THOMSON REUTERS ENDNOTE

[Tomar* *et al.*, 5(7): July, 2016] ISSN: 2277-9655
IC™ Value: 3.00 Impact Factor: 4.116

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

AN IMPLEMENTATION OF PEARSON CORRELATION METHOD FOR PREDICTING ITEMS TO USER IN E-COMMERCE

Arjun Singh Tomar*

* Dept. of Computer Science & Engineering JAYPEE University of Engineering & Technology Guna, India

DOI: 10.5281/zenodo.57944

ABSTRACT

Collaborative filtering algorithms (CFAs) are recommender systems for the collaborating one another to filter documents they read from last decade. CFAs have various features that create them different from other algorithms. Algorithm of user-based collaborative filtering is one of filtering algorithms, known for their effectiveness and simplicity. In the present paper, pearson calculation is applied which works on user-based data and finds out the similarity measure of the products and then recommend them according to the similarity calculated. An application is built to perform this research work whose web pages have been attached as a part of results calculated. The model is an improved model working well on collaborative technique and recommending items to the users. Along with this few add-on implementations have been done so as to improve the functionality of the application.

KEYWORDS:Data mining, Data Filtration, Collaborative Filtering, Pearson correlation, Recommender System.

INTRODUCTION

In our daily life, virtually all of us have asked a trusted friend to recommend a movie or a restaurant. The underlying assumption is that our friend shares our taste, and if she recommends an item, we are likely to enjoy it. If a friend consistently provides good recommendations, she becomes more trusted, but if she provides poor recommendations, she becomes less trusted and eventually ceases to be an advisor.

Collaborative filtering (CF) describes a variety of processes that automate the interactions of human advisors; a collaborative filter recommends items based upon the opinions of a clique of human advisors. Amazon.com and CDNow.com are two well-known e-commerce sites that use collaborative filtering to provide recommendations on books, music and movie titles; this service is provided as a means to advance client maintenance, steadfastness and deals, and so forth [13].

Some Important Cfas With Prediction

Memory based CFA

User rating data is utilized to compute similarity between users or things. This is utilized for making suggestions [1] and is utilized as a part of numerous business frameworks. It is anything but difficult to implement and is successful. Some critical algorithms are: User-based collaborative filtering, Item-based collaborative filtering and Similarity fusion collaborative filtering. Memory-based CFAs utilize the whole or an example of the user-item database to create a prediction. By identifying the neighbors of a new user, a prediction of preferences on new items for him or her can be produced. At the point when the task is to create a top-N proposal, we have to locate the most comparable users or items in the wake of figuring the similitudes, and afterward total the neighbors to get the top-N most continuous things as the suggestion.

Model based CFA

These are utilized to make predictions for real data. To discover designs taking into training data models are created utilizing data mining and machine learning algorithms [1]. Some important algorithms are: Bayesian belief nets collaborative filtering, Regression based collaborative filtering, Slope one collaborative filtering, Latent Semantic Indexing collaborative filtering and Cluster based smoothing collaborative filtering The design and development of models can allow the system to learn to perceive complex examples. At that point make intelligent predictions for the collaborative filtering errands for real-world data. The aforementioned, Model-based CF algorithms have been examined to solve the drawbacks of memory-based CF algorithms.

Hybrid CFA

Various applications join the memory-based and the model-based CF algorithms. These overcome the limitations of local CF approaches [13]. It enhances the forecast execution and conquers the CF issues, for example, sparsity and loss of data. A hybrid CF otherwise called Con tent Boosted Collaborative Filtering, methodology was proposed to exploit bulk data intended for careful product grouping to address the data sparsity issue of CF suggestions. Some critical algorithm is: Personality conclusion collaborative filtering Hybrid CFAs join CF with other suggestion methods to make forecasts. Content-based CF makes suggestions by breaking down the substance of printed data. Numerous components add to the significance of the textual content. A content- based recommender then uses classification algorithms to make suggestions.

Content based methods have the cold-start issue, in which they should have enough data to assemble a solid classifier.

They are restricted by the components explicitly connected with the items they prescribe, while collaborative filtering can make suggestions with no descriptive data. Content based techniques only recommend items that score highly against a user's profile. Other recommender systems includes demographic information i.e. gender, postcode, occupation etc,. Utility-based recommender systems and knowledge-based recommender systems are the systems which require knowledge about how a particular object satisfies the user needs.

Pearson Corelation Techniques

In insights, the **Pearson product-moment correlation coefficient** (here and there alluded to as the **PPMCC** or **PCC** or **Pearson's *r***) is a measure of the direct connection between's two variables X and Y, giving a worth amongst +1 and −1 comprehensive, where 1 is all out positive correlation, 0 is no correlation, and −1 is complete negative correlation. It is generally utilized as a part of the sciences as a measure of the level of direct reliance between two variables. It was produced by Karl Pearson from a related thought presented by Francis Galton in the 1880s.

Pearson's connection coefficient is the covariance of the two variables isolated by the product of their standard deviations. The type of the definition includes an "product moment", that is, the mean (the main minute about the root) of the result of the mean-balanced irregular variables; subsequently the modifier *product-moment* in the name.

Pearson's r The Pearson product-moment connection coefficient, also called the relationship coefficient, or as r, is the most generally utilized connection coeffi-cient. Values of r for pairs of variables are commonly reported in research reports and journals as a means of summarizing the extent of the relationship between two variables. Pearson's r condenses the relationship between two variables that have a straight line or direct association with each other. On the off chance that the two variables have a straight line relationship in the positive bearing, then r will be certain and extensively above 0. On the off chance that the straight relationship is in the negative course, so that expansions in one variable, are connected with abatements in the other, then r < 0. The possible values of r range from -1 to +1, with values close to 0 signifying little relationship between the two variables. Precisely how not the same as 0 the estimation of r must be before giving confirmation of a relationship can be resolved on the premise of a speculation test. It will likewise be found in Section 11.4.7 that the measure of r can contrast rather impressively contingent upon what sort of data is being inspected. The Pearson connection coefficient r can be characterized as takes after. Assume that there are two variables X and Y , each having n values X1, X2, . …, Xn and Y1, Y2, . . . , Yn respectively. Let the mean of X be X¯ and the mean of Y be Y¯ . Pearson's r is:

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}$$

where the summation proceeds across all n possible values of X and Y . A method of computing r is presented next, with an example. Following this, there is some discourse of the significance and interpretation of the connection coefficient.

RELATED WORK
Two of the first automated collaborative filtering systems use Pearson correlation to identify similarities between users of Usenet[11] and music album aficionados[14]. In [14], the constrained Pearson correlation is introduced to account for the implicit positivity and negativity of a rating scale. Ringo also provides an innovative solution that inverts the basic CF approach; music albums are treated as 'participants' that can recommend users to other music album participants.

When the rating density is low, most CF systems have difficulty generating accurate recommendations [11] [5]. Unlike the problem of scalability, however, rating sparsity is an open issue that has received significant research attention. [12] and [5] attempt to ameliorate this issue by using bots and agents to artificially increase the rating density. Bots assign ratings based on criteria such as the number of spelling errors, the length of the Usenet message, the existence of included messages [12] or the genre of the movie title [5]. Agents are trained, using IF techniques, to mimic the rating distribution of each user. An agent regenerates its ratings as it becomes better trained which may force large portions of the similarity matrix to be updated [5]. In both of these works, the relevancy of the bots' and agents' ratings to a particular user is decided by the CF system as it identifies potential advisors.

In their recent paper, Goldberg et al. [4] describe Eigentaste, which for certain domains does not suffer from the sparsity and scalability problems. They note that rating sparsity is introduced during the profiling stage when users are given the freedom to select the items they rate. In contrast, the Eigentaste algorithm forces participants to rate all items in a *gauge set*. The dimensionality of the resulting dense rating matrix is reduced using principal component analysis to the first two dimensions. All of the users are then projected onto this eigen-plane and a divisive clustering algorithm is applied to partition the users into neighbourhoods. When a new user joins the system their neighbourhood is located by projecting their responses to the gauge set onto the eigen-plane. A recommendation is generated by taking neighbourhood's average rating for an item. Eigentaste is a linear collaborative filter and requires O(*j2n*) time to compute its cluster structure. For small values of *j*, the size of the gauge set, Eigentaste can be very fast.

Eigentaste is however limited in that it requires the definition of a gauge set. In the Jester recommendation service, the gauge set consists of a set of jokes. After reading a joke, each user can immediately supply a rating. However, there are few domains where the items of interest can be consumed so quickly and evaluated[19,20]. It is worth noting that many e-commerce sites provide a simplified form of collaborative filtering that is based on the complementary technologies of data warehousing and on-line analytical processing (OLAP). Often-seen examples of OLAP style collaborative filtering are the factoids that attempt to cross-sell/up-sell products: *Item X has been downloaded Z times*. These rudimentary filters make the implicit assumption that all users are equally good advisors to the active user. A more sophisticated approach would be to mine patterns from the database and data warehouse [21] and to use these as the basis of a recommendation to the user.

PEARSON CORELATION TECHNIQUES
Web application for recommendation using pearson r formula is proposed and created.
The website created along with the recommendation of the products is shown. The proposed code can be explained as below:

1. Create a class named PearsonCalculation and initialize int pid=0,uid=0,rating=0;
   int usernum=1,productnum=1;
2. Applying sql queries to retrieve count(pid) and count(uid).
3. Calculate average rating of the items using avgrating(i).
4. Now for similarity calculation applying the similarity calculation formuls:
   a. x=Math.abs(arr[i][j]-avgrating[i]);
   b. w[i]=(sumx[i]+sumx[j])/(Math.sqrt(sumx_sq[i])*Math.sqrt(sumx_sq[j]));
5. Displaying the similar products of the users as that of the products that can be similarly recommended to the user.

6. End.

Datasets

The dataset is synthetically prepared and a web based online products recommendation system is made. It contains items of various types. This data has multiple users collected with a rating from 0 to 5. We used correlation formula to show the recommendations to the users in their accounts.

IMPLEMENTATION

The following jsp programs are made for the application developed:

1. adminhome.jsp

Sql query- ResultSet rs1=stmt1.executeQuery("SELECT orderdetail.pcode, product.product_name, product.price FROM orderdetail INNER JOIN product ON orderdetail.pCode = product.product_id where orderdetail.order_no="+ono);

2. Single.jsp

To show the products, their availability, price , description, ratings, reviews and add to cart option.

String query="select * from product where product_id='"+pid+"'";

2. recommendation.jsp

Pearson calculation

```
int pid=0,uid=0,rating=0;
    int usernum=1,productnum=1;

    String q="select count(pid) as pid from product";
    ResultSet rs=st.executeQuery(q);
    if (rs.next())
    {
      productnum=rs.getInt(1);
    }
    String q1="select count(uid) as uid from registration";
    ResultSet rs1=st1.executeQuery(q1);
    if (rs1.next())
    {
      usernum=rs1.getInt(1);
     }
      int uname[]=new int[usernum];
      String q2="select uid from registration order by uid asc";
      ResultSet rs3=st2.executeQuery(q2);
      int i=0,j=0,k=0;
      while (rs3.next())
      {
          uname[i]=rs3.getInt(1);
           i++;
       }
      int arr[][]=new int[usernum][productnum];
      double avgrating[]=new double[usernum];
      double w[]=new double[usernum];
     for (i=0;i<usernum;i++)
     {
           String query="select * from reviews where uid="+uname[i]+" order by pid asc";
      ResultSet rs2=st3.executeQuery(query);
```

```
                    while(rs2.next())
                {
                        uid=Integer.parseInt(rs2.getString(1));
                        pid=Integer.parseInt(rs2.getString(2));
                        rating=Integer.parseInt(rs2.getString(3));
                                arr[i][pid-1]=rating;
                        avgrating[i]=avgrating[i]+rating;
                              }
                              avgrating[i]=avgrating[i]/productnum;
            }
           double sumx[]=new double[usernum];
           double sumx_sq[]=new double[usernum];
                  double x=0;
                   for (i=0;i<usernum;i++)
                       {
                                sumx[i]=0;
                                sumx_sq[i]=0;
                                for (j=0;j<productnum;j++)
                                {
                                          if (arr[i][j] !=0)
                                          {
  x=Math.abs(arr[i][j]-avgrating[i]);
   sumx[i]=sumx[i]+x;
   sumx_sq[i]=sumx_sq[i]+x*x;
          }
   }
   }

   i=require-1;
  for(j=0;j<usernum;j++)
   {
       if (i!=j)
      {
  w[i]=(sumx[i]+sumx[j])/(Math.sqrt(sumx_sq[i])*Math.sqrt(sumx_sq[j]));
  if (w[i]>=0.8)
 {
        String query="SELECT distinct product.product_id, product_name,description,product.price,size,pphoto1,
product.pid FROM product,orderdetail,orders where product.product_id=orderdetail.pcode and
orders.order_no=orderdetail.order_no and orders.uid="+(j+1);
ResultSet rs4=stmt.executeQuery(query);
while(rs4.next())
{
 pname=rs4.getString(2);
description=rs4.getString(3);
 price=rs4.getString(4);
 size=rs4.getString(5);
pphoto1=rs4.getString(6);
pphoto="./images/"+pphoto1;
   %>
 <tr>
  <td> <img src="<%=pphoto %>" height="150" width="250" alt=""/></td>
<td> <%=pname %></td>
<td> <%=description %></td>
```

ISSN: 2277-9655
Impact Factor: 4.116

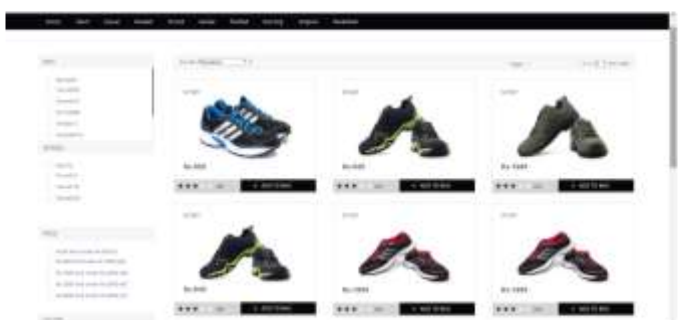
```
<td> <%=price %></td>
<td> <%=size %></td>
 <%
  }}}}
  %>
```

RESULT ANALYSIS

The results are shown in a sequential manner from the home page to login page to recommendation list to the pearson output on the console. Starting from the home page, the results are shown below:

1. Home page

2. Products page



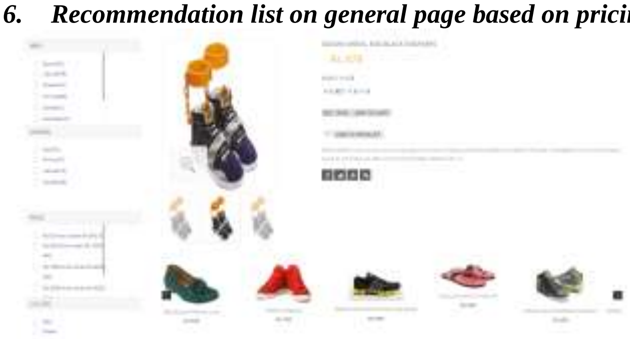
3. Registration page

ISSN: 2277-9655
Impact Factor: 4.116

4. Login page

5. On selecting an item

6. Recommendation list on general page based on pricing



7. Add to cart

ISSN: 2277-9655
Impact Factor: 4.116

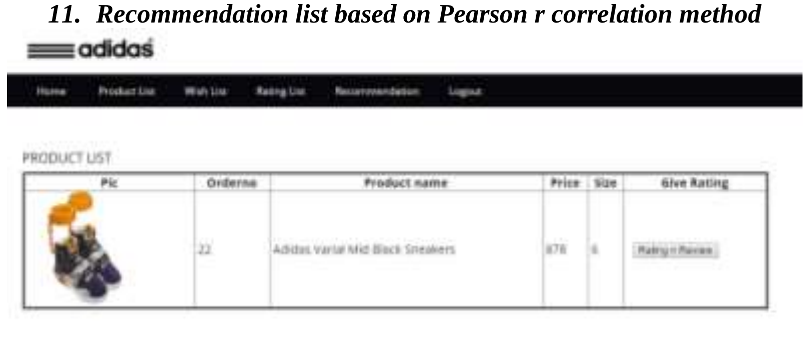
8. Confirming shipping address

9. Proceeding to payment and item confirmed

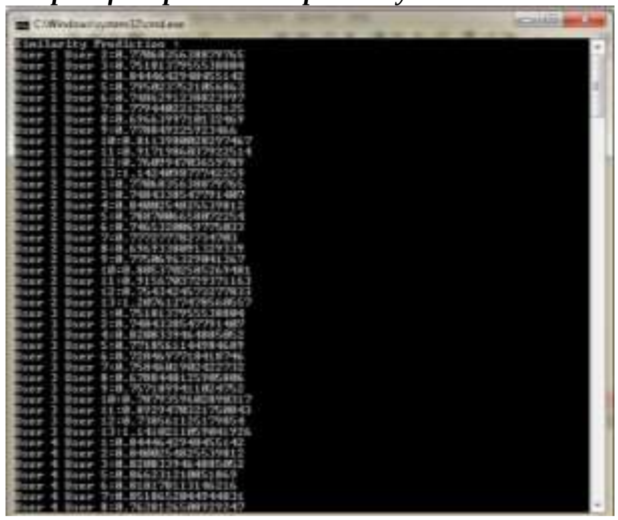
<u>*User Account Details*</u>

10. Product purchased

11. Recommendation list based on Pearson r correlation method



12. Console output of the pearson comparability between the client based items



CONCLUSION

In this paper, user based collaborative filtering algorithm has been implemented and evaluated. The quality of the predictions is evaluated with similar algorithms. It shows that it exhibits a behavior that is equivalent to that of the best algorithms. The main aim of this paper is to improve the quality of the results and make it easier for the user to find relevant information from the internet.

REFERENCES

[1] Xiaoyuan Su and Taghi M. Khoshgoftaar, “Review Article a Survey of Collaborative Filtering Techniques” published in Hindawi Publishing Corporation, Advances in Artificial Intelligence Volume 2009, Article ID 421425, 19 pages, doi:10.1155/2009/421425

[2] R. Bell and Y. Koren, “Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights”, IEEE International Conference on Data Mining (ICDM’07), pp. 43–52, 2007.

[3] Manos Papagelis and Dimitris Plexousakis “Qualitative analysis of userbased and item-based prediction algorithms for recommendation agents” Engineering Applications of Artificial Intelligence 18 (2005) 781–789 www.elsevier.com/locate/engappai

[4] R. Salakhutdinov, A. Mnih and G. Hinton, “Restricted Boltzmann Machines for Collaborative Filtering”, Proc. 24th Annual International Conference on Machine Learning, pp. 791–798, 2007.

[5] Bhaskar Mehta, Thomas Hofmann, and Peter Fankhauser, Lies and propaganda: detecting spam users in collaborative filtering, IUI ’07: Proceedings of the 12th international conference on Intelligent user interfaces (New York, NY, USA), ACM Press, 2007, pp. 14–21.

[6] R. Salakhutdinov and A. Mnih, “Probabilistic Matrix Factorization”, Advances in Neural Information Processing Systems 20 (NIPS’07), pp. 1257–1264, 2008.

[7] Bhaskar Mehta and Wolfgang Nejdl, Attack-resistant Collaborative Filtering, To appear: In Proceedings of the 31st ACM SIGIR Conference, ACM Press, 2008.

[8] J. Sandvig, Bamshad Mobasher, and Robin Burke, Robustness of collaborative recommendation basedon association rule mining, RecSys ’07: Proceedings of the 2007 ACM conference on Recommender systems (New York, NY, USA), ACM, 2007, pp. 105–112.

[9] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” *Communications of ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[10] Jester data, http://www.grouplens.org/. X. Su, T. M. Khoshgoftaar, and R. Greiner, “A mixtureImputationboosted collaborative filter,” in Proceedings ofthe 21th International Florida Artificial Intelligence Research Society Conference (FLAIRS ’08), pp. 312–317, Coconut Grove, Fla, USA, May 2008.

[11] Marko Balabanovic and Yoav Shoham. FAB: Content-based collaborative recommendation. *Communications of the ACM*, 40(3), March 1997.

[12] Chumki Basu, Haym Hirsh, and William Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the 1998 Workshop on Recommender Systems*, pages 11–15. AAAI Press, 1998.

[13] Doug Beeferman and Adam Berger. Agglomerative clustering of a search engine query log. In *Proceedings of ACM SIGKDD International Conference*, pages 407–415, 2000.

[14] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proceedings of ICML*, pages 46–53, 1998.

[15] P. Chan. A non-invasive learning approach to building web user profiles. In *Proceedings of ACM SIGKDD International Conference*, 1999.

[16] Adamopoulos, Panagiotis; Tuzhilin, Alexander (January 2015). "On Unexpectedness in Recommender Systems: Or How to Better Expect the Unexpected". ACM Transactions on Intelligent Systems and Technology (TIST). doi:10.1145/2559952.

[17] Adamopoulos, Panagiotis (October 2013). "Beyond rating prediction accuracy: on new perspectives in recommender systems". Proceedings of the 7th ACM conference on Recommender systems, doi:10.1145/2507157. 2508073.

[18] "Robust collaborative filtering". Portal.acm.org. 19 October 2007. doi:10.1145/1297231.1297240. Retrieved 2012-05-15.