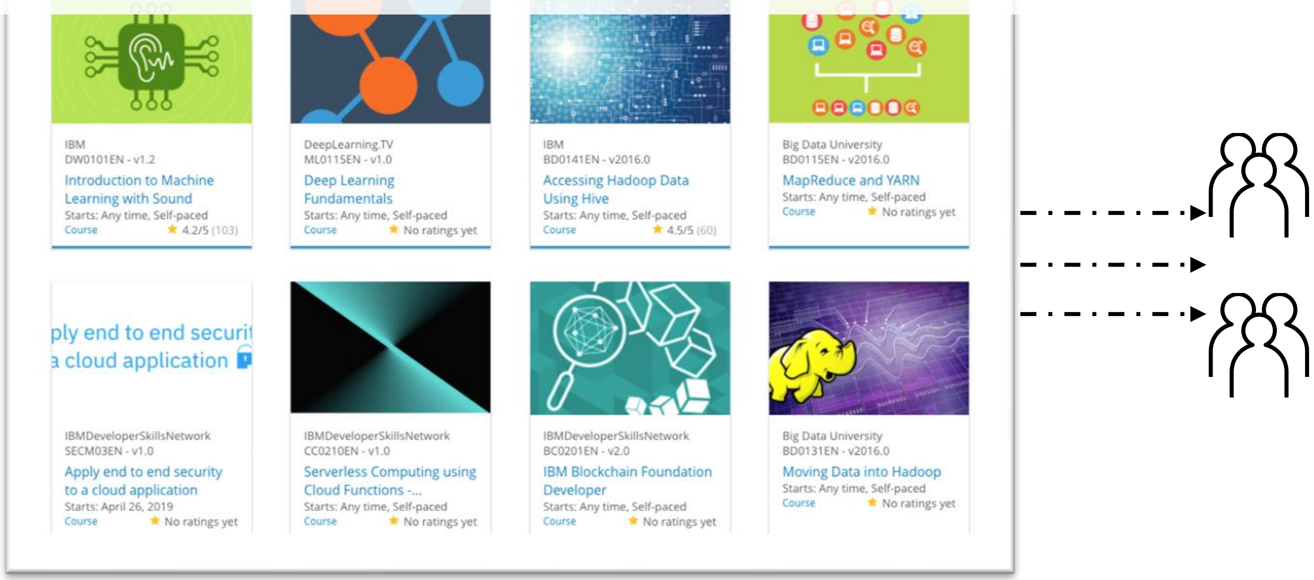
Build a Personalized Online Course Recommender System with Machine Learning

Azlaan Ranjha
31/08/2023



IBM
DW0101EN - v1.2
Introduction to Machine Learning with Sound
Starts: Any time, Self-paced
Course ★ 4.2/5 (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 YARN
Starts: Any time, Self-paced
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
SECM03EN - v1.0
Apply end to end security to a cloud application
Starts: April 26, 2019
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
CC0210EN - v1.0
Serverless Computing using Cloud Functions -...
Starts: Any time, Self-paced
Course ★ No ratings yet

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

Big Data University
BD0131EN - v2016.0
Moving Data into Hadoop
Starts: Any time, Self-paced
Course ★ No ratings yet

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

- Project background and context

Acting as a new machine learning engineer, I have been given the task of leading a recommender system PoC so that we can enhance learner experience.

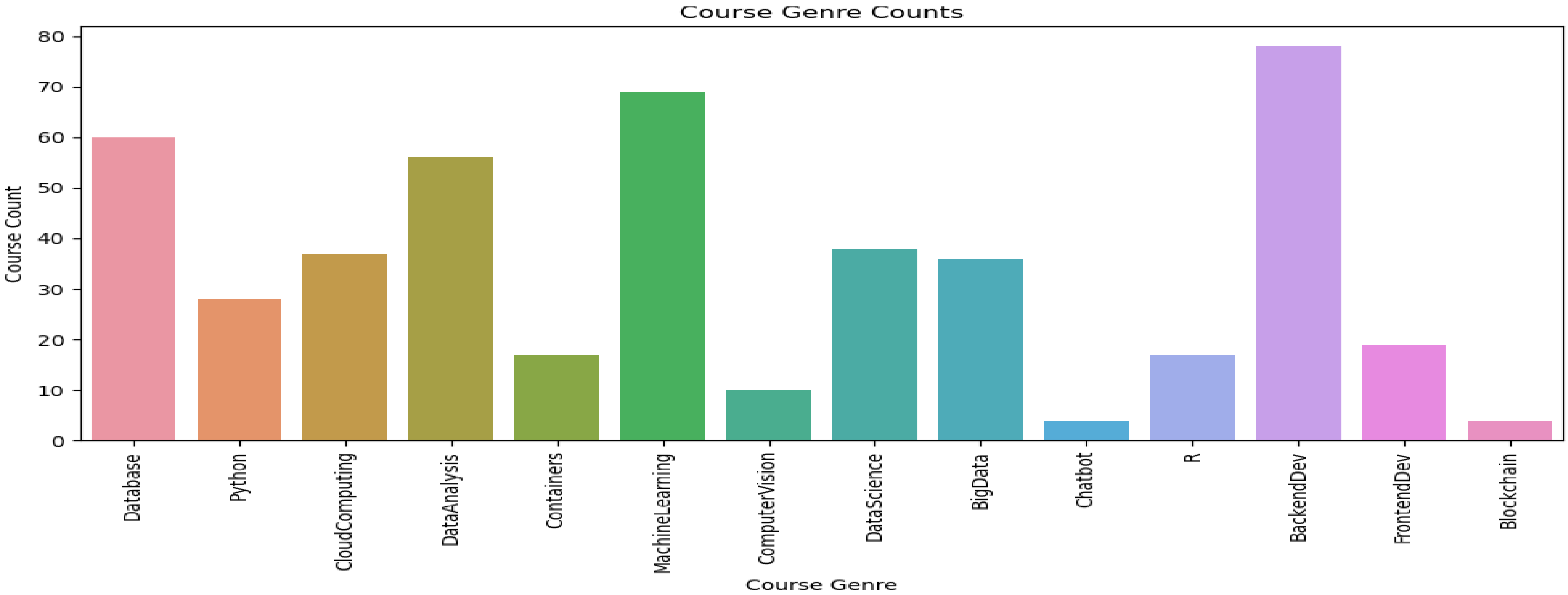
- Problem states and hypotheses

Problem Statement: Developing an efficient recommender system for AI Training Room's diverse course offerings to enhance learner engagement and potentially drive revenue growth.

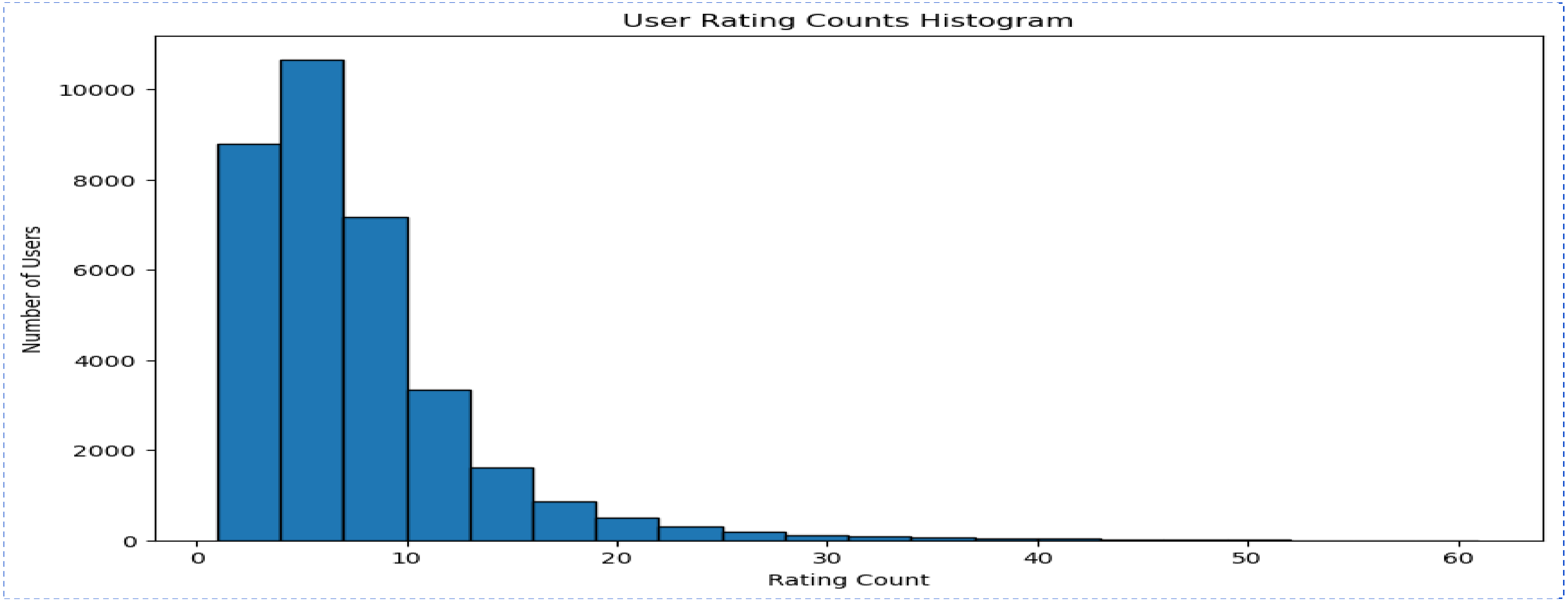
Hypothesis: Implementing personalized recommendations will improve course enrollment rates and user satisfaction, positively impacting both engagement and revenue.

Exploratory Data Analysis

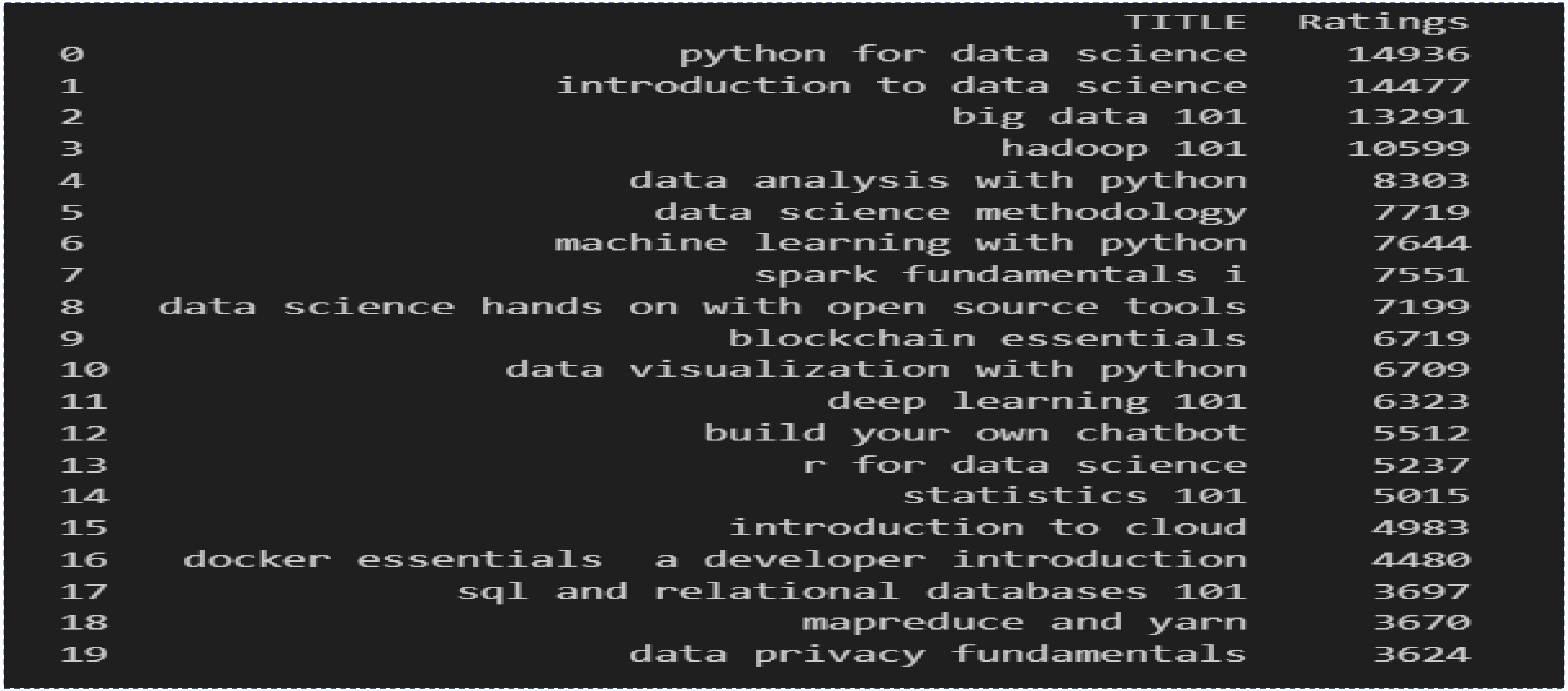
Course counts per genre



Course enrollment distribution

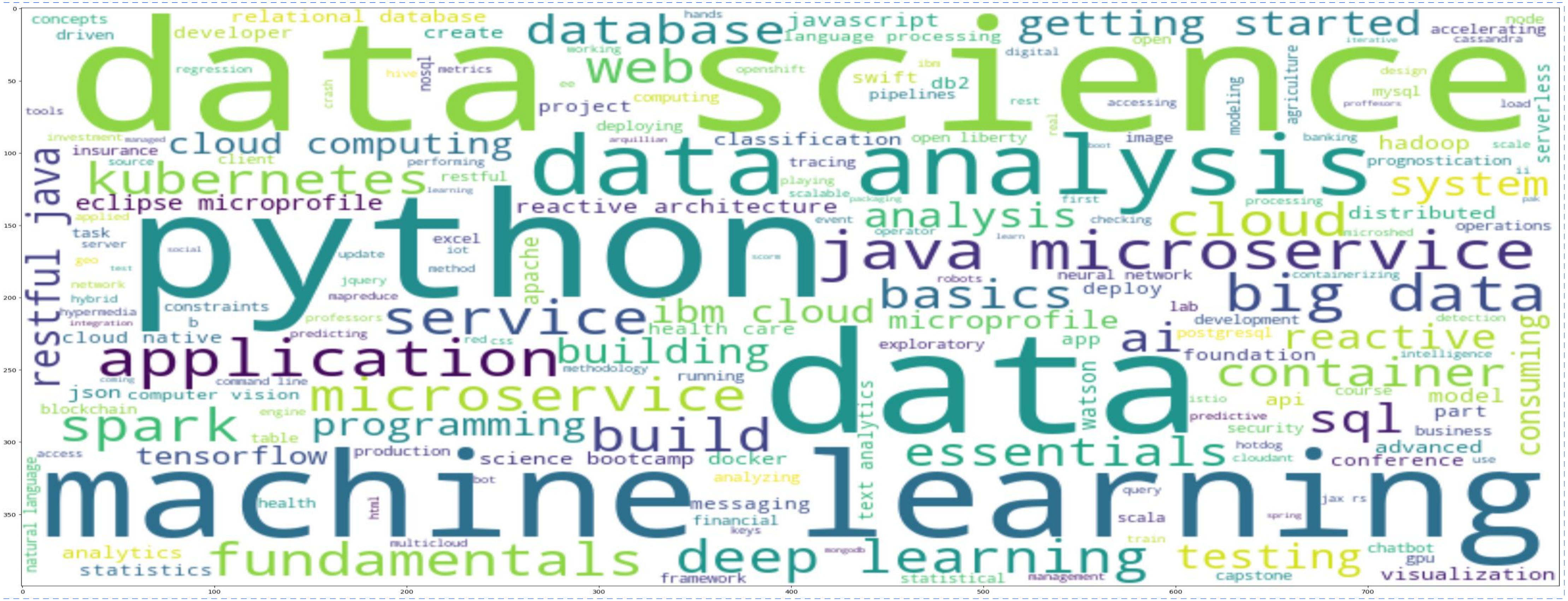


20 most popular courses

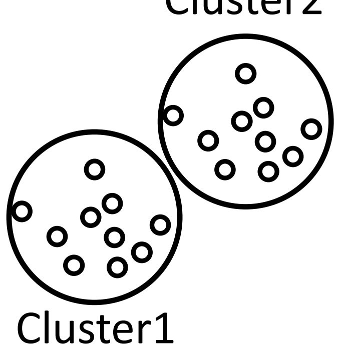


```
                                           TITLE  Ratings
0                        python for data science    14936
1                   introduction to data science    14477
2                                   big data 101    13291
3                                     hadoop 101    10599
4                      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
```

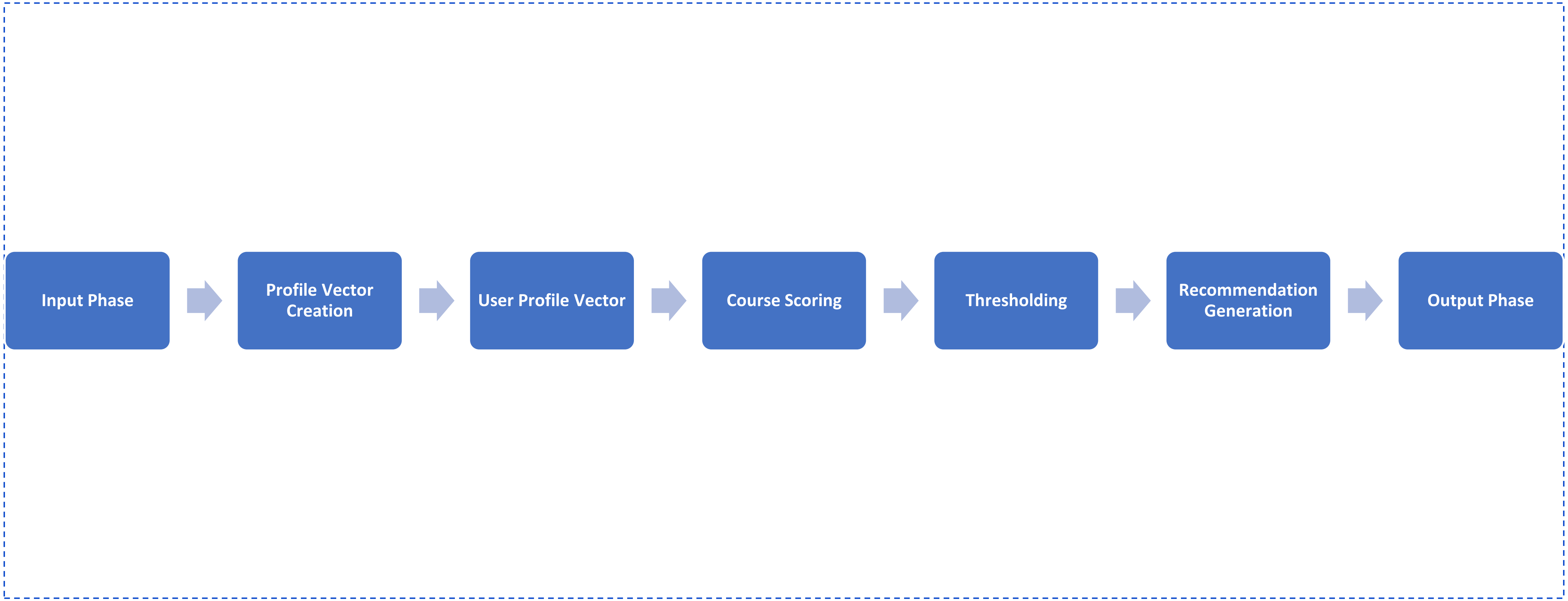
Word cloud of course titles



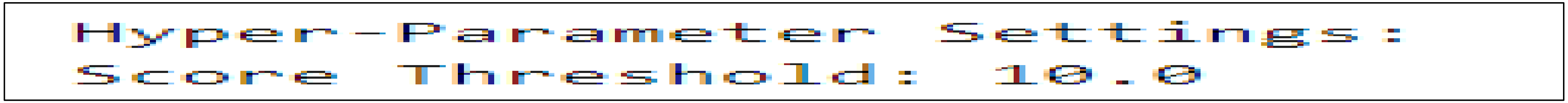
Content-based Recommender System using Unsupervised Learning



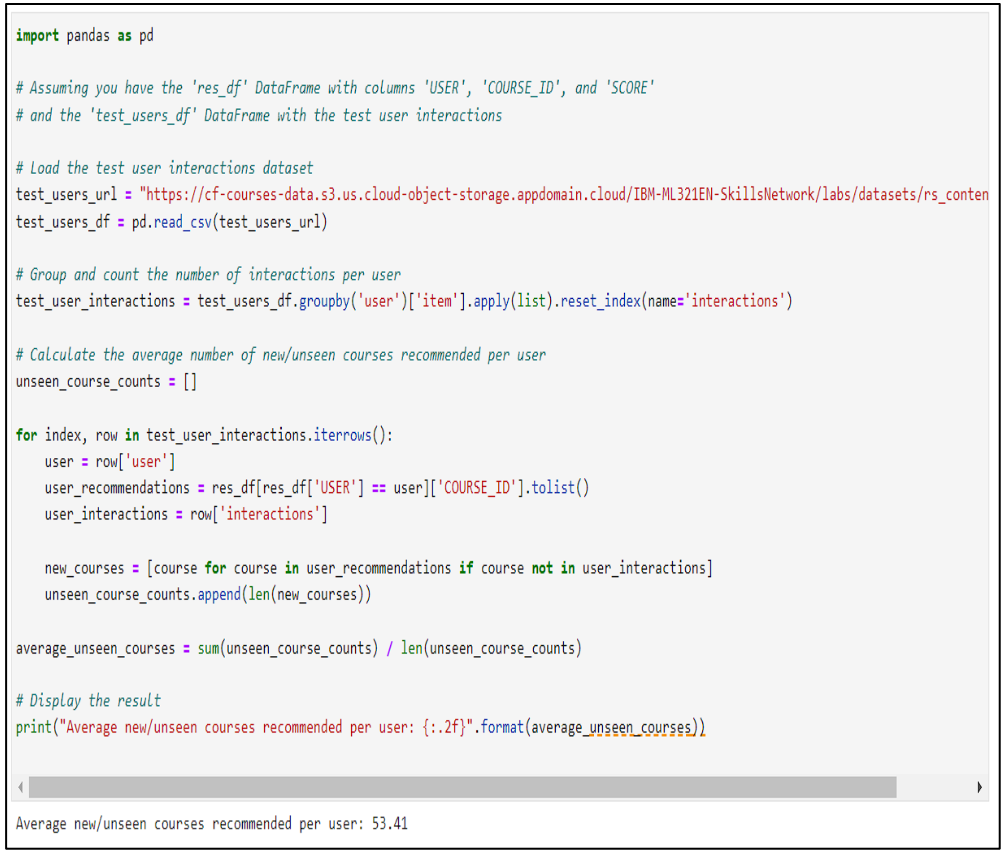
Flowchart of content-based recommender system using user profile and course genres



Evaluation results of user profile-based recommender system

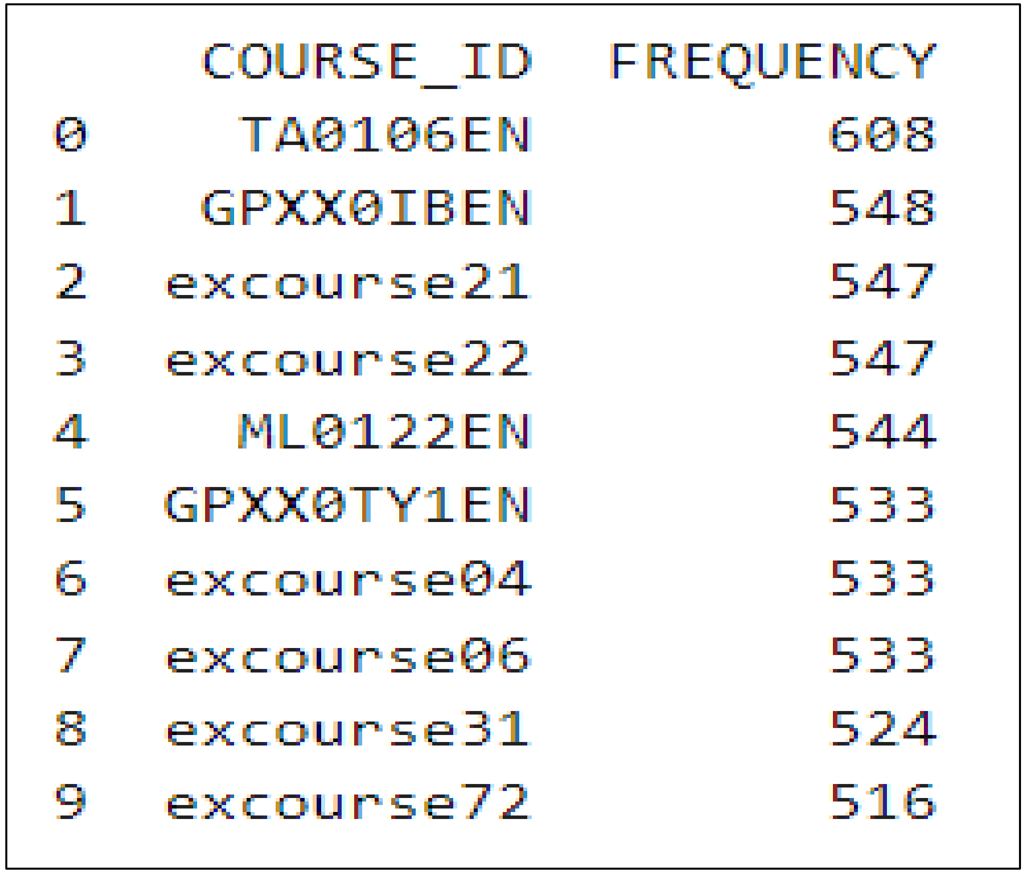


```
Hyper-Parameter Settings:
Score Threshold: 10.0
```



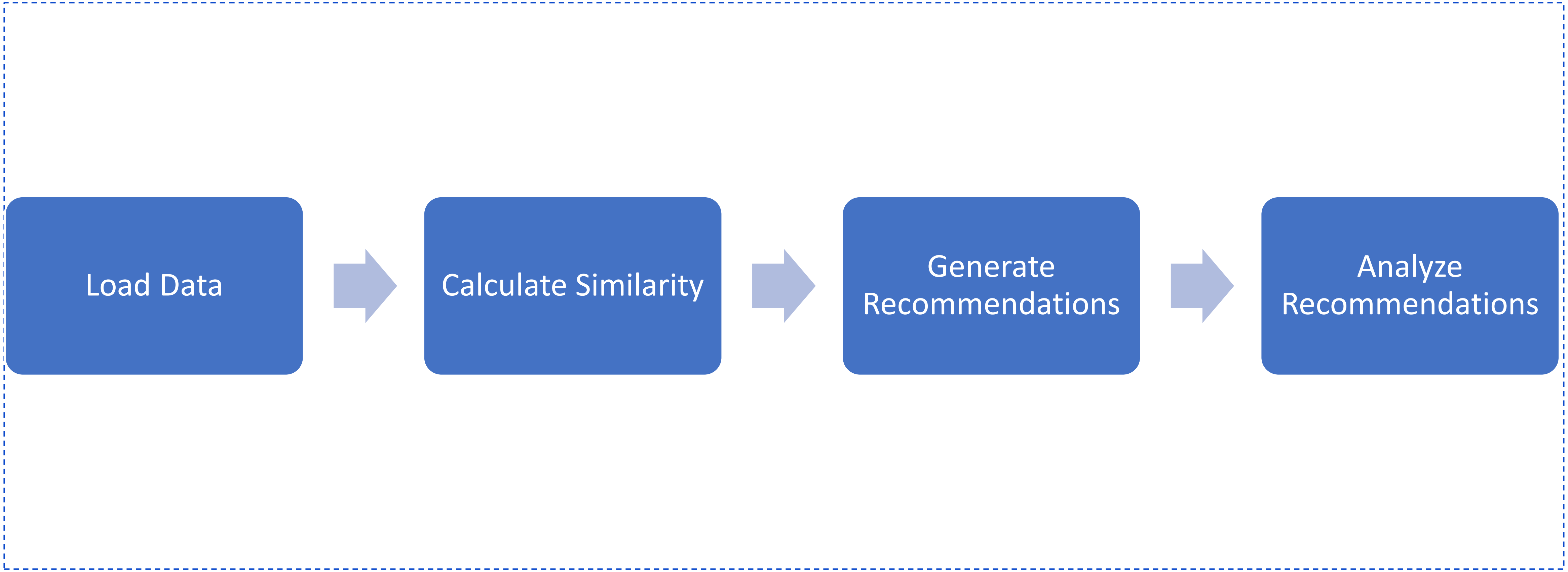
```python
import pandas as pd

# Assuming you have the 'res_df' DataFrame with columns 'USER', 'COURSE_ID', and 'SCORE'
# and the 'test_users_df' DataFrame with the test user interactions

# Load the test user interactions dataset
test_users_url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-ML321EN-SkillsNetwork/labs/datasets/rs_conten
test_users_df = pd.read_csv(test_users_url)

# Group and count the number of interactions per user
test_user_interactions = test_users_df.groupby('user')['item'].apply(list).reset_index(name='interactions')

# Calculate the average number of new/unseen courses recommended per user
unseen_course_counts = []

for index, row in test_user_interactions.iterrows():
    user = row['user']
    user_recommendations = res_df[res_df['USER'] == user]['COURSE_ID'].tolist()
    user_interactions = row['interactions']

    new_courses = [course for course in user_recommendations if course not in user_interactions]
    unseen_course_counts.append(len(new_courses))

average_unseen_courses = sum(unseen_course_counts) / len(unseen_course_counts)

# Display the result
print("Average new/unseen courses recommended per user: {:.2f}".format(average_unseen_courses))
```

```
Average new/unseen courses recommended per user: 53.41
```

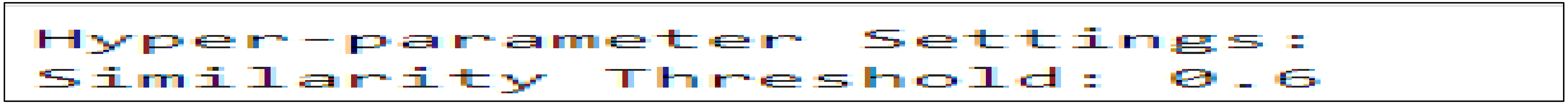


```
     COURSE_ID  FREQUENCY
0     TA0106EN        608
1    GPXX0IBEN        548
2   excourse21        547
3   excourse22        547
4     ML0122EN        544
5   GPXX0TY1EN        533
6   excourse04        533
7   excourse06        533
8   excourse31        524
9   excourse72        516
```

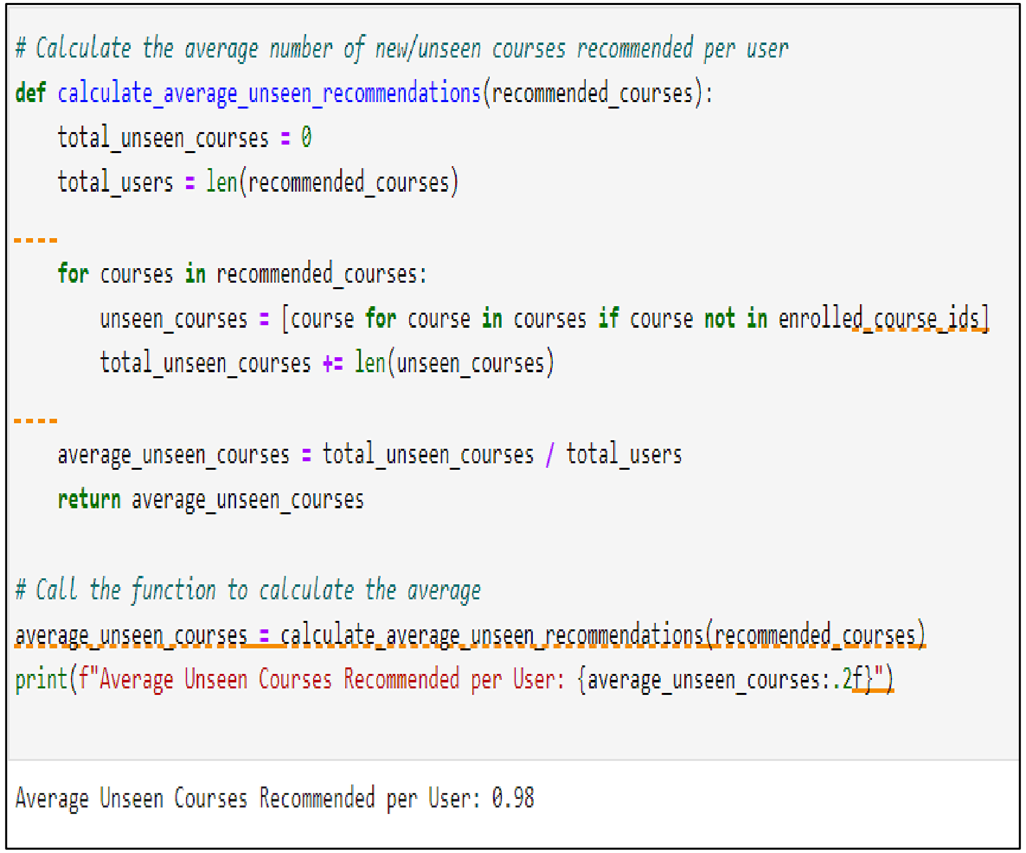
Flowchart of content-based recommender system using course similarity



Evaluation results of course similarity based recommender system

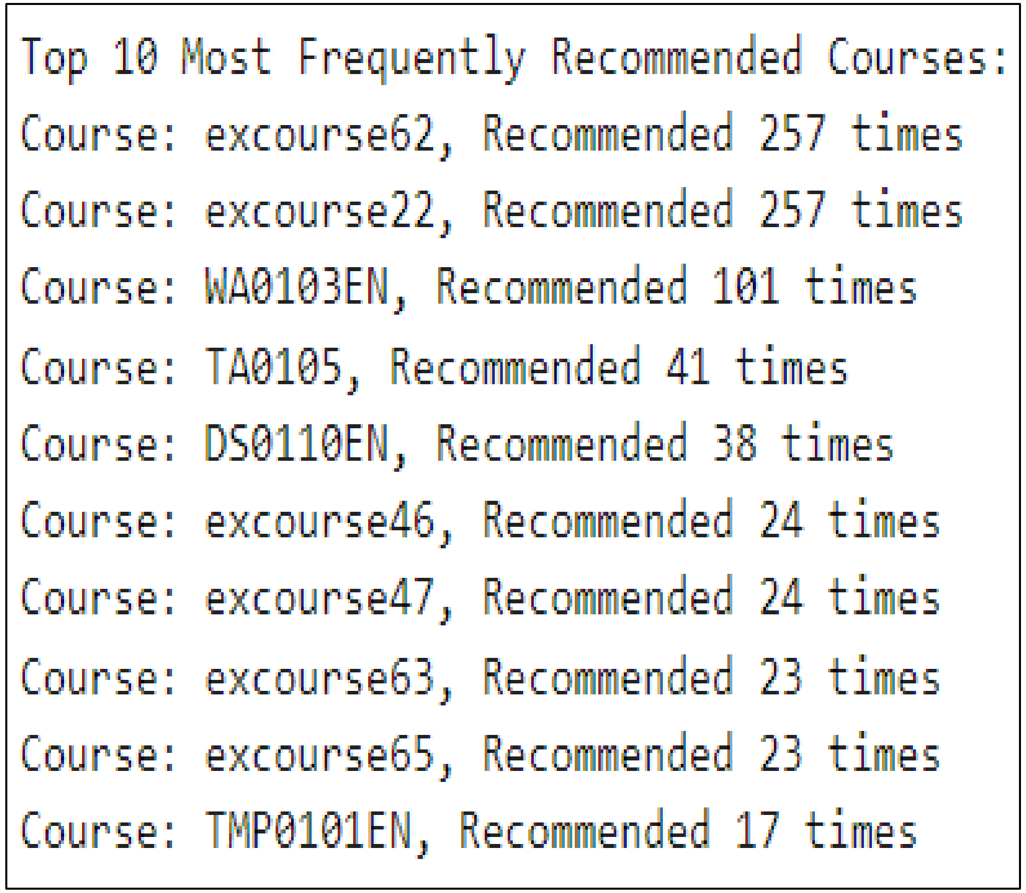


```
Hyper-parameter Settings:
Similarity Threshold: 0.6
```



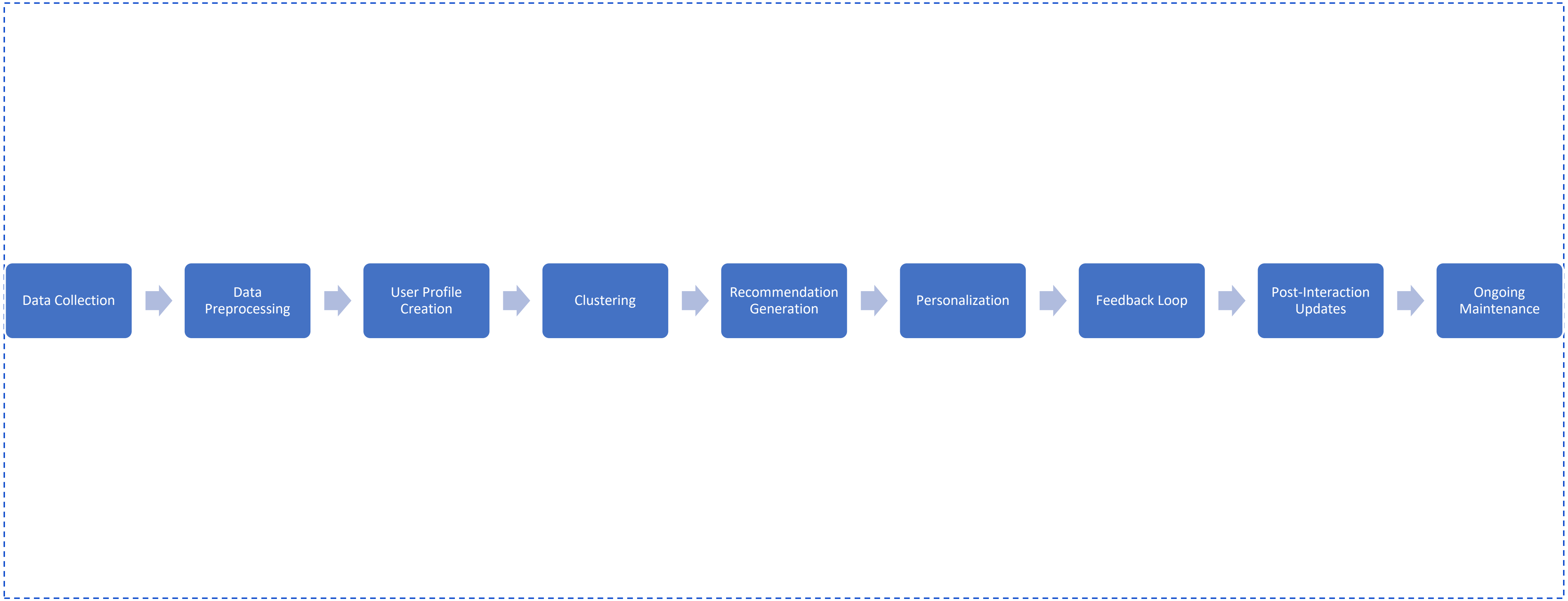
```python
# Calculate the average number of new/unseen courses recommended per user
def calculate_average_unseen_recommendations(recommended_courses):
    total_unseen_courses = 0
    total_users = len(recommended_courses)

    for courses in recommended_courses:
        unseen_courses = [course for course in courses if course not in enrolled_course_ids]
        total_unseen_courses += len(unseen_courses)

    average_unseen_courses = total_unseen_courses / total_users
    return average_unseen_courses

# Call the function to calculate the average
average_unseen_courses = calculate_average_unseen_recommendations(recommended_courses)
print(f"Average Unseen Courses Recommended per User: {average_unseen_courses:.2f}")
```

```
Average Unseen Courses Recommended per User: 0.98
```



```
Top 10 Most Frequently Recommended Courses:
Course: excourse62, Recommended 257 times
Course: excourse22, Recommended 257 times
Course: WA0103EN, Recommended 101 times
Course: TA0105, Recommended 41 times
Course: DS0110EN, Recommended 38 times
Course: excourse46, Recommended 24 times
Course: excourse47, Recommended 24 times
Course: excourse63, Recommended 23 times
Course: excourse65, Recommended 23 times
Course: TMP0101EN, Recommended 17 times
```

Flowchart of clustering-based recommender system



Evaluation results of clustering-based recommender system

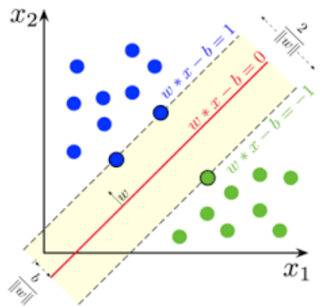
Your hyper-parameter settings, such as a score or similarity threshold

Note if you have tried multiple hyper-parameters, you may show your results in a grouped bar chart

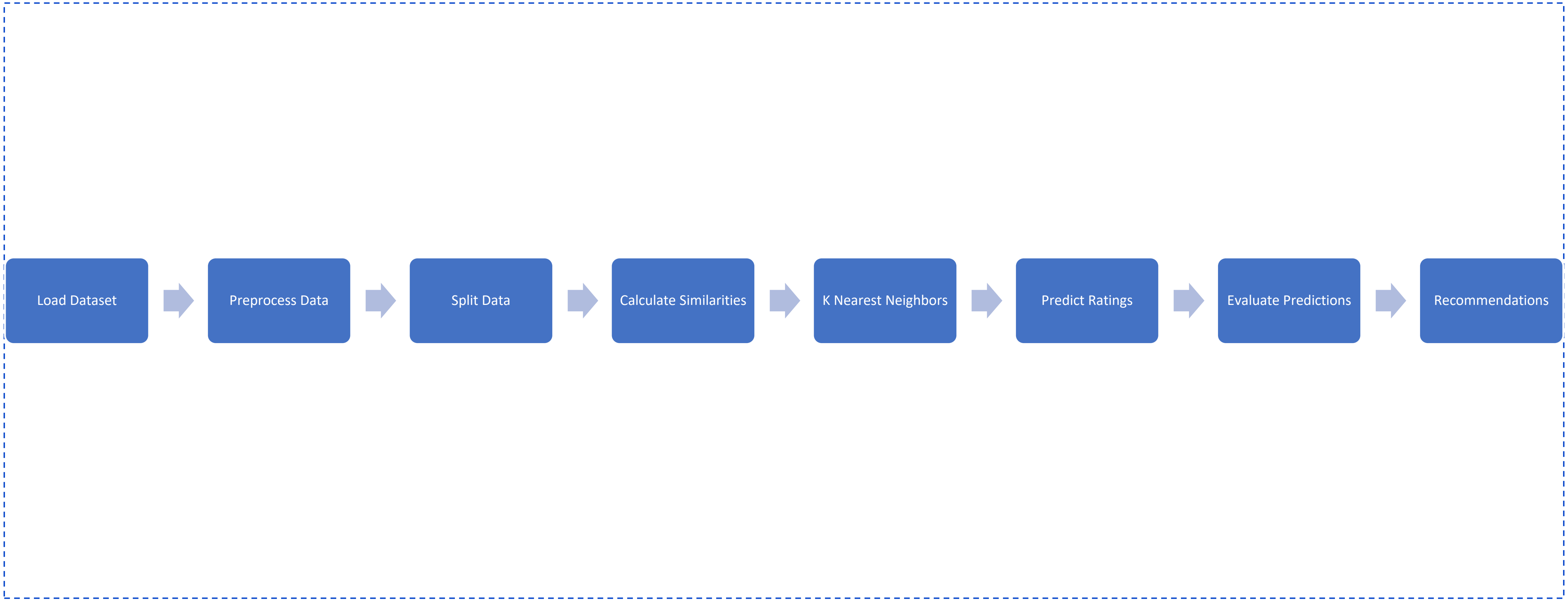
On average, how many new/unseen courses have been recommended per user (in the test user dataset)

What are the most frequently recommended courses? Return the top-10 commonly recommended courses

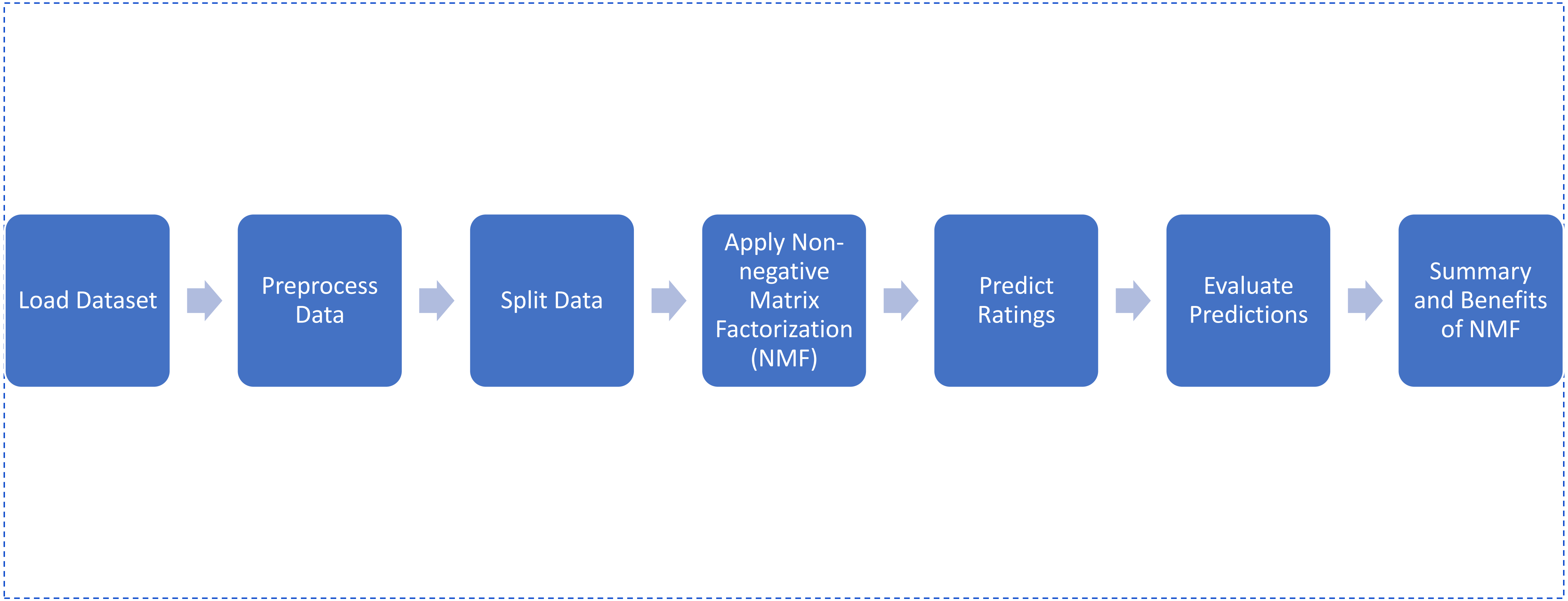
Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system



Flowchart of NMF based recommender system



Load Dataset

Preprocess Data

Split Data

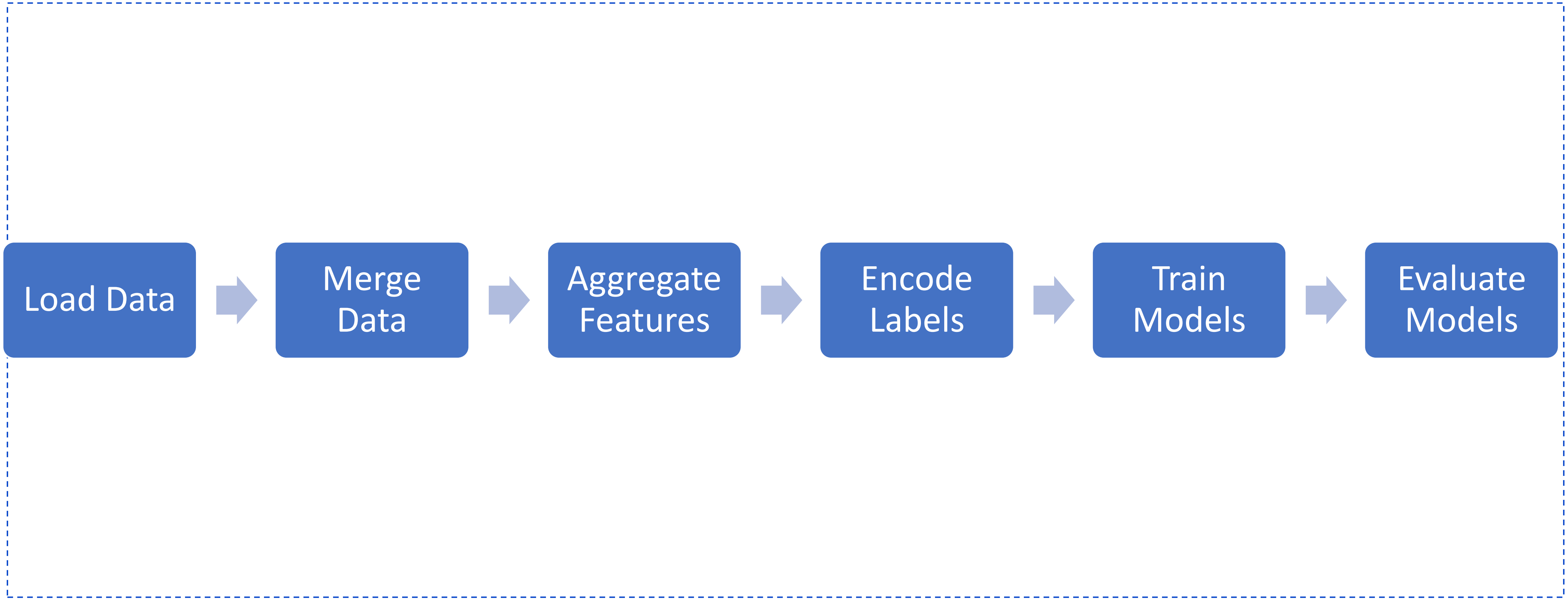
Apply Non-negative Matrix Factorization (NMF)

Predict Ratings

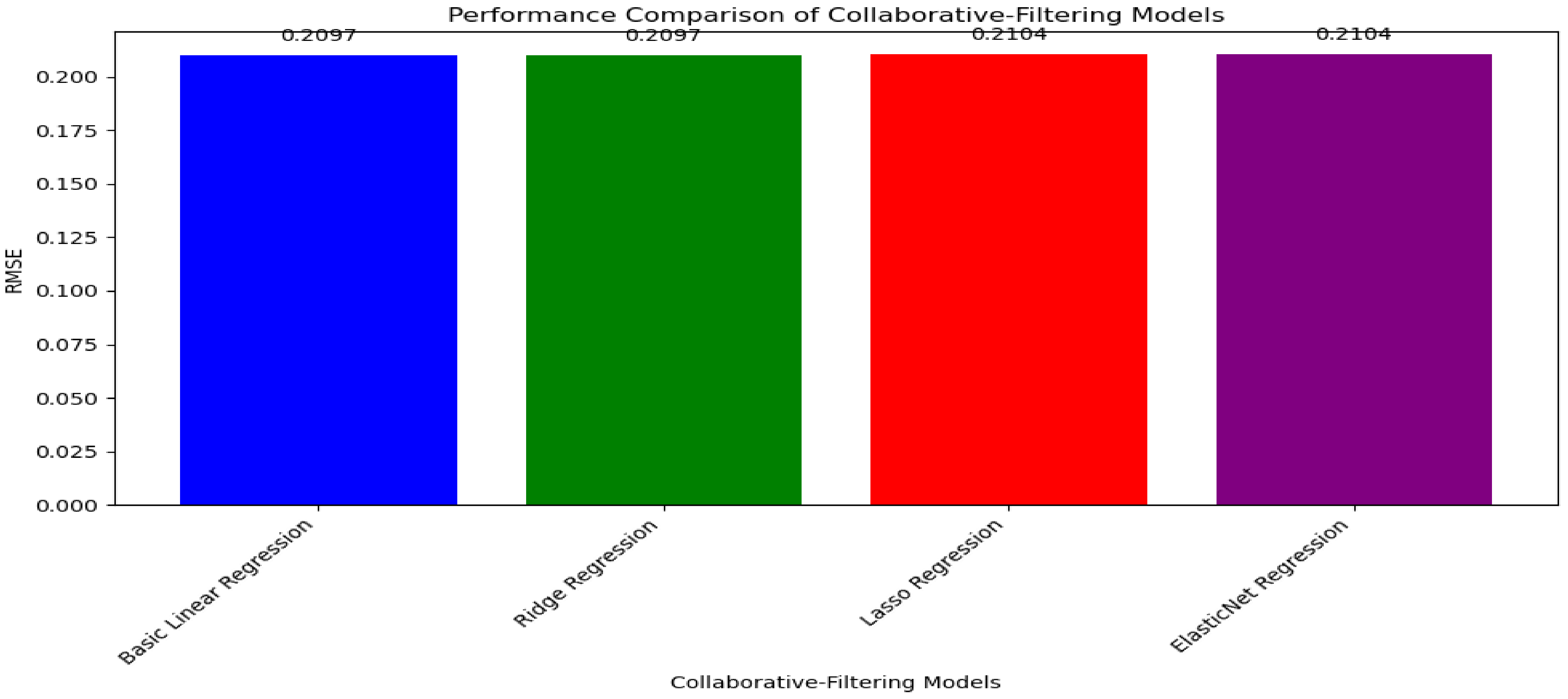
Evaluate Predictions

Summary and Benefits of NMF

Flowchart of Neural Network Embedding based recommender system



Compare the performance of collaborative-filtering models



Conclusions

- We can conclude, that the ratings of the courses had a direct impact on the course popularity. Centre of student interest were topics such as ML, Python and Data Science etc.
- Hyperparameters had a strong impact on the frequency of recommendations of a particular course.
- User-based content had a tendency of being recommended more strongly than content-based recommendations.
- Recommender Systems using Neural networks and Regression Methods etc, although are time consuming but are most accurate and consistent.