Building a Personalized Online Course Recommender System with Machine Learning

Tanvi Pathak

(103)

DeepLearning.TV
ML0115EN - v1.0
Deep Learning Fundamentals
Starts: Any time, Self-paced
Course ★ No ratings yet

IBM
BD0141EN - v2016.0
Accessing Hadoop Data Using Hive
Starts: Any time, Self-paced
Course ★ 4.5/5 (60)

Big Data University
BD0115EN - v2016.0
MapReduce and YA
Starts: Any time, Self-pa
Course ★ No r

curit

IBMDeveloperSkillsNetwork
CC0210EN - v1.0
erverless Computing using
d Functions -...
Any time, Self-paced
★ No ratings yet

IBMDeveloperSkillsNetwork
BC0201EN - v2.0
IBM Blockchain Foundation Developer
Starts: Any time, Self-paced
Course ★ No ratings ye

Big Dat
BD

Outline

- INTRODUCTION AND BACKGROUND
- EXPLORATORY DATA ANALYSIS
- CONTENT-BASED RECOMMENDER SYSTEM USING UNSUPERVISED LEARNING
- COLLABORATIVE-FILTERING BASED RECOMMENDER SYSTEM USING SUPERVISED LEARNING
- CONCLUSION
- APPENDIX

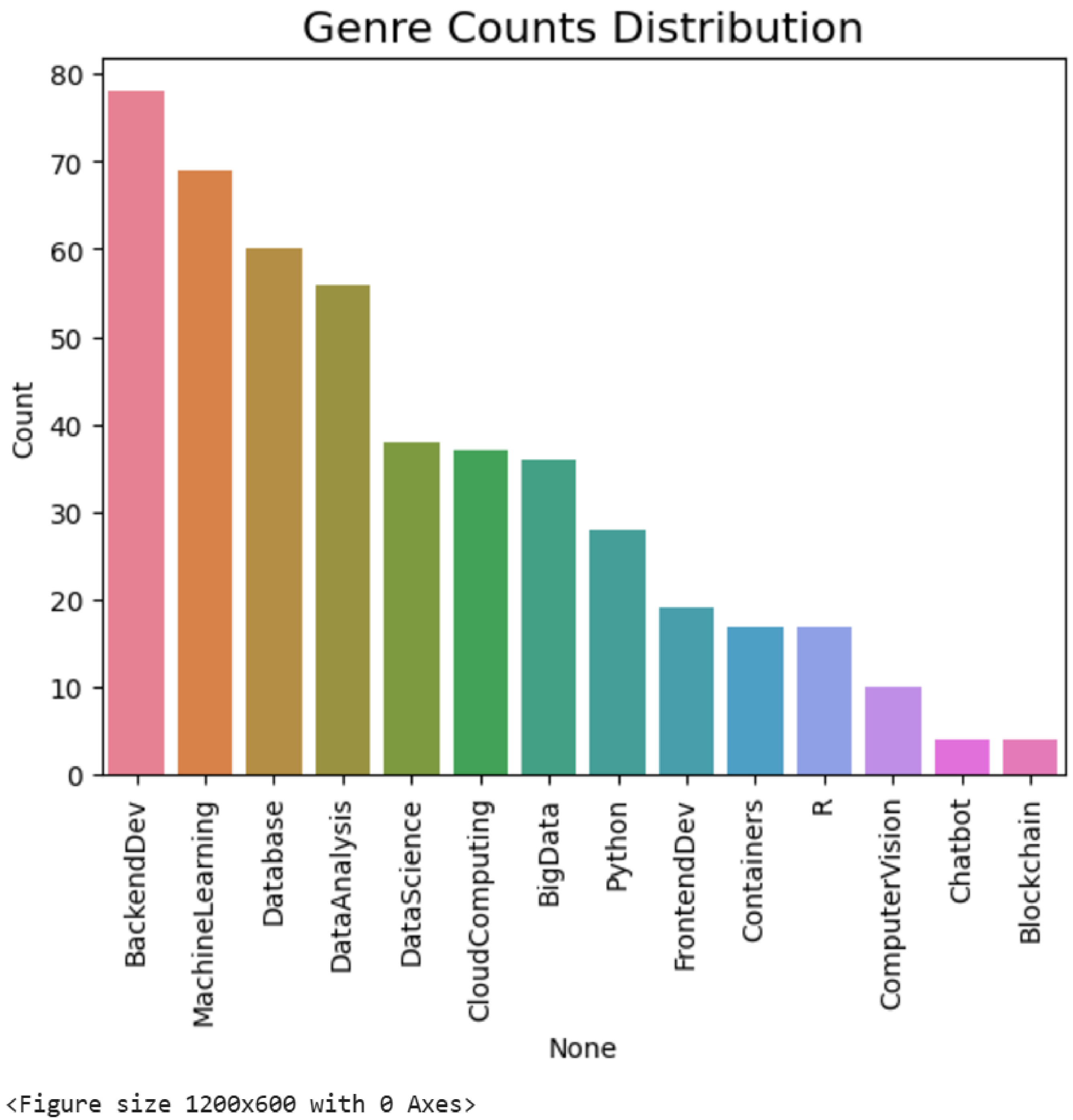
Introduction and Background

- We are going to build a course recommender system for Coursera users.
- The course recommender system aims to:
    - Find courses of similar interest based on the user's interest as well as based on the in interests of people enrolled in similar courses.
    - The system will also recommend unique courses that could be of interest to a user that may not have crossed their minds.
- Background-
    - One of the issues with such data heavy platforms like Coursera is that it may be difficult to gather user information and analyze it. Hence, we will be using different machine learning techniques including unsupervised and supervised learning techniques to build a recommender system.

Exploratory Data Analytics

Counts per Genre

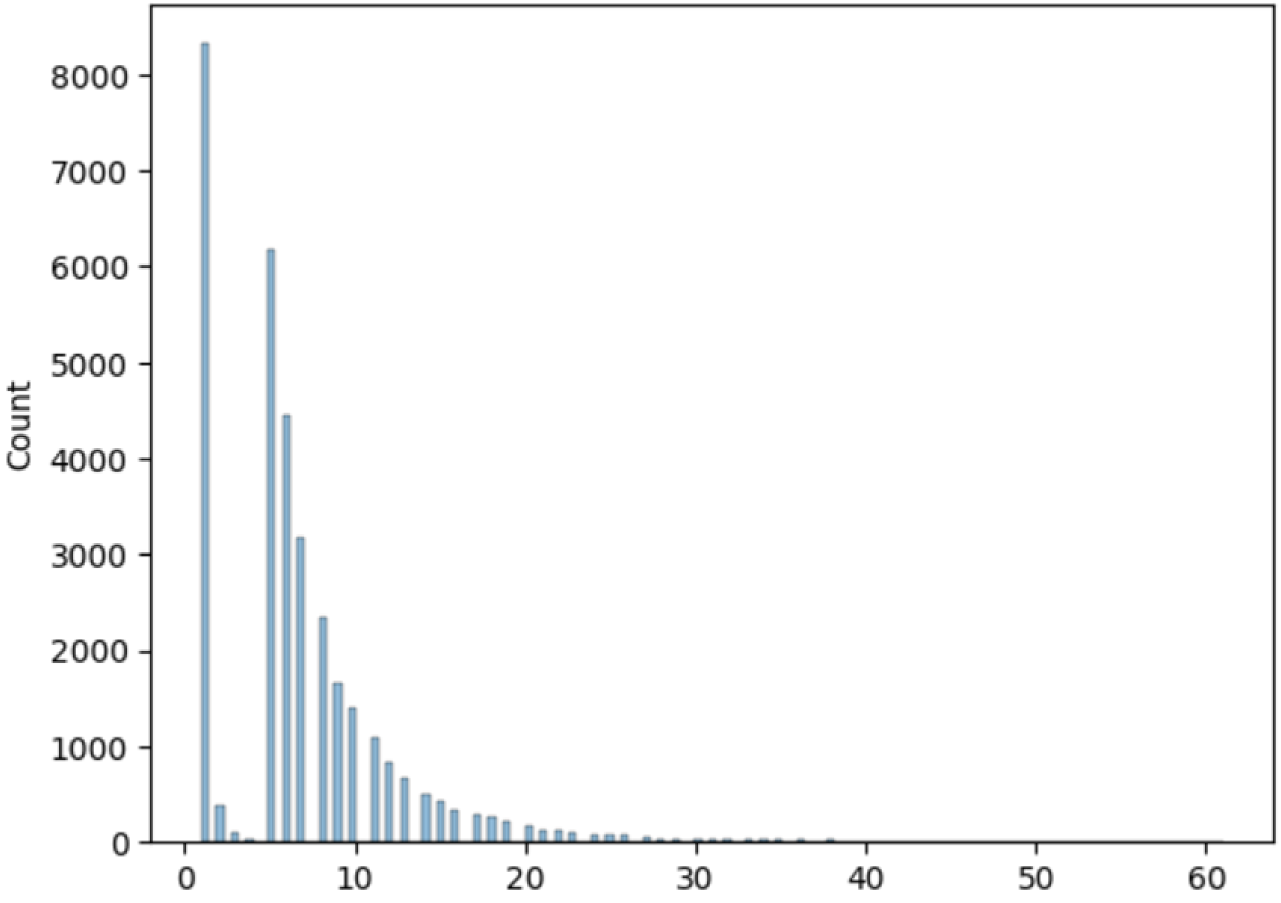
- This graph contains data from a course genre data set that contains information about the course ID, course title, and what genre it belongs to.
- We can see that the most common course genre is BackendDev or Back end Development.
- The least common genre of courses is Blockchain.



```
<Figure size 1200x600 with 0 Axes>
```

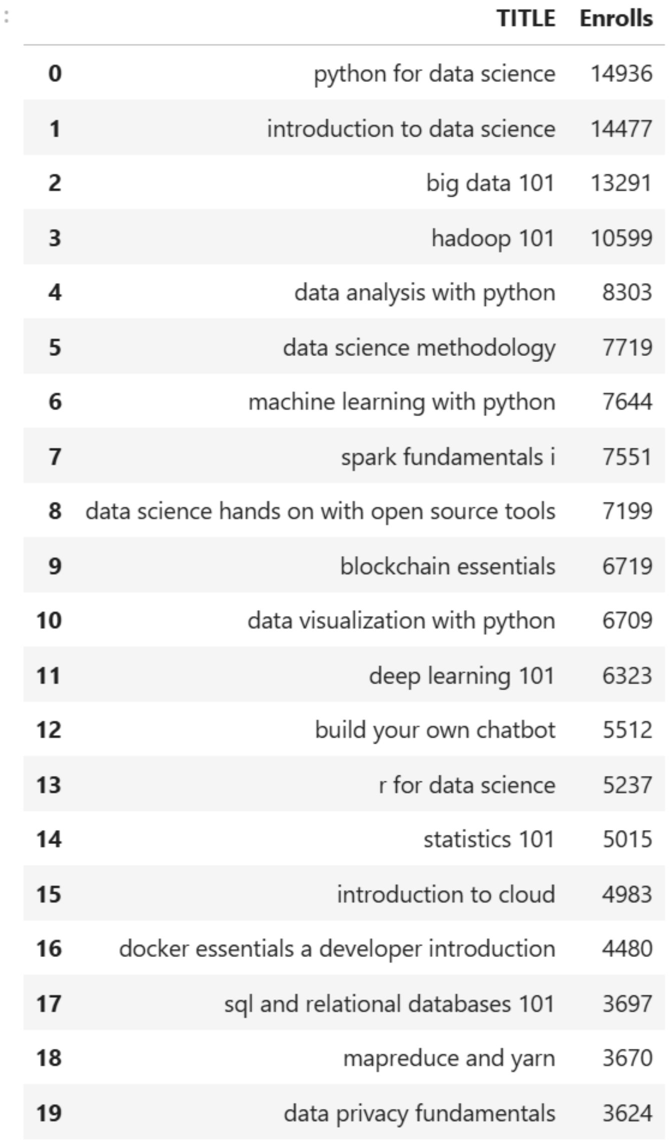
Course Enrollment Distribution

- This graph shows how many times each user rated. This also shows how many different courses each user has taken.
- This shows us how many users rrated just 1 item versus how many users rated more than 1 item.
- For eg. Over 8000 users rated less than 10 items.



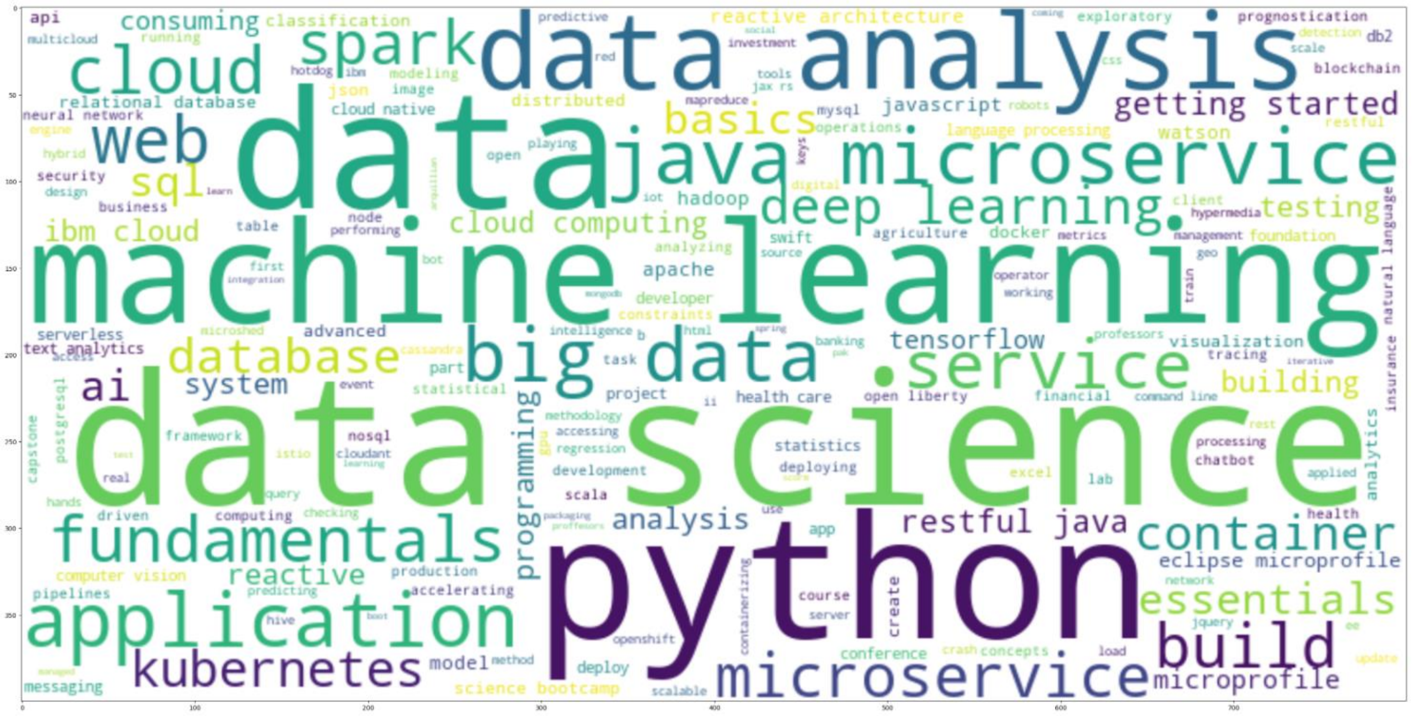
20 Most Popular Courses

- The most popular courses were based on the number of erollments the course had.
- The most popular/most enrolled in course is python for data science.



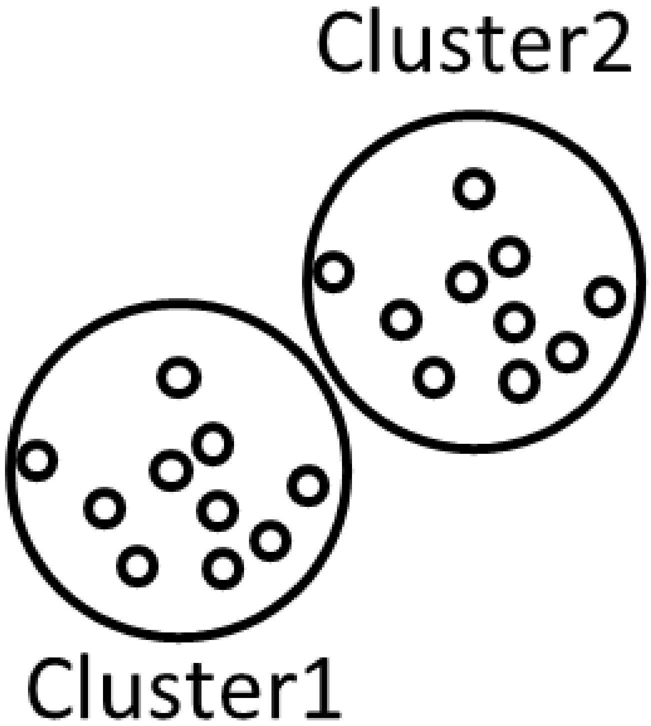
	TITLE	Enrolls
0	python for data science	14936
1	introduction to data science	14477
2	big data 101	13291
3	hadoop 101	10599
4	data analysis with python	8303
5	data science methodology	7719
6	machine learning with python	7644
7	spark fundamentals i	7551
8	data science hands on with open source tools	7199
9	blockchain essentials	6719
10	data visualization with python	6709
11	deep learning 101	6323
12	build your own chatbot	5512
13	r for data science	5237
14	statistics 101	5015
15	introduction to cloud	4983
16	docker essentials a developer introduction	4480
17	sql and relational databases 101	3697
18	mapreduce and yarn	3670
19	data privacy fundamentals	3624

Word Cloud of Course Titles

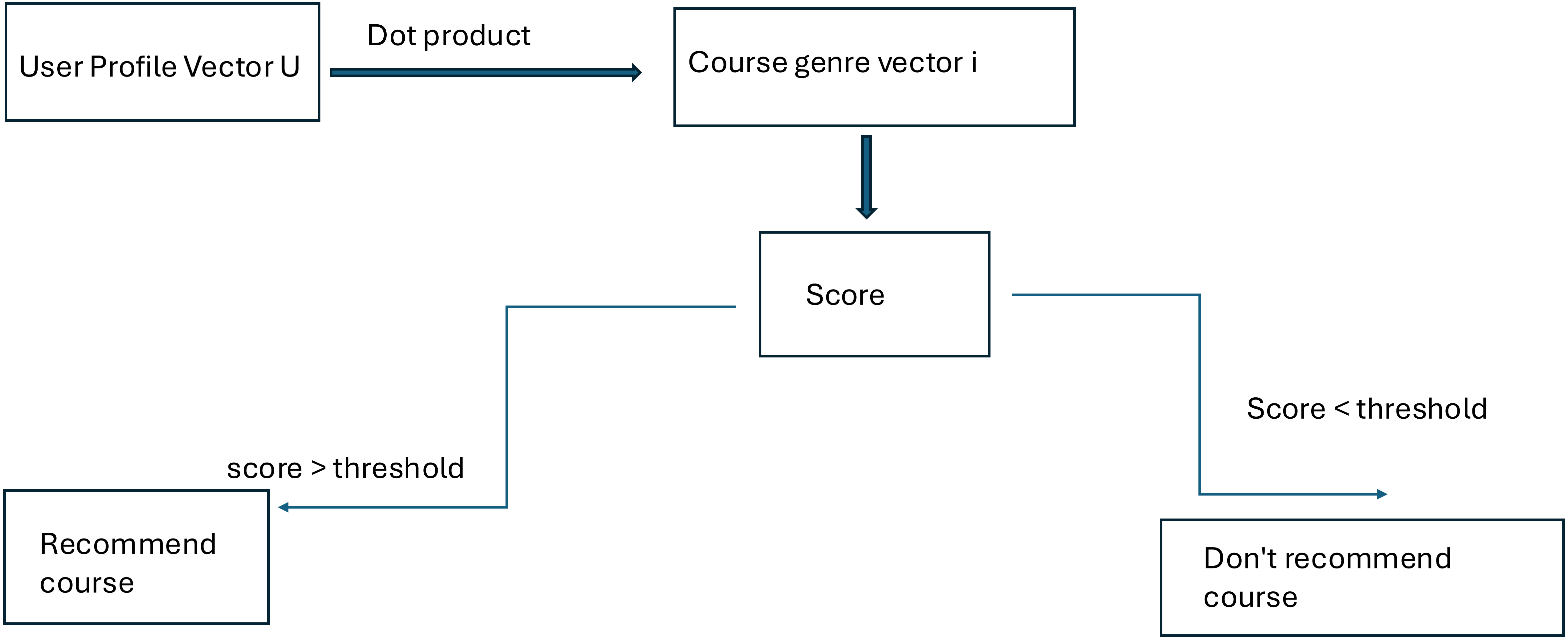


- These are the most common words that appear in each of the course titles.
- Data science, data, python, and machine learning are the most common.

Content Based Recommender System using Unsupervised Learning



Flowchart of Content Based Recommender System using User Profile and Course Genres



User Profile Vector U

Dot product

Course genre vector i

Score

score > threshold

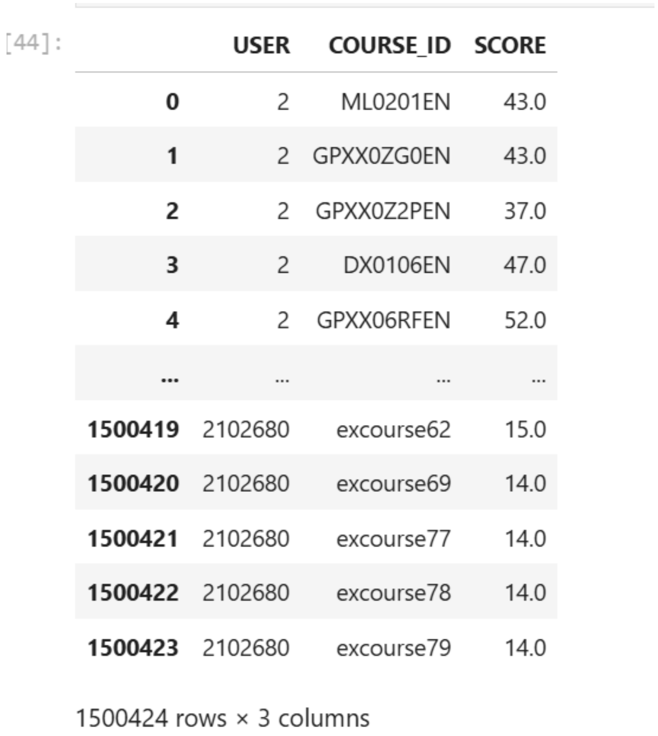
Recommend course

Score < threshold

Don't recommend course

Evaluation Results of User profile-based Recommender System

We used a Score threshold = 10.0

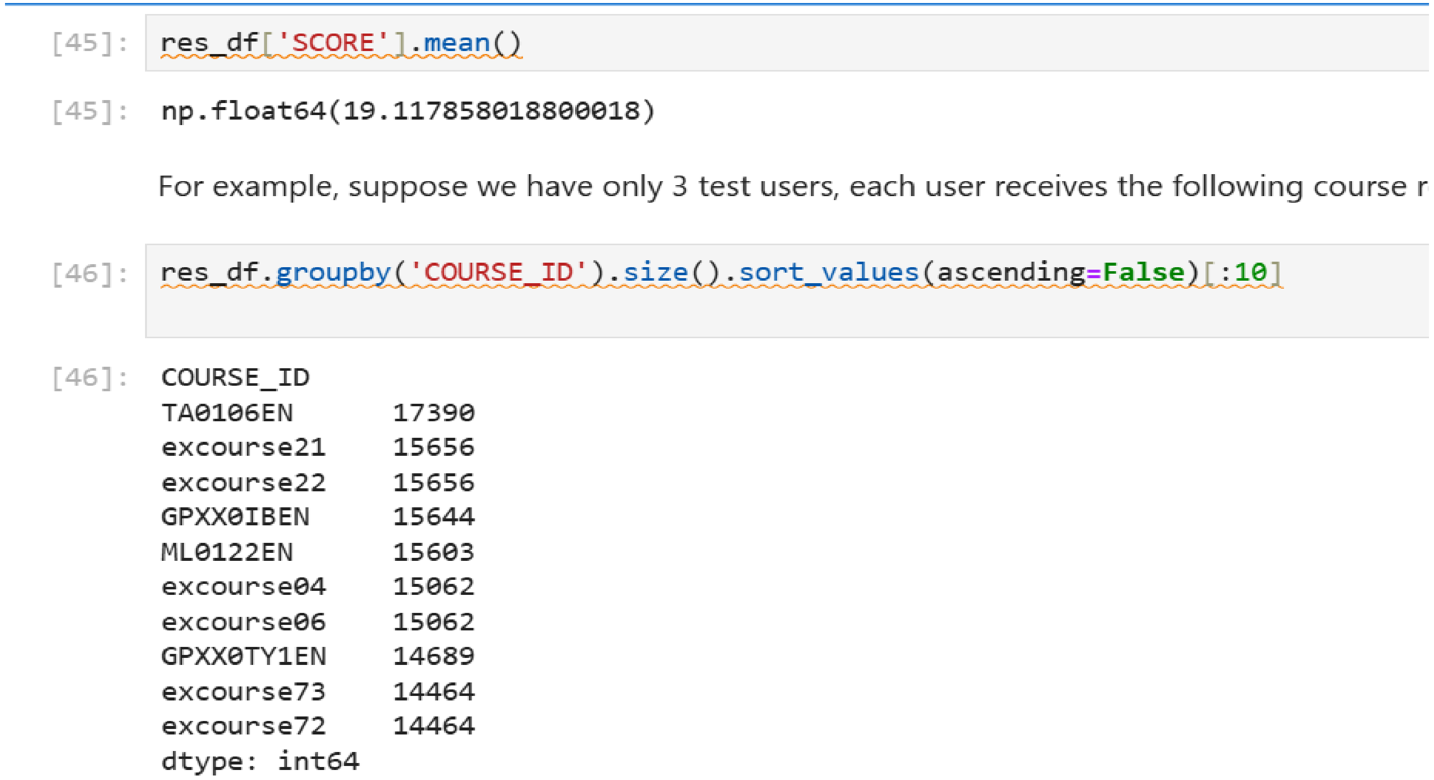


[44]:

	USER	COURSE_ID	SCORE
0	2	ML0201EN	43.0
1	2	GPXX0ZG0EN	43.0
2	2	GPXX0Z2PEN	37.0
3	2	DX0106EN	47.0
4	2	GPXX06RFEN	52.0
...	...	...	...
1500419	2102680	excourse62	15.0
1500420	2102680	excourse69	14.0
1500421	2102680	excourse77	14.0
1500422	2102680	excourse78	14.0
1500423	2102680	excourse79	14.0

1500424 rows × 3 columns

List of recommended courses per user



```
[45]: res_df['SCORE'].mean()

[45]: np.float64(19.117858018800018)
```

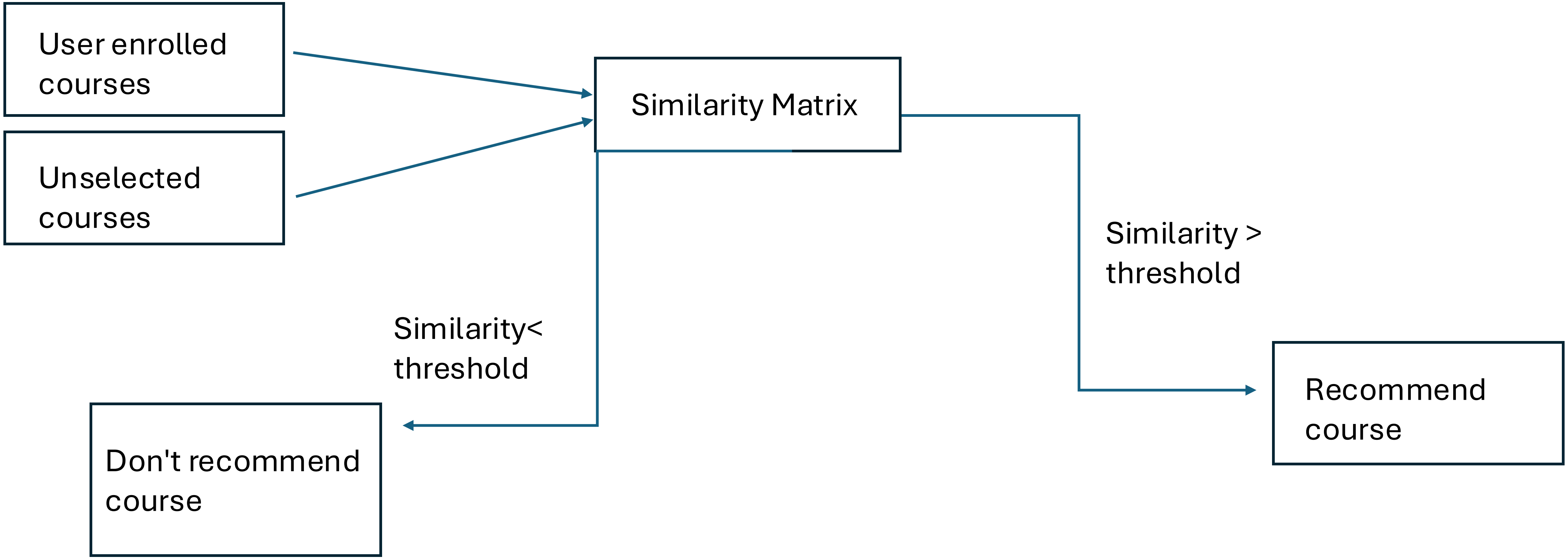
For example, suppose we have only 3 test users, each user receives the following course r

```
[46]: res_df.groupby('COURSE_ID').size().sort_values(ascending=False)[:10]

[46]: COURSE_ID
      TA0106EN       17390
      excourse21     15656
      excourse22     15656
      GPXX0IBEN      15644
      ML0122EN       15603
      excourse04     15062
      excourse06     15062
      GPXX0TY1EN     14689
      excourse73     14464
      excourse72     14464
      dtype: int64
```

This shows us that on average 19 courses have been recommended per test user.
We can also see the top 10 most recommended courses.

Flowchart of Content-based Recommender System using Course Similarity



User enrolled courses

Unselected courses

Similarity Matrix

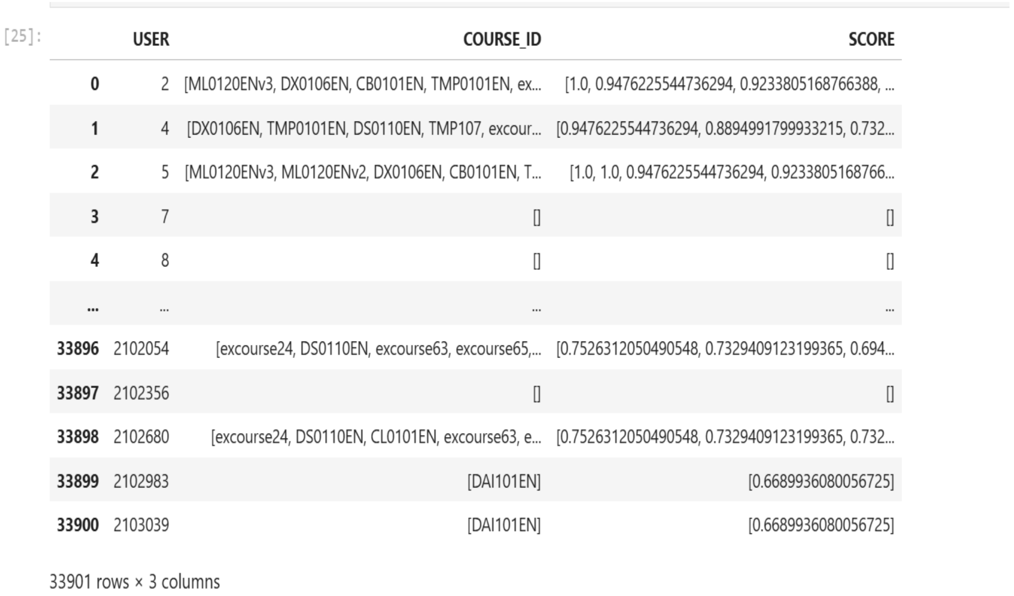
Similarity > threshold

Recommend course

Similarity< threshold

Don't recommend course

Evaluation Results of course similarity based Recommender System

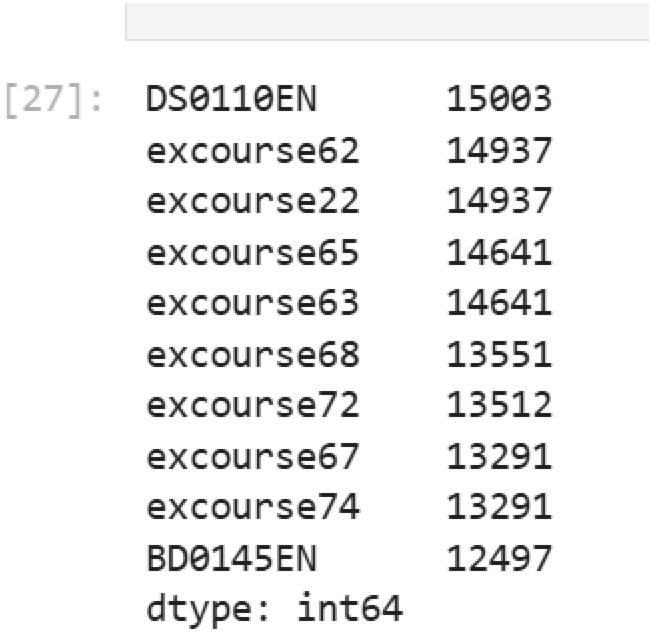


[25]:

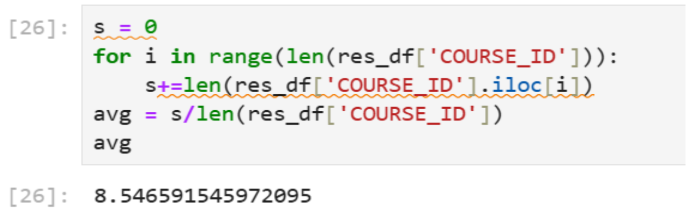
	USER	COURSE_ID	SCORE
0	2	[ML0120ENv3, DX0106EN, CB0101EN, TMP0101EN, ex...	[1.0, 0.9476225544736294, 0.9233805168766388, ...
1	4	[DX0106EN, TMP0101EN, DS0110EN, TMP107, excour...	[0.9476225544736294, 0.8894991799933215, 0.732...
2	5	[ML0120ENv3, ML0120ENv2, DX0106EN, CB0101EN, T...	[1.0, 1.0, 0.9476225544736294, 0.9233805168766...
3	7	[]	[]
4	8	[]	[]
...	...	...	...
33896	2102054	[excourse24, DS0110EN, excourse63, excourse65,...	[0.7526312050490548, 0.7329409123199365, 0.694...
33897	2102356	[]	[]
33898	2102680	[excourse24, DS0110EN, CL0101EN, excourse63, e...	[0.7526312050490548, 0.7329409123199365, 0.732...
33899	2102983	[DAI101EN]	[0.6689936080056725]
33900	2103039	[DAI101EN]	[0.6689936080056725]

33901 rows × 3 columns

- This is the output dataframe.
- It shows us each user, a list of recommended courses based on the similarity scores, and a column showing the similarity scores of each of the recommended courses to the user's enrolled courses.
- **I used a similarity threshold of 0.6**



```
[27]:  DS0110EN       15003
       excourse62     14937
       excourse22     14937
       excourse65     14641
       excourse63     14641
       excourse68     13551
       excourse72     13512
       excourse67     13291
       excourse74     13291
       BD0145EN       12497
       dtype: int64
```



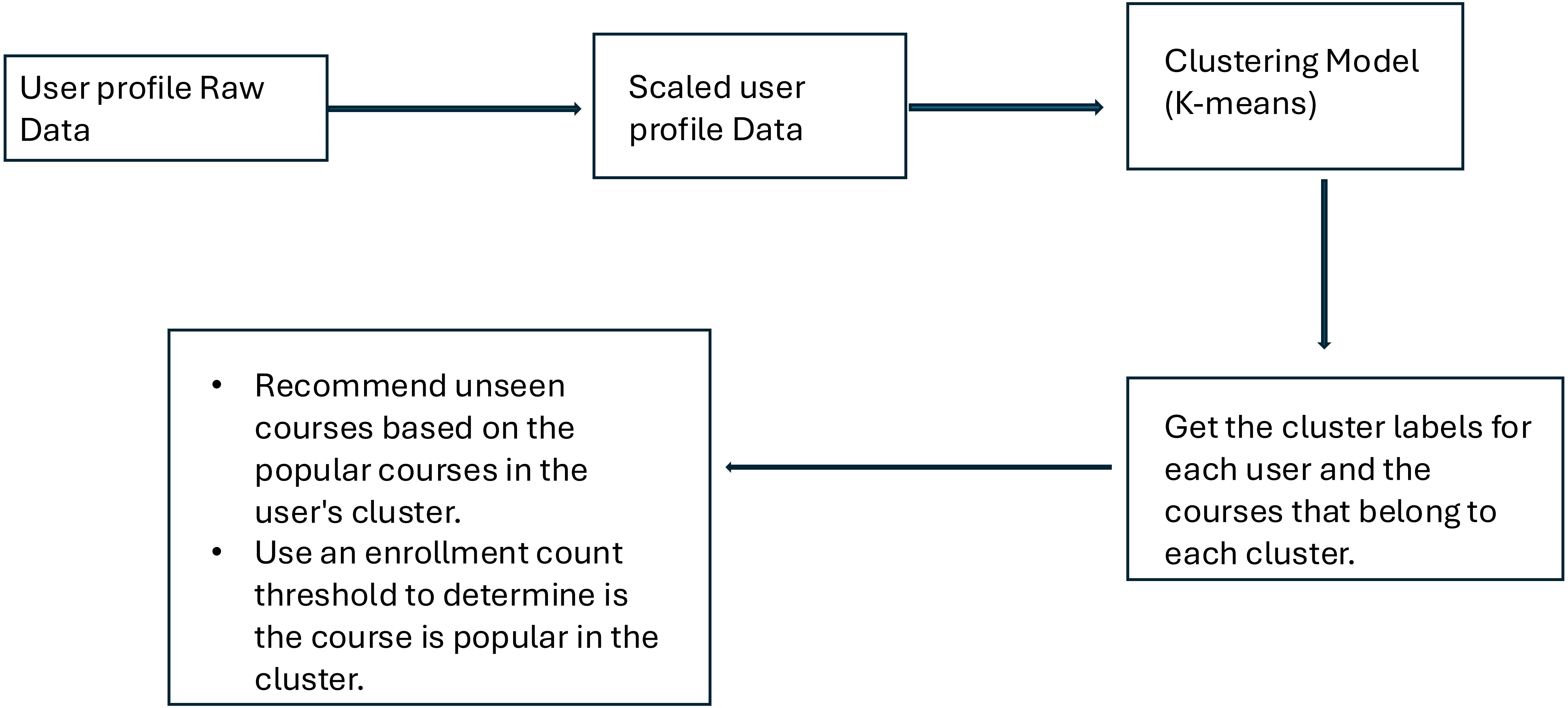
```
[26]:  s = 0
       for i in range(len(res_df['COURSE_ID'])):
           s+=len(res_df['COURSE_ID'].iloc[i])
       avg = s/len(res_df['COURSE_ID'])
       avg

[26]:  8.546591545972095
```

These are the top 10 most frequently recommended courses.

We see that on average, 8.5 courses were recommended per user.

Flowchart of Clustering Based Recommender System



User profile Raw Data

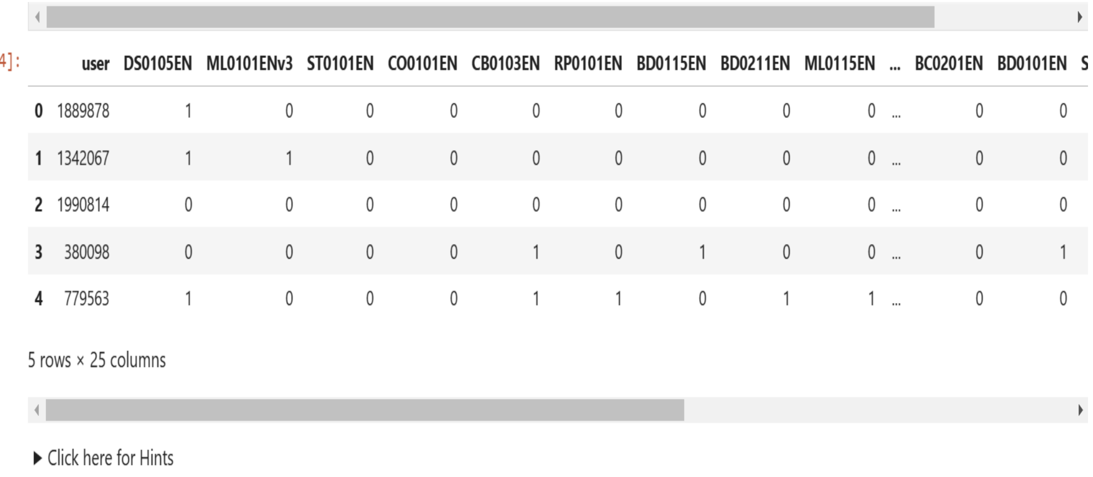
Scaled user profile Data

Clustering Model (K-means)

Get the cluster labels for each user and the courses that belong to each cluster.

- Recommend unseen courses based on the popular courses in the user's cluster.
- Use an enrollment count threshold to determine is the course is popular in the cluster.

Evaluation Results of Clustering-based Recommender System

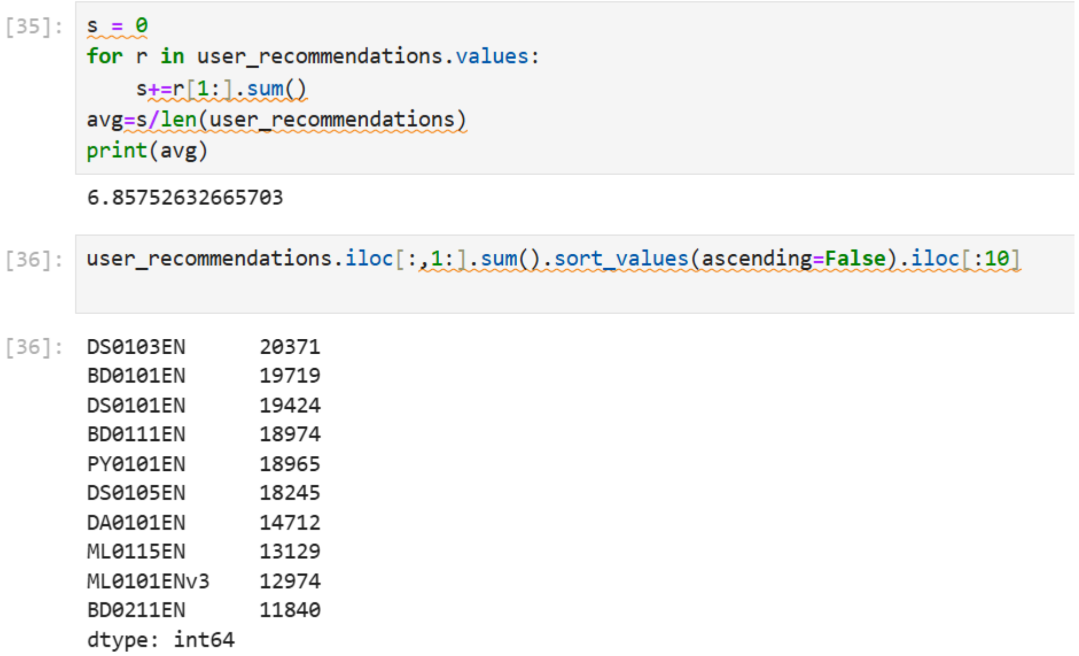


4]:

	user	DS0105EN	ML0101ENv3	ST0101EN	CO0101EN	CB0103EN	RP0101EN	BD0115EN	BD0211EN	ML0115EN	...	BC0201EN	BD0101EN	S
0	1889878	1	0	0	0	0	0	0	0	0	...	0	0	
1	1342067	1	1	0	0	0	0	0	0	0	...	0	0	
2	1990814	0	0	0	0	0	0	0	0	0	...	0	0	
3	380098	0	0	0	0	1	0	1	0	0	...	0	1	
4	779563	1	0	0	0	1	1	0	1	1	...	0	0	

5 rows × 25 columns

▸ Click here for Hints



[35]:
```
s = 0
for r in user_recommendations.values:
    s+=r[1:].sum()
avg=s/len(user_recommendations)
print(avg)
```

```
6.85752632665703
```

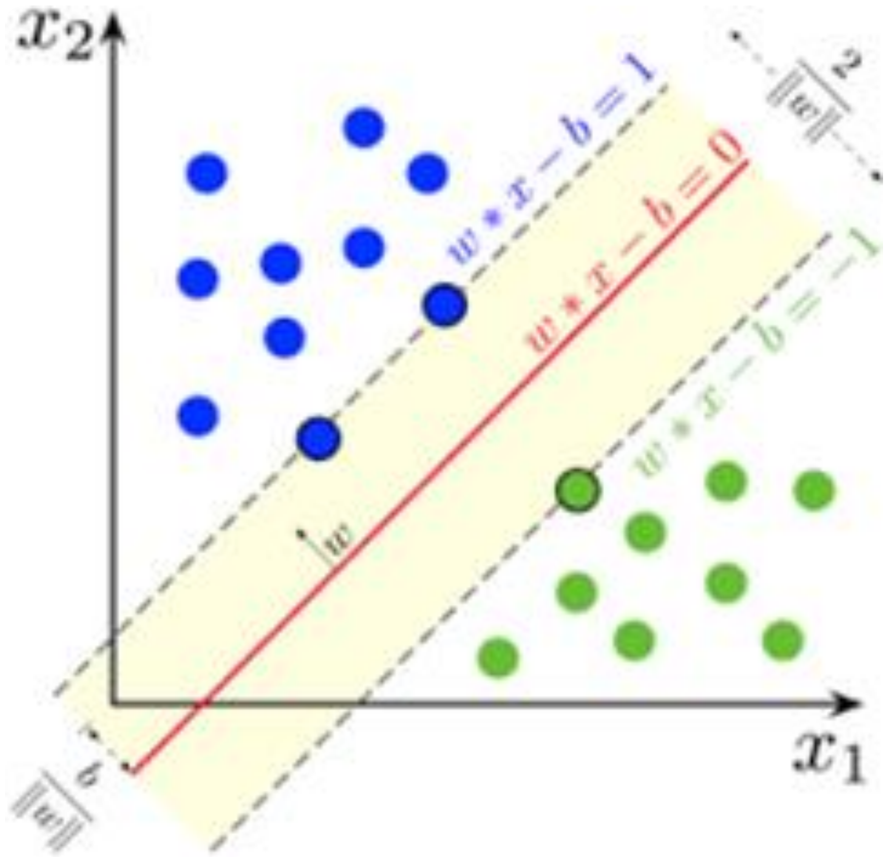
[36]:
```
user_recommendations.iloc[:,1:].sum().sort_values(ascending=False).iloc[:10]
```

[36]:
```
DS0103EN      20371
BD0101EN      19719
DS0101EN      19424
BD0111EN      18974
PY0101EN      18965
DS0105EN      18245
DA0101EN      14712
ML0115EN      13129
ML0101ENv3    12974
BD0211EN      11840
dtype: int64
```

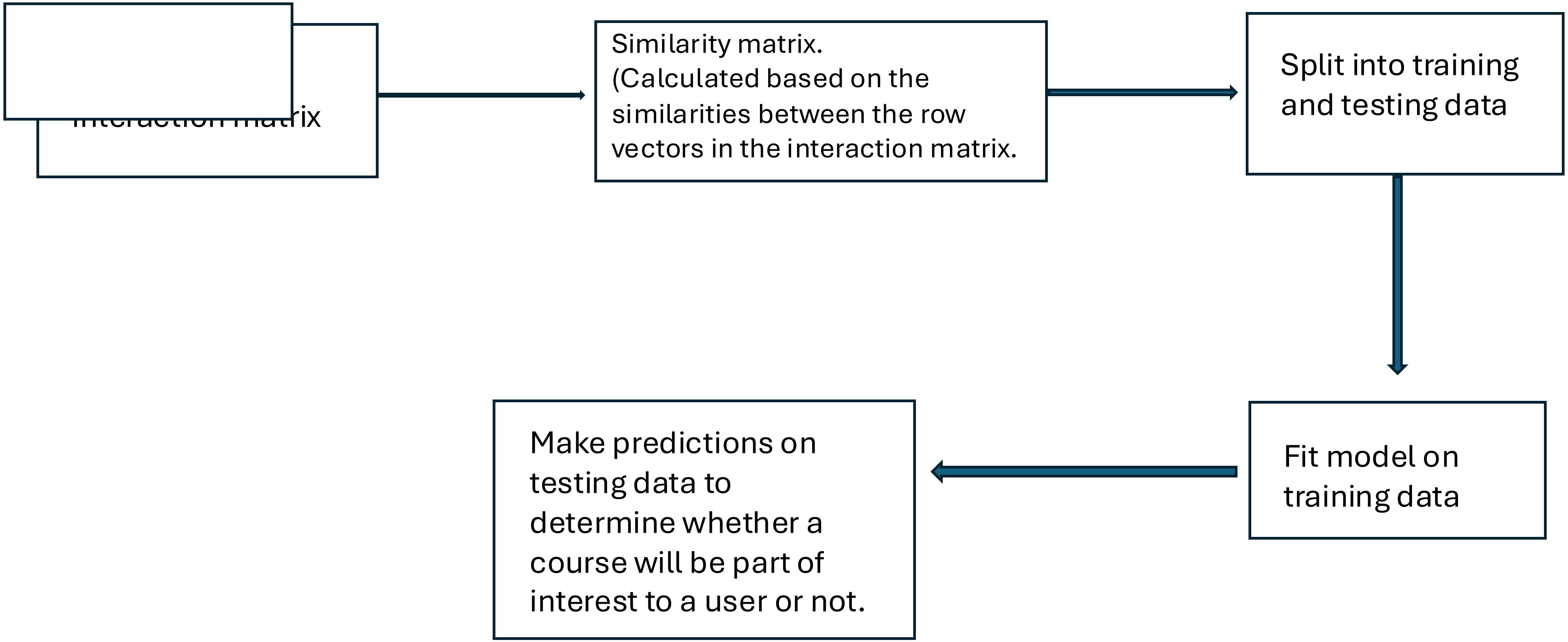
- We get the following output.
- It is a data frame listing the user and all the courses and a boolean value indicating whether it is recommended to the suer or not.
- I input a value of 20 clusters.

- This image shows us that on average 6.9 new courses were recommended to each user.
- We can also see the top 10 most commonly recommended courses.

Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based Recommender System



Interaction matrix

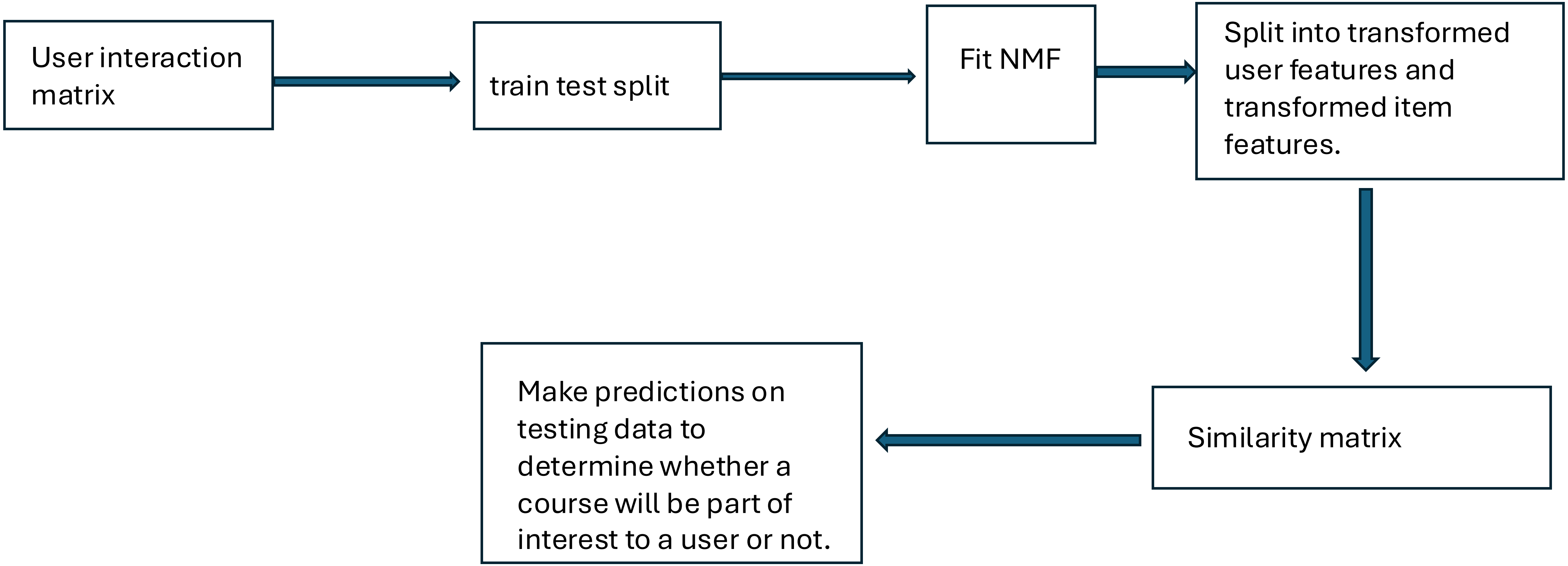
Similarity matrix.
(Calculated based on the similarities between the row vectors in the interaction matrix.

Split into training and testing data

Fit model on training data

Make predictions on testing data to determine whether a course will be part of interest to a user or not.

Flowchart of NMF based Recommender System



User interaction matrix

train test split

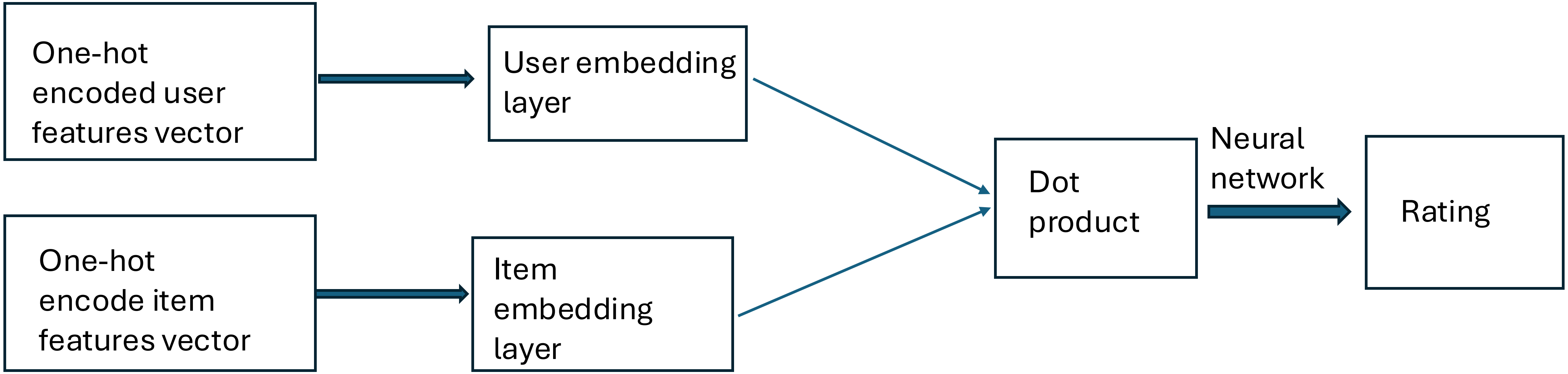
Fit NMF

Split into transformed user features and transformed item features.

Similarity matrix

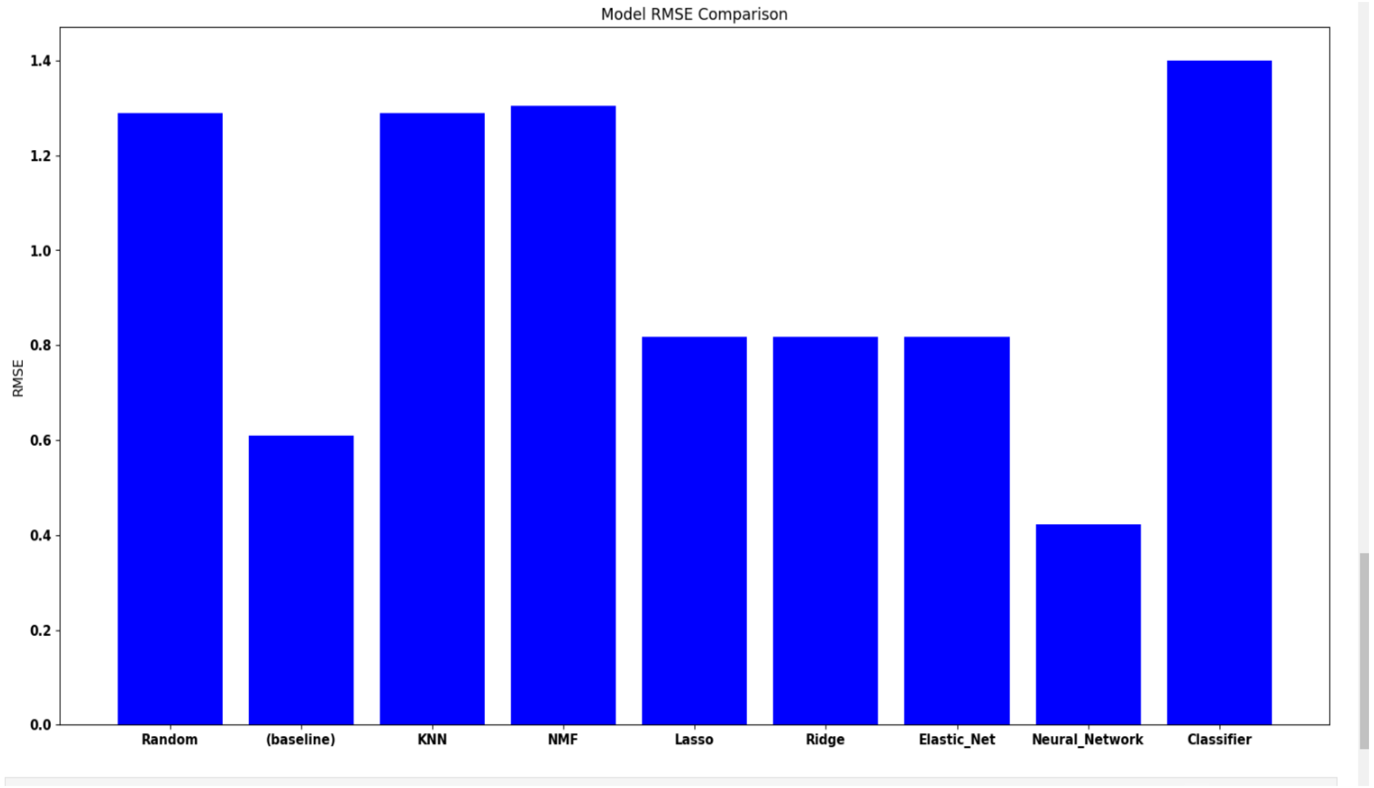
Make predictions on testing data to determine whether a course will be part of interest to a user or not.

Flowchart of Neural Network Embedding based Recommender System


One-hot encoded user features vector
User embedding layer
One-hot encode item features vector
Item embedding layer
Dot product
Neural network
Rating

Compare the Performance of Collaborative-filtering Models

We see that the Neural Network seems to be the best collaborative filtering model.



Conclusion

- We can see that the content based recommender system using unsupervised learning seems to give better more conclusive results.
    - I found it easier to use and the output of a data frame containing recommended courses is more conclusive.
- In the collaborative filtering based recommender system using supervised learning we can see that the neural network embedding has the best performance.
- Data such as the data used for this project is very dense and we need to choose the right method based on the density of the data.
- In this case, due to efficiently purposes I would choose a unsupervised learning method.
- One thing to note is that there seem to be higher RMSE values for the supervised learning methods, this could be due to certain data processing errors that may have occurred during the data cleaning process.

Appendix

- All code was taken from the IBM Skills Network Labs notebooks provided during the course.