.wikipedia.org/wiki/Naive_Bayes_classifier), [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis), [decision trees](https://en.wikipedia.org/wiki/Decision_trees), and [artificial neural networks](https://en.wikipedia.org/wiki/Artificial_neural_networks) in order to estimate the probability that the user is going to like the item.[[47]](https://en.wikipedia.org/wiki/Recommender_system#cite_note-47)

A key issue with content-based filtering is whether the system can learn user preferences from users' actions regarding one content source and use them across other content types. When the system is limited to recommending content of the same type as the user is already using, the value from the recommendation system is significantly less than when other content types from other services can be recommended. For example, recommending news articles based on news browsing is useful. Still, it would be much more useful when music, videos, products, discussions, etc., from different services, can be recommended based on news browsing. To overcome this, most content-based recommender systems now use some form of the hybrid system.

Content-based recommender systems can also include opinion-based recommender systems. In some cases, users are allowed to leave text reviews or feedback on the items. These user-generated texts are implicit data for the recommender system because they are potentially rich resources of both feature/aspects of the item and users' evaluation/sentiment to the item. Features extracted from the user-generated reviews are improved [meta-data](https://en.wikipedia.org/wiki/Metadata) of items, because as they also reflect aspects of the item like [meta-data](https://en.wikipedia.org/wiki/Metadata), extracted features are widely concerned by the users. Sentiments extracted from the reviews can be seen as users' rating scores on the corresponding features. Popular approaches of opinion-based recommender system utilize various techniques including [text mining](https://en.wikipedia.org/wiki/Text_mining), [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval), [sentiment analysis](https://en.wikipedia.org/wiki/Sentiment_analysis) (see also [Multimodal sentiment analysis](https://en.wikipedia.org/wiki/Multimodal_sentiment_analysis)) and deep learning.

|  |
| --- |
| Explain Collaborative filtering systems  Answer for postlab question |