

CALIFORNIA STATE UNIVERSITY SAN MARCOS

PROJECT SIGNATURE PAGE

PROJECT SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

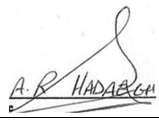
PROJECT TITLE: PRODUCT RECOMMENDATION SYSTEM USING MACHINE LEARNING
TECHNIQUES

AUTHOR: MUKHUL KANAGALA

DATE OF SUCCESSFUL DEFENSE: 12/10/2020

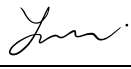
THE PROJECT HAS BEEN ACCEPTED BY THE PROJECT COMMITTEE IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF
SCIENCE IN COMPUTER SCIENCE

Dr. Ahmad R. Hadaegh
PROJECT COMMITTEE CHAIR


SIGNATURE

12/16/2020
DATE

Dr. Yanyan Li
PROJECT COMMITTEE MEMBER


SIGNATURE

12/16/2020
DATE

Name of Committee Member
PROJECT COMMITTEE MEMBER

SIGNATURE

DATE

PRODUCT BASED
RECOMMENDATION SYSTEM USING
MACHINE LEARNING TECHNIQUES.

Name: MUKHUL KANAGALA

Defense Date:12/10/2020

Table of Contents

1.Abstract.....	3
2.Introduction:.....	4
3.Architecture:.....	6
4.Implementation:	10
5.Conclusion:	18
6 References:	19
7 Appendix:	222

1.Abstract

In our day-to-day life everyone of us happen to use the e-commerce websites to shop many items. This leads to an increasing diversity of consumers' demand, turning into a challenge for a retail store to provide the right products accordingly to customer preferences. These e-commerce websites use different types of recommendation systems to provide a positive shopping experience to the user. Recommender systems are a tool to cope with this challenge, through product recommendation it is possible to fulfill customers' needs and expectations, helping maintaining loyal customers while attracting new customers. In this project we have used the ratings given to a product by other users to make recommendations to the current user. We have created a matrix to represent the ratings given to the product by the different users. Next we have created the user profile and item profile and provide recommendations depending on the similarity to the other groups. Then we apply KNN algorithm which is a part of machine learning and Pearson's Correlation Coefficient to make recommendations to the user. To evaluate the performance of the model, RMSE was used which gives information on the closeness of recommendations being generated.

2.Introduction:

The history of recommendation system date back to early 90's. Ever since then the recommendation systems have been evolving continuously. These recommendation systems have become the integral part of the web applications. This is mainly because the e-commerce websites have a large variety of product inventory's available the data which is to be displayed to the user can be overwhelming. As a result the user may face difficulties to search for the product that they maybe looking for. At this point the recommendation system's comes to usage. Harvard Business Review said that these recommendation algorithms are the key difference between "born digital" enterprises and legacy companies. This is mainly because the recommendation engine can provide personalized suggestion of products to the users depending on their browsing patterns and previous purchase history. As it provides these personalized recommendations users will be much interested to come back and do more purchases in the interactive environment provided by our system. With more number of visits from the user to our website, we can be able to collect more data about the user and the products, which in turn provides a chance to know the areas of improvements of the product. The better quality of product we have there is a much likely chance of acquiring more number of users. This is a whole continuous cyclic process which can be seen in the figure below[1].The recommendation system helps to increase the sales of items related to a particular product too. For example if a user buys a monitor from the website, our recommendation system will suggest him different types of keyboards and mouse. In this way it helps in increasing the

sales of certain items which the user by himself may not be searching. This recommendation system reduces the load on the database as they provide a personal recommendation to every individual users instead of displaying whole inventory. These recommendation systems usually are based on two different filtering techniques namely Collaborative filtering and Content based filtering. content based filtering is based on the content i.e. item-item relationship. In this method we form a relationship or cluster between group of items and display the relevant items to the customer when the customer searches for one item from the cluster. Collaborative filtering is based on the user behavior .Depending on the use behavior we do the further recommendations. In Collaborative filtering we form a cluster among different customers depending on their previous purchases and history. From this we display the similar products to remaining users in the cluster.

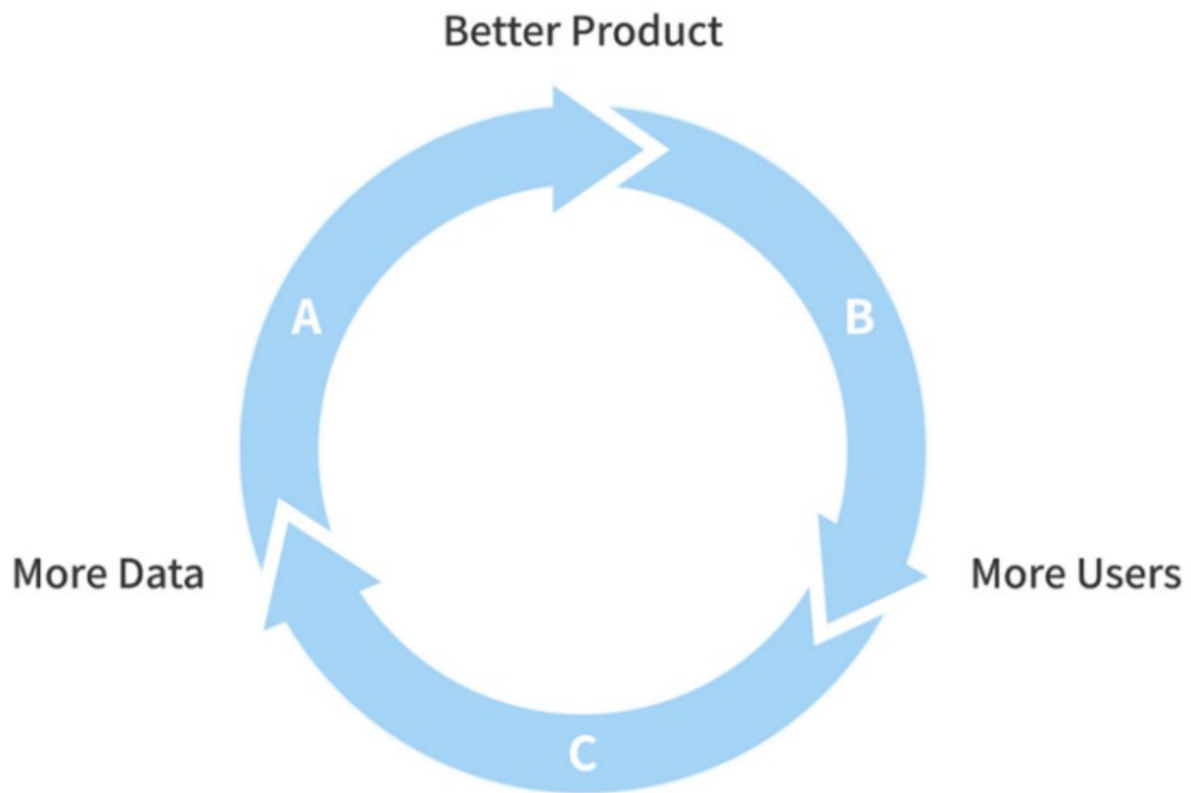


Fig.1 Overview of use of a recommendation system.

3. Architecture:

There are multiple steps involved in the architecture of the recommender system.

3.1 Candidate Generation

In candidate generation we generate a small subset of products for the use from the large size of products available. To get this small subset of products we consider the users previous activity as input.

3.2 Scoring

In the scoring part of the system, the model will give a rank to the candidates generated on the scale of 10. The candidates with the higher scores are recommended to the user.

3.3 Re-ranking

Once the scoring process is done the system undergoes re-ranking to satisfy the additional constraints like ensuring diversity of products, freshness. To achieve this the system usually do not suggest the product which was disliked by the user and usually suggest the products that are newly available.

The figure 2 below gives an overview of the architecture of the recommender system.



Fig.2 Overall architecture of recommender system

The pipeline of a recommendation system has the following five phases

- 1).Pre-processing
- 2).Model Training
- 3).Hyper Parameter Optimization
- 4).Post Processing
- 5).Evaluation.

Pre-Processing

The key idea to perform the pre-processing is to generate a user-item matrix as shown in the figure 3

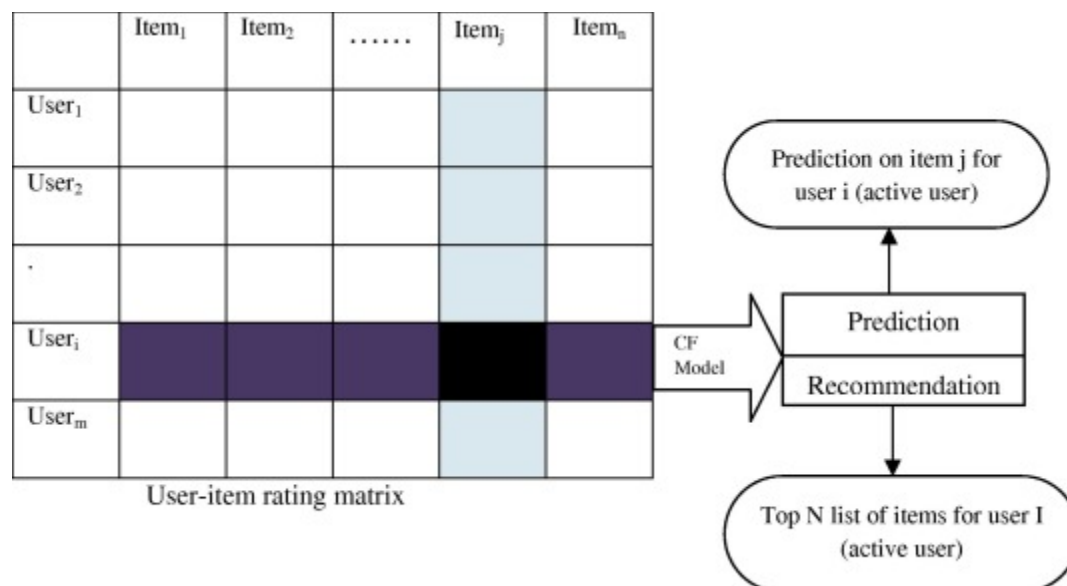


Figure 3: User-Item Rating Matrix

Normalization:

In the dataset there are users who are over reactive and provide high ratings or least ratings. We need to normalize the ratings from such users. For this we find the average rating of the product and average rating of the user and add them to the global average .

Formula for normalization is

$$\text{User-item rating bias} = \text{global average} + \text{item's average rating} + \text{user's average rating}.$$

Model Training:

After the pre-processing the next step is to train the model. In model building the common technique is to use matrix factorization We have used Single Value Decomposition (SVD) to do the decomposing of the original matrix and obtain the latent factor which helps in generating the recommendations .

Hyper Parameter Optimization:

An evaluation metric is to be chosen before the parameters are being tuned. In this we choose the value of K which indicates the number of recommendations to be generated as suggestion to the

user. In our work conducted we have chosen the value of K as twenty five 5 so it recommends the top twenty five products to the user.

Post Processing:

As a part of post-processing we identify if there are any items that are previously purchased by the user and remove them from the generated recommendations. Thus in post-processing we ensure that there are no duplicates of the products.

Evaluation:

The performance of the model can be tested using one of the evaluation metric. In this project we have chosen Root Mean Square Error (RMSE) as the evaluation metric. This paper makes use of python libraries like NumPy, pandas, matplotlib, and Seaborn, Scikit learn. These libraries are part of the system design as they work on the data set and provide the appropriate results.

4.Implementation:

The system was built using Python as the programming language. Python was chosen because of its ease of use and lot of libraries are available inbuilt, which makes the implementation part convenient and less stressful. In the initial step we consider the dataset that we need for the recommender system. The dataset has four columns. The first column is User id which is unique for every user, Second column is the Product id which is unique for every product in the dataset, Third column has the rating for a product given by the respective user. Fourth column is timestamp which gives information about the time of the rating. After considering the dataset the next step is

to perform Exploratory Data Analysis(EDA) as it helps in providing the context that is needed for building a good model. EDA gives information which is used for identifying any errors in the dataset. We pre-process the dataset to check if there is accurate data in each field of all the four columns of the dataset and to avoid any null values. The results of the pre-processing of the dataset can be observed from the table 1.

Number of missing values across columns :

User ID	0
Product ID	0
Rating	0
Time Stamp	0
Dtype	0

table 1: Results of Pre-Processing data.

After reading the dataset from the CSV file we want to ensure that all the columns are read into the system correctly without any missing values. table 2 displays the total number of ratings , users and number of products in the dataset.

Total data	
Total no of ratings	7824482
Total no of users	4201696
Total no of products	476002

Table2:Information about total values in dataset

By using the .info() and .dtypes we can get a information about index dtype,column dtype and usage of the memory. The results can be seen in the table 3.

User Id	object
Product Id	object
Rating	Float64
Time Stamp	Int64
Dtype	object
Range index	7824882 entries, 0 to 7824881
Data columns (total 4 columns)	
User Id	object
Product Id	object
Rating	Float64
TimeStamp	Int64
dtypes	Float64(1), int64(1),object(2),
Memory usage	238.8 +MB

Table 3:Concise summary of the dataframe

To see the structure of the dataset we have used the `.describe()` function which gives information about the count, mean, standard deviation and other quantile values like min, 50% and max are shown in the table 4 below

count	4.201696e+06
mean	1.862220e+00
Std	2.885110e+00
min	1.000000e+00
25%	1.000000e+00
50%	1.000000e+00
75%	2.000000e+00
max	5.200000e+02
Name	Rating

Table 4: Summary of the dataset

After reading the dataset and finding all the details, we move to the ratings column in the dataset. Since it uses the ratings from different users, we plot the overall ratings to see if they are well distributed as shown in Figure 4. The plot shows that five star ratings are given the most by the users to different products and lowest number of users rated the products with two star ratings. To

plot this graph we have used the seaborn library which is built on top of matplotlib and is integrated with the pandas.

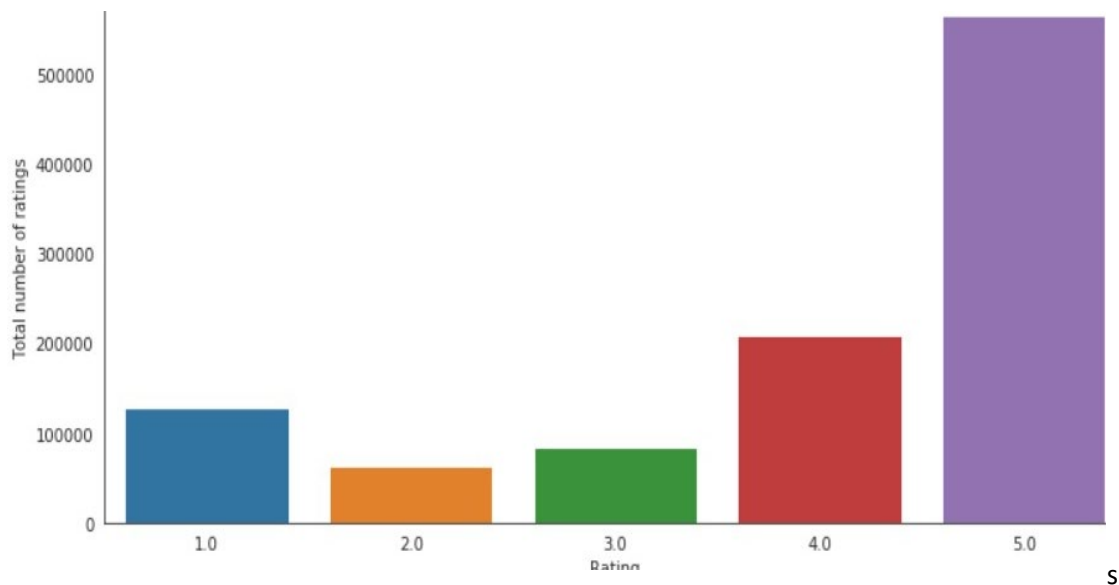


Figure 4. Distribution Of Ratings

From this dataset we drop the unnecessary columns that we may not need to make it more usable. Next we find the popular products among all the products and average rating for the products. In the next step we have to choose a classification algorithm to generate the recommendations. In this paper it is decided to use the KNN classification algorithm. This is mainly because there is a large amount of training data available compared to the features or columns of the dataset. So KNN can do a better job compared to SVM which is another classification algorithm. The KNN algorithm is imported from the “Surprise” python library. The next step is to use KNN algorithm to find similarity between the products as shown in the figure 5 depending on the other products features and their ratings, and similar purchases done by the other users. The KNN algorithm is imported from the “Surprise” python library.

k-Nearest Neighbor (kNN)

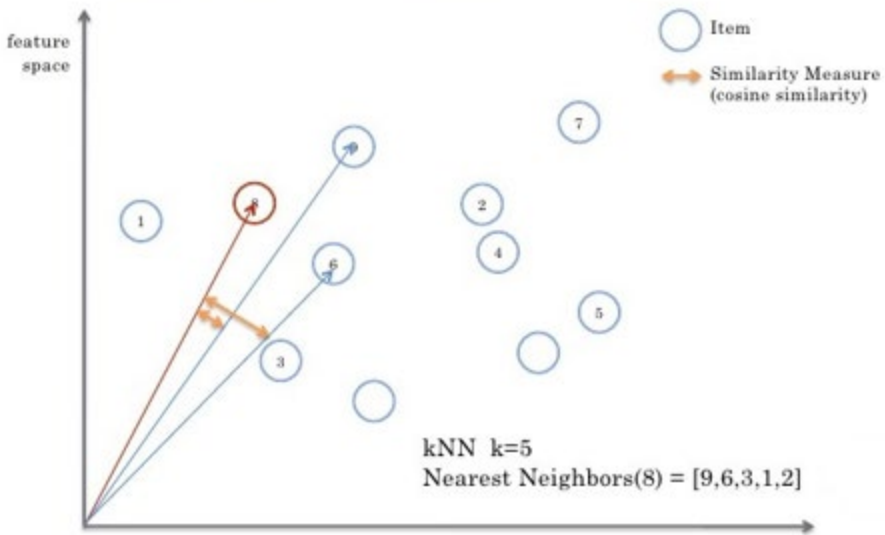


Figure5: KNN Algorithm functioning

Once all this process is completed, we train our model on the dataset. The model will now be able to predict different products for the users. In the table 5 we can see the results which contains the Product id's that are finally recommended to the user by the Recommender System

'1400501520'
'1400532736'
'1400599997'
'1400698987'
'6301977173'
'9573212919'

'9575871979'
'9625993428'
'9888002198'
'9966694544'
'9984984354'
'9985511476'
'B0000010M5'
'B00000J0D2'
'B00000J0D8'
'B00000J3UJ''
' B00000J4EY'
'B00000J6WY'
'B00000JBAT'
'B00000JBHP'
'B00000JD34'
'B00000JFIF'
'B00000JMUG'
'B00000JYLO'

Table5: Results of recommendations generated

To evaluate the performance of the model we have used the Root Mean Square Error (RMSE) which gives the accuracy of the recommendations that are generated. The results of the RMSE can gives a value of 1.3436.To eliminate the products that were already bought by the users, we

decompose the outputs. This removes the products that were already bought by the user. Therefore, the final recommendation will not have any duplicates in it.

5. Conclusion:

The primary goal of this project is to provide recommendations to the user in a e-commerce website by making use of machine learning algorithms. We have designed and implemented the system using collaborative filtering and Pearson correlation coefficient. The dataset considered has the ratings given by the other users to a specific product and depending on the similarity between the rated product we try to recommend the products to our current user. The future work of the project includes improving the efficiency of the system. And it should also be able to give appropriate recommendations to the users who don't have any previous purchase history or to the new users. In future we can try to use recurrent neural networks and deep learning. With the help of deep learning techniques we can overcome some of the drawbacks of the matrix factorization technique. Deep learning uses recurrent neural networks to accommodate time in the recommender system which is not possible in the matrix factorization method. We can also work on providing sub-optimal recommendations to the user and record the reaction of the user and it can be used in the future by the system.

6 References:

- 1]. [Wikipedia \(https://en.wikipedia.org/wiki/Recommender_system\)](https://en.wikipedia.org/wiki/Recommender_system)
- 2]. [Wikipedia \(https://en.wikipedia.org/wiki/Collaborative_filtering\)](https://en.wikipedia.org/wiki/Collaborative_filtering)
- 3]. [Wikipedia \(https://en.wikipedia.org/wiki/Matrix_factorization_\(recommender_systems\)\)](https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems))
- 4]. [Wikipedia \(https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm\)](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm)
- 5]. [Wikipedia \(https://en.wikipedia.org/wiki/Pearson_correlation_coefficient\)](https://en.wikipedia.org/wiki/Pearson_correlation_coefficient)
- 6]. <https://numpy.org/>
- 7]. <https://pandas.pydata.org/>
- 8]. <https://matplotlib.org/>
- 9]. <https://scikit-learn.org/stable/>
- 10]. <http://surpriselib.com/>
- 11]. <https://seaborn.pydata.org/>
- 12]. <https://www.scipy.org/>
- 13]. Silvana Aciar ; Debbie Zhang ; Simeon Simoff ; John Debenham, Recommendation System Based On Consumer Product Reviews Online source: [IEEE Xplore \(https://ieeexplore.ieee.org/document/4061458\)](https://ieeexplore.ieee.org/document/4061458)
- 14]. Kunal Shah ; Akshaykumar Salunke ; Saurabh Dongare ; Kisandas Antala, Recommender systems: An overview of different approaches to recommendations.
Online source: [IEEE Xplore \(https://ieeexplore.ieee.org/document/8276172\)](https://ieeexplore.ieee.org/document/8276172)
- 15]. Yuri Stekh ; Mykhoylo Lobur ; Vitalij Artsibasov ; Vitalij Chystyak, Methods and tools for building recommender systems.

Online source: [IEEE Xplore \(https://ieeexplore.ieee.org/document/7230862\)](https://ieeexplore.ieee.org/document/7230862)

16].GregLinden,BrentSmith,JeremyYork,Amazon.com Recommendations item-to-item Collaborative Filtering. Online source: [IEEE Computer Society \(https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf\)](https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf)

17].Badrul Sarwar,George Karypis,Joseph Konstan,John Riedl, Item-Based Collaborative Filtering Recommendations Algorithms, Online source: [Group Lens \(http://files.grouplens.org/papers/www10_sarwar.pdf\)](http://files.grouplens.org/papers/www10_sarwar.pdf)

18]. Online source: [Towards Data Science \(https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15\)](https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15)

19]. [Amazon \(https://www.kaggle.com/saurav9786/amazon-product-reviews\)](https://www.kaggle.com/saurav9786/amazon-product-reviews)

20]. K.Yogeswara Rao,G.S.N.Murthy,S.Adhinarayana “Product Recommendation System from Users Reviews using Sentiment Analysis”[Research Gate \(https://www.researchgate.net/publication/318493578_Product_Recommendation_System_from_Users_Reviews_using_Sentiment_Analysis\)](https://www.researchgate.net/publication/318493578_Product_Recommendation_System_from_Users_Reviews_using_Sentiment_Analysis)

21].Jatinder Kaur,Rajeev Kumar Bedi, S.K.Gupta “Product Recommendation Systems a Comprehensive Review.” [Research Gate \(https://www.researchgate.net/publication/326555590_Product_Recommendation_Systems_a_Comprehensive_Review\)](https://www.researchgate.net/publication/326555590_Product_Recommendation_Systems_a_Comprehensive_Review)

22].Neha Varma,Devanand,Bhavna Arora “Experimental Analysis of Recommendation System in e-Commerce”
[Academic Journal \(https://www.ijitee.org/wp-content/uploads/papers/v8i8s3/H10330688S319.pdf\)](https://www.ijitee.org/wp-content/uploads/papers/v8i8s3/H10330688S319.pdf)

International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075,
Volume-8 Issue-8S3, June 2019

- 23]. Fatima Rodrigues, Bruno Ferreira “Product Recommendation based on Shared Customer's Behaviour” [Science Direct \(https://www.sciencedirect.com/science/article/pii/S1877050916323018\)](https://www.sciencedirect.com/science/article/pii/S1877050916323018)
- 24]. Jianfeng Hu, Bo Zhang “Product Recommendation System” [Stanford \(http://snap.stanford.edu/class/cs224w-2012/projects/cs224w-044-final.v01.pdf\)](http://snap.stanford.edu/class/cs224w-2012/projects/cs224w-044-final.v01.pdf)
- 25]. <https://www.anaconda.com/>
- 26]. <https://pypi.org/project/pip/>
- 27]. [Tutorial \(https://packaging.python.org/tutorials/installing-packages/#use-pip-for-installing\)](https://packaging.python.org/tutorials/installing-packages/#use-pip-for-installing)
- 28]. [Amazon \(https://www.amazon.science/the-history-of-amazons-recommendation-algorithm\)](https://www.amazon.science/the-history-of-amazons-recommendation-algorithm)
- 29]. Iateilang Rynksai, L. Chameikho “Recommender Systems: Types Of Filtering Techniques” [IJert \(https://www.ijert.org/research/recommender-systems-types-of-filtering-techniques-IJERTV3IS110197.pdf\)](https://www.ijert.org/research/recommender-systems-types-of-filtering-techniques-IJERTV3IS110197.pdf)
- 30]. [Towards Data Science \(https://towardsdatascience.com/recommender-systems-the-most-valuable-application-of-machine-learning-part-1-f96ecbc4b7f5\)](https://towardsdatascience.com/recommender-systems-the-most-valuable-application-of-machine-learning-part-1-f96ecbc4b7f5)

7 Appendix:

```
# loading packages
```

```
import numpy as np
```

```
import pandas as pd
```

```
import os
```

```
from IPython.core.interactiveshell import InteractiveShell
```

```
InteractiveShell.ast_node_interactivity = "all"
```

```
import math
```

```
import json
```

```
import time
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
from sklearn.metrics.pairwise import cosine_similarity
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.neighbors import NearestNeighbors
```

```
import scipy.sparse
```

```
from scipy.sparse import csr_matrix
```

```
from scipy.sparse.linalg import svds
```

```
import warnings; warnings.simplefilter('ignore')
```

```
%matplotlib inline

# loading data

electronics_data=pd.read_csv("C:/Users/mukhu/OneDrive/Desktop/New
folder/Dataset.csv",names=['userId', 'productId','Rating','timestamp'])

# Display the data

electronics_data.head()

#Shape of the data

electronics_data.shape

#Taking subset of the dataset

electronics_data=electronics_data.iloc[:1048576,0:]

#Check the datatypes

electronics_data.dtypes

electronics_data.info()

#Five point summary

electronics_data.describe()['Rating'].T

#Find the minimum and maximum ratings

print('Minimum rating is: %d' %(electronics_data.Rating.min()))

print('Maximum rating is: %d' %(electronics_data.Rating.max()))
```



```
#Check for missing values
```

```
print('Number of missing values across columns: \n',electronics_data.isnull().sum())
```

```
# Check the distribution of the rating
```

```
with sns.axes_style('white'):
```

```
    g = sns.factorplot("Rating", data=electronics_data, aspect=2.0,kind='count')
```

```
    g.set_ylabels("Total number of ratings")
```

```
print("The above bar chart represents the distribution of ratings by user. From the chart, the  
highest proportion of user rated the products 5.0 while the lowest number of users rated the  
products 2.0")
```

```
# Unique user products
```

```
print("Total data ")
```

```
print("-"*50)
```

```
print("\nTotal no of ratings :",electronics_data.shape[0])
```

```
print("Total No of Users  :", len(np.unique(electronics_data.userId)))
```

```
print("Total No of products  :", len(np.unique(electronics_data.productId)))
```

```
#Dropping the Timestamp column
```

```
electronics_data.drop(['timestamp'], axis=1,inplace=True)
```

```
#Analysis of rating given by the user
```

```

no_of_rated_products_per_user =
electronics_data.groupby(by='userId')['Rating'].count().sort_values(ascending=False)

no_of_rated_products_per_user.head()

print("The output above represents the top 5 most preferred products by users.")

no_of_rated_products_per_user.describe()

quantiles = no_of_rated_products_per_user.quantile(np.arange(0,1.01,0.01),
interpolation='higher')

plt.figure(figsize=(10,10))

plt.title("Quantiles and their Values")

quantiles.plot()

# quantiles with 0.05 difference

plt.scatter(x=quantiles.index[::5], y=quantiles.values[::5], c='orange', label="quantiles with 0.05
intervals")

# quantiles with 0.25 difference

plt.scatter(x=quantiles.index[::25], y=quantiles.values[::25], c='m', label = "quantiles with 0.25
intervals")

plt.ylabel('No of ratings by user')

plt.xlabel('Value at the quantile')

plt.legend(loc='best')

```

```

plt.show()

print('\n No of rated product more than 50 per user :

{}\n'.format(sum(no_of_rated_products_per_user >= 50)))

# Popularity Based Recommendation

#Getting the new dataframe which contains users who has given 50 or more ratings

new_df=electronics_data.groupby("productId").filter(lambda x:x['Rating'].count() >=50)

no_of_ratings_per_product =

new_df.groupby(by='productId')['Rating'].count().sort_values(ascending=False)

fig = plt.figure(figsize=plt.figaspect(.5))

ax = plt.gca()

plt.plot(no_of_ratings_per_product.values)

plt.title('# RATINGS per Product')

plt.xlabel('Product')

plt.ylabel('No of ratings per product')

ax.set_xticklabels([])

plt.show()

#Average rating of the product

new_df.groupby('productId')['Rating'].mean().head()

new_df.groupby('productId')['Rating'].mean().sort_values(ascending=False).head()

```

```
#Total no of rating for product
```

```
new_df.groupby('productId')['Rating'].count().sort_values(ascending=False).head()
```

```
ratings_mean_count = pd.DataFrame(new_df.groupby('productId')['Rating'].mean())
```

```
ratings_mean_count['rating_counts'] =
```

```
pd.DataFrame(new_df.groupby('productId')['Rating'].count())
```

```
ratings_mean_count.head()
```

```
print('The above output indicates that the predicted products with the highest number of ratings  
as well as the corresponding number of rating by the users.')
```

```
ratings_mean_count['rating_counts'].max()
```

```
plt.figure(figsize=(8,6))
```

```
plt.rcParams['patch.force_edgecolor'] = True
```

```
ratings_mean_count['rating_counts'].hist(bins=50)
```

```
plt.figure(figsize=(8,6))
```

```
plt.rcParams['patch.force_edgecolor'] = True
```

```
ratings_mean_count['Rating'].hist(bins=50)
```

```
print('The above bar chart indicates that the ratings were negatively skewed. This implies that  
there were less products that had lower ratings but most products had higher ratings.')
```

```
plt.figure(figsize=(8,6))
```

```
plt.rcParams['patch.force_edgecolor'] = True
```

```

sns.jointplot(x='Rating', y='rating_counts', data=ratings_mean_count, alpha=0.4)

popular_products = pd.DataFrame(new_df.groupby('productId')['Rating'].count())

most_popular = popular_products.sort_values('Rating', ascending=False)

most_popular.head(30).plot(kind = "bar")

# Collaborative filtering (Item-Item recommendation)

from surprise import KNNWithMeans

from surprise import Dataset

from surprise import accuracy

from surprise import Reader

import os

from surprise.model_selection import train_test_split

#Reading the dataset

reader = Reader(rating_scale=(1, 5))

data = Dataset.load_from_df(new_df,reader)

#Splitting the dataset

trainset, testset = train_test_split(data, test_size=0.3,random_state=10)

# Use user_based true/false to switch between user-based or item-based collaborative filtering

algo = KNNWithMeans(k=5, sim_options={'name': 'pearson_baseline', 'user_based': False})

```

```

algo.fit(trainset)

# run the trained model against the testset

test_pred = algo.test(testset)

test_pred[1:100]

print('The output above represents the predicted products and the user who predicted the product
as well as the predicted rating by the user')

# get RMSE

print("Item-based Model : Test Set")

accuracy.rmse(test_pred, verbose=True)

print("root mean square error (RMSE) was used for evaluating the performance of the recommen
der on the test data. The RMSE was 1.3446 which implied that the recommender system moderat
ely performed")

# Model-based collaborative filtering system

new_df1=new_df.head(10000)

ratings_matrix = new_df1.pivot_table(values='Rating', index='userId', columns='productId', fill_
value=0)

ratings_matrix.head()

ratings_matrix.shape

X = ratings_matrix.T

X.head()

X.shape

#Decomposing the Matrix

```

```
from sklearn.decomposition import TruncatedSVD

SVD = TruncatedSVD(n_components=10)

decomposed_matrix = SVD.fit_transform(X)

decomposed_matrix.shape

#Correlation Matrix

correlation_matrix = np.corrcoef(decomposed_matrix)

correlation_matrix.shape

X.index[75]

i = "B00000K135"

product_names = list(X.index)

product_ID = product_names.index(i)

product_ID

correlation_product_ID = correlation_matrix[product_ID]

correlation_product_ID.shape

Recommend = list(X.index[correlation_product_ID > 0.65])

# Removes the item already bought by the customer

Recommend.remove(i)

Recommend[0:24]

print("The output above represents the recommended products to the user")

Recommend = list(X.index[correlation_product_ID > 0.65])

# Removes the item already bought by the customer

Recommend.remove(i)

Recommend[0:24]
```

```
Out[53]: ['1400501520',  
          '1400532736',  
          '1400599997',  
          '1400698987',  
          '6301977173',  
          '9573212919',  
          '9575871979',  
          '9625993428',  
          '9888002198',  
          '9966694544',  
          '9984984354',  
          '9985511476',  
          'B0000010M5',  
          'B00000J0D2',  
          'B00000J0D8',  
          'B00000J3UJ',  
          'B00000J4EY',  
          'B00000J6WY',  
          'B00000JBAT',  
          'B00000JBHP',  
          'B00000JD34',  
          'B00000JFIF',  
          'B00000JMUG',  
          'B00000JYLO']
```

print('The output above represents the products recommended to the user which are not part of the products that were already bought by the user')