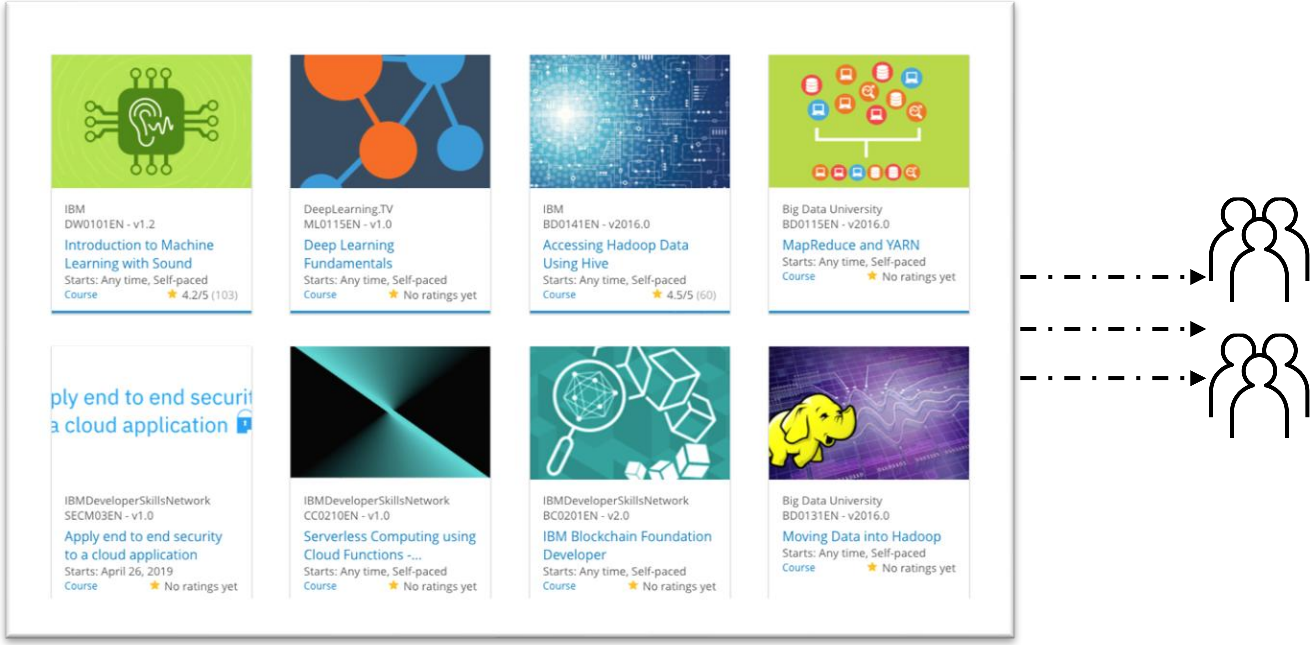
Online Course Recommender System with Machine Learning

Mohammed Khalil
July 11, 2025



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

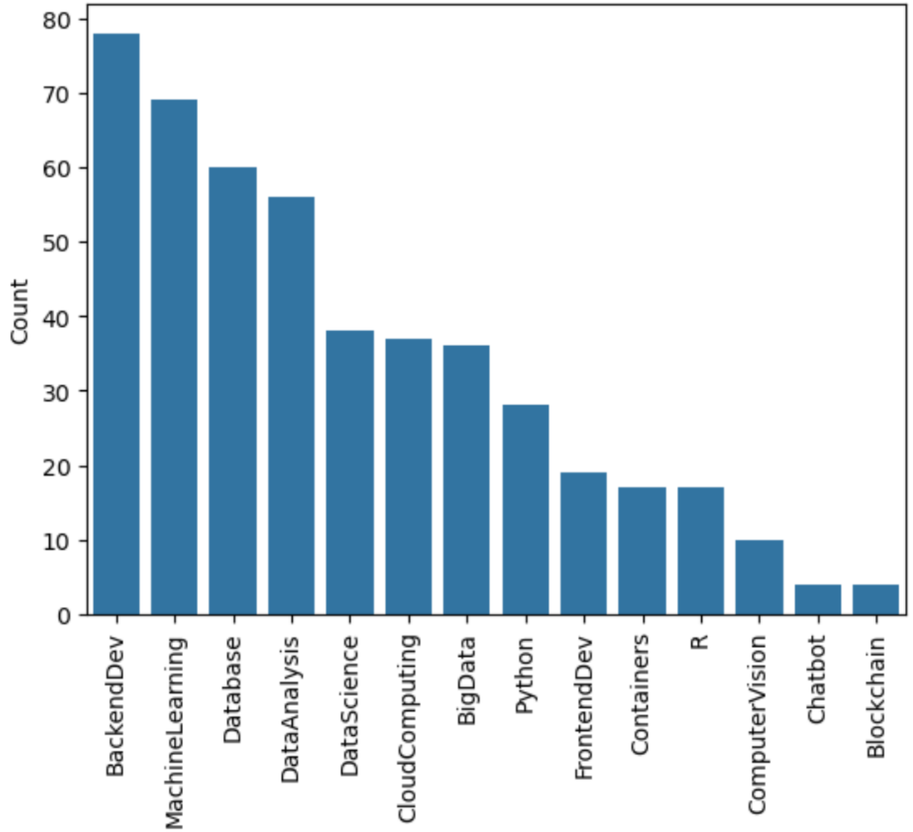
Introduction

- Project background and context
  - Coursera is an online course provider with over 10,000 courses. This can it can make it difficult for users to quickly find courses relevant for them.
  - In this project, we build a course recommender system using machine learning.
- Problem states and hypotheses
  - Help learners to easily find courses they might be interested in.
  - Increase user engagement on the platform.

Exploratory Data Analysis

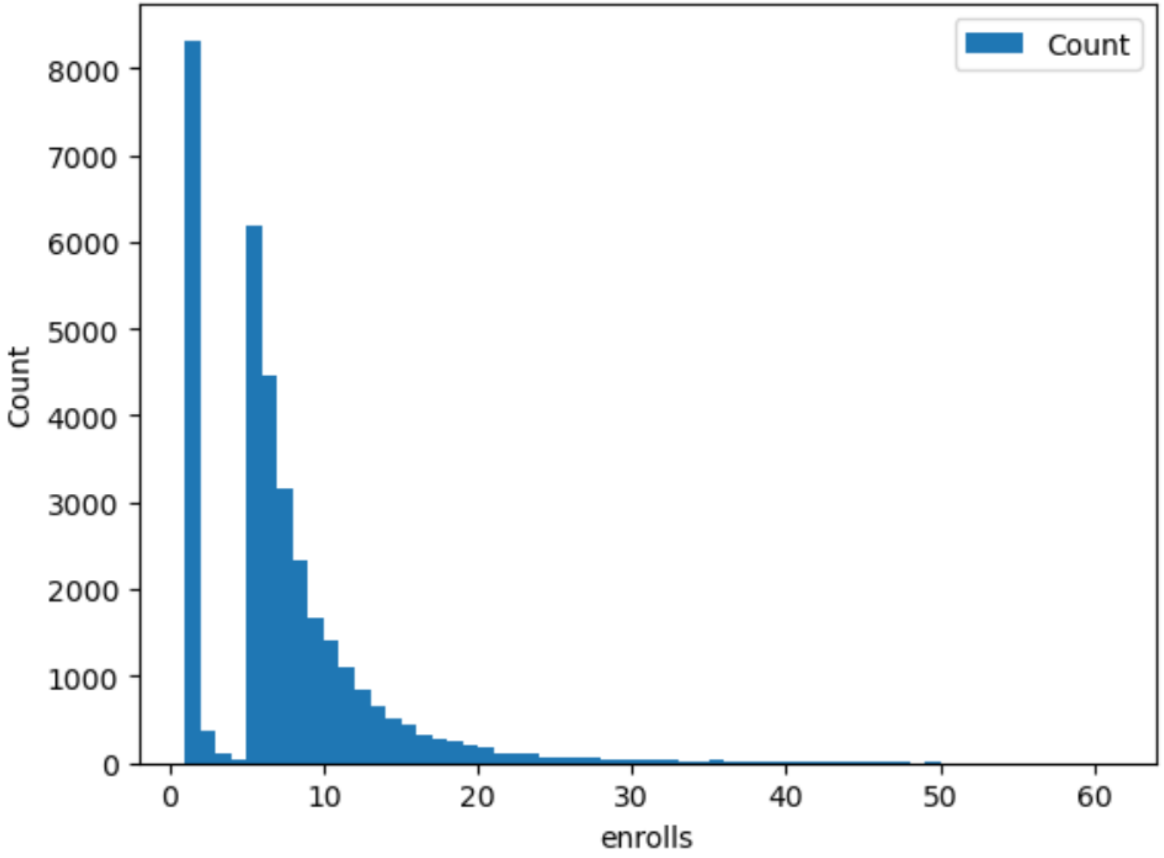
Course counts per genre

- The dataset contains 307 courses in 14 genres.
- The bar plot shows the number of courses per genre.
- The genres with the highest number of courses (60 or more courses) are Backend Dev, Machine Learning, and Databases.



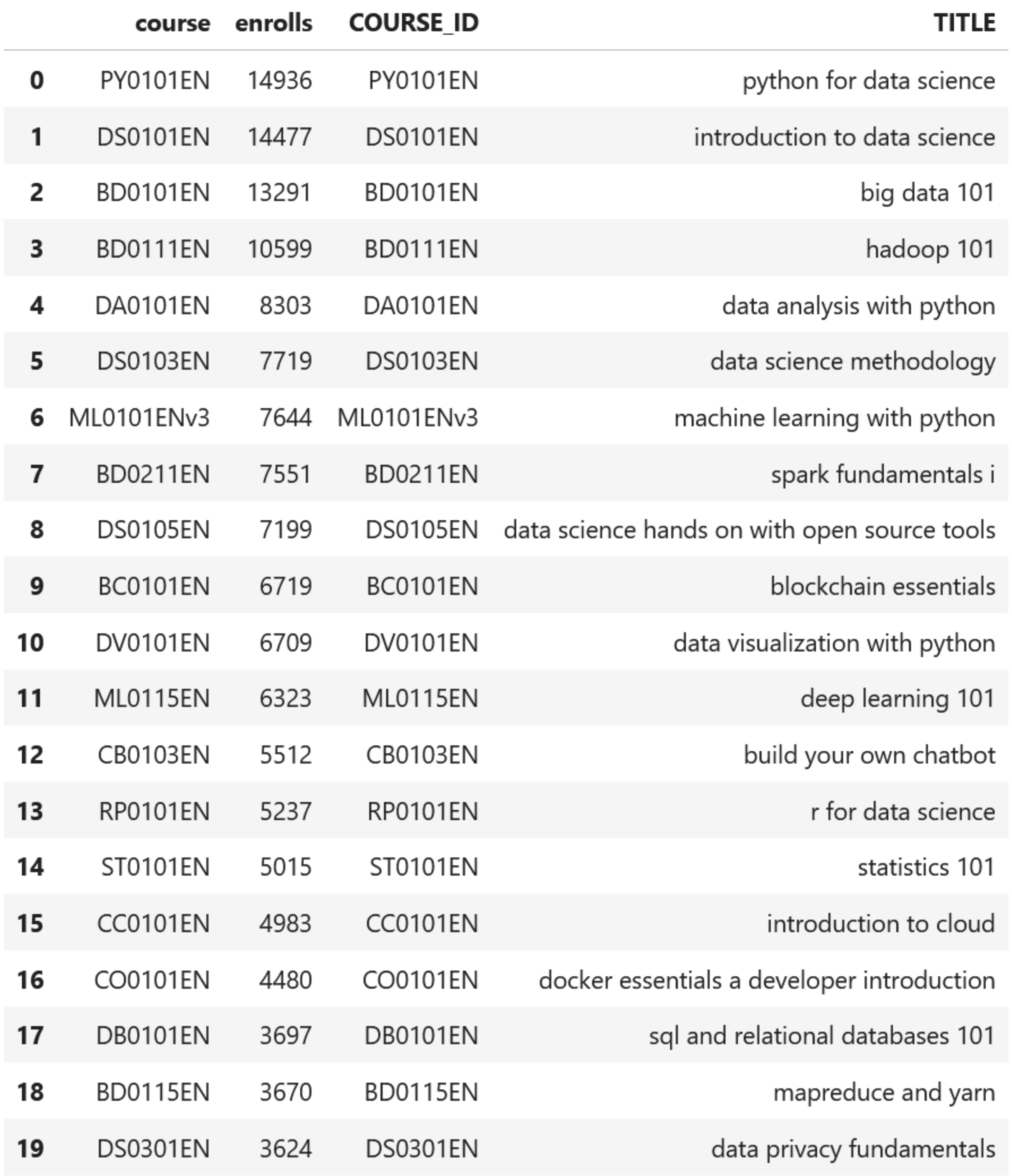
Course enrollment distribution

- Course enrollment distributions, showing how many users enrolled in a given number of courses.
- The vast majority of users enroll in fewer than 10 courses.
- Out of 33,901 users, over 8,000 are enrolled in one course.
- Few users enroll in 2-4 courses.
- Most users enroll in 5 or more courses.



20 most popular courses

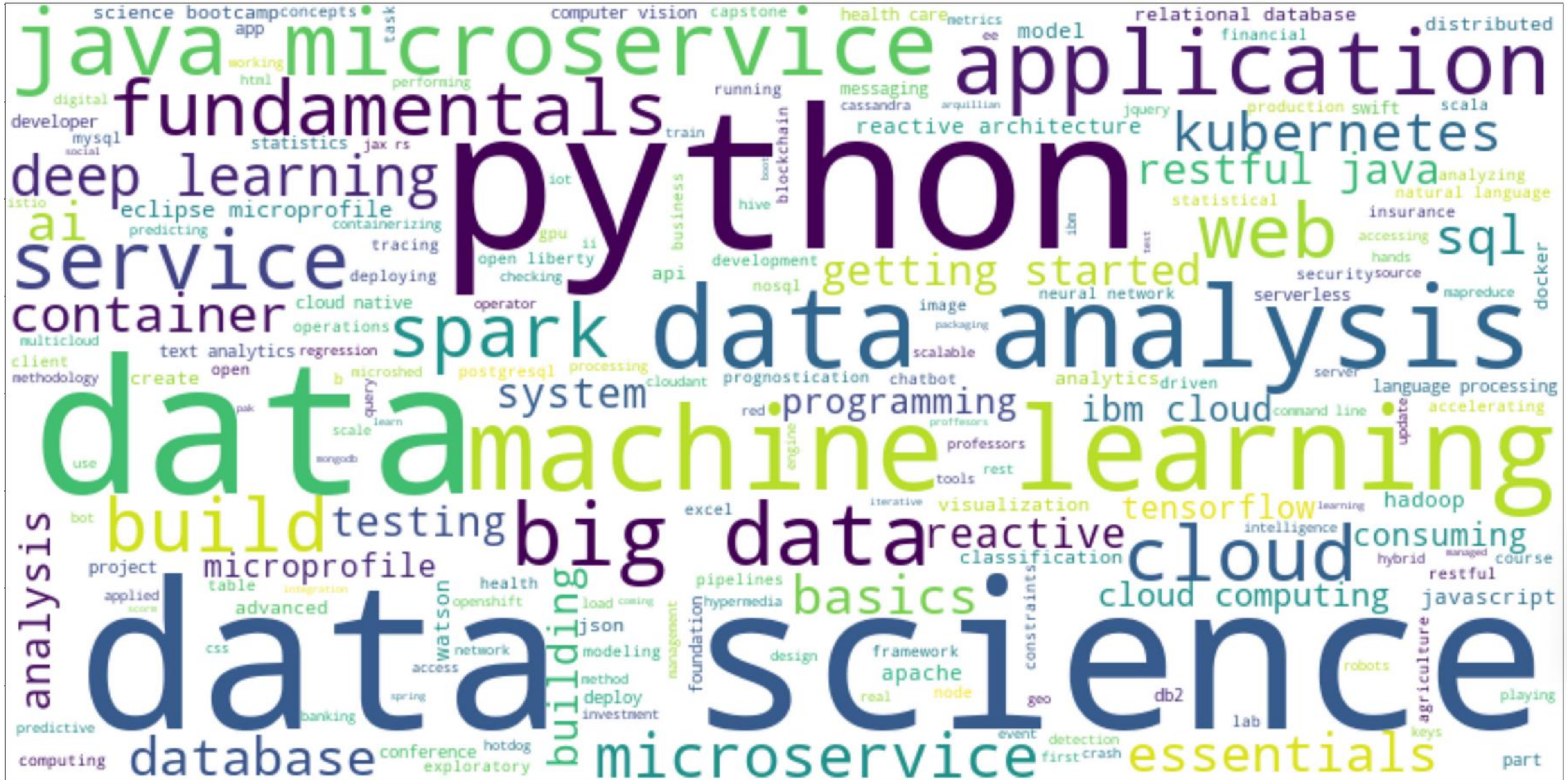
- List of the most popular 20 courses.
- Most of the popular courses are related to data, machine learning, and python.
- The percentage of users enrolled in those 20 courses is 63.3%.



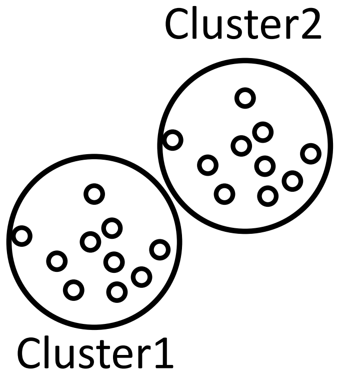
	course	enrolls	COURSE_ID	TITLE
0	PY0101EN	14936	PY0101EN	python for data science
1	DS0101EN	14477	DS0101EN	introduction to data science
2	BD0101EN	13291	BD0101EN	big data 101
3	BD0111EN	10599	BD0111EN	hadoop 101
4	DA0101EN	8303	DA0101EN	data analysis with python
5	DS0103EN	7719	DS0103EN	data science methodology
6	ML0101ENv3	7644	ML0101ENv3	machine learning with python
7	BD0211EN	7551	BD0211EN	spark fundamentals i
8	DS0105EN	7199	DS0105EN	data science hands on with open source tools
9	BC0101EN	6719	BC0101EN	blockchain essentials
10	DV0101EN	6709	DV0101EN	data visualization with python
11	ML0115EN	6323	ML0115EN	deep learning 101
12	CB0103EN	5512	CB0103EN	build your own chatbot
13	RP0101EN	5237	RP0101EN	r for data science
14	ST0101EN	5015	ST0101EN	statistics 101
15	CC0101EN	4983	CC0101EN	introduction to cloud
16	CO0101EN	4480	CO0101EN	docker essentials a developer introduction
17	DB0101EN	3697	DB0101EN	sql and relational databases 101
18	BD0115EN	3670	BD0115EN	mapreduce and yarn
19	DS0301EN	3624	DS0301EN	data privacy fundamentals

Word cloud of course titles

- Word cloud showing the most used words in course titles and description.
- Python, data, data science, and machine learning are the most common words.

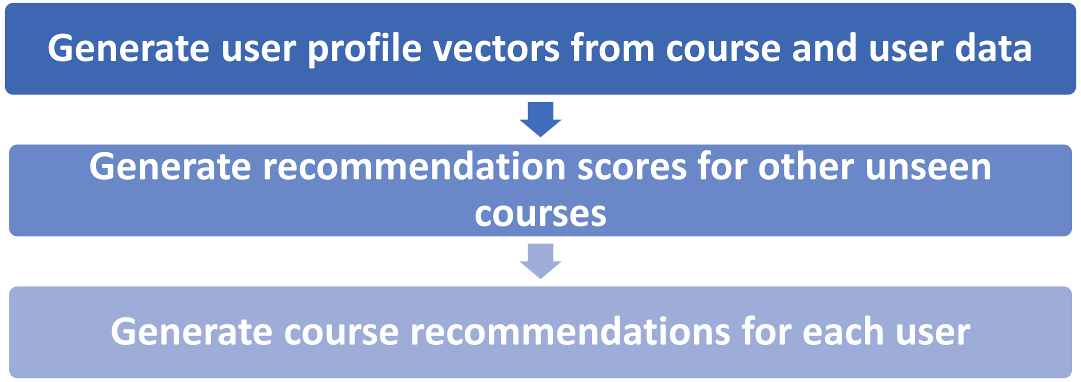


Content-based Recommender System using Unsupervised Learning



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

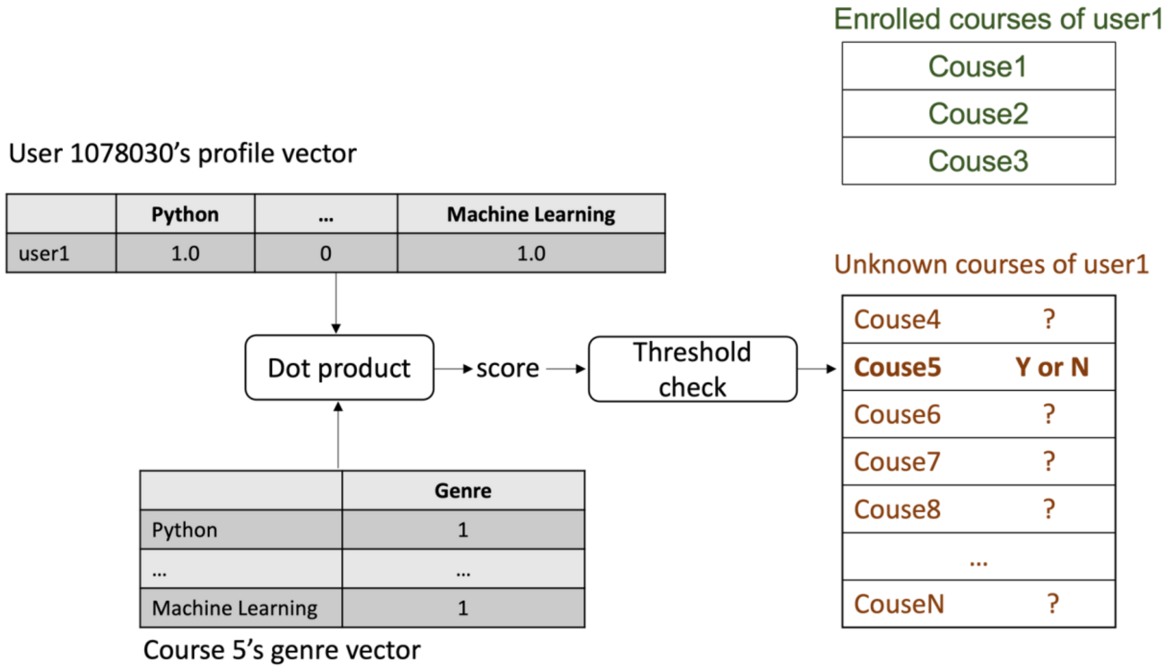
- We recommend courses to users based on which course genres are in their profile.
- We then compute recommendation scores for new unseen courses.
- Finally, recommend the courses with highest scores.



Generate user profile vectors from course and user data

Generate recommendation scores for other unseen courses

Generate course recommendations for each user



User 1078030's profile vector

	Python	...	Machine Learning
user1	1.0	0	1.0

	Genre
Python	1
...	...
Machine Learning	1

Course 5's genre vector

Enrolled courses of user1

Couse1
Couse2
Couse3

Unknown courses of user1

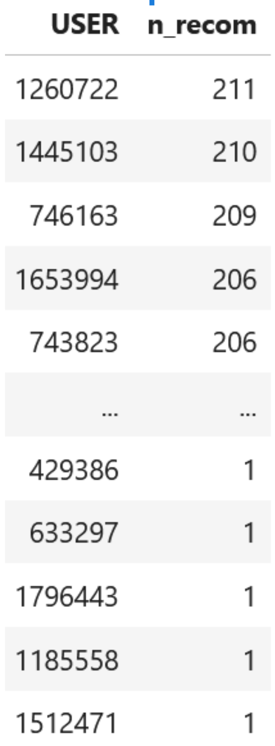
Couse4	?
Couse5	**Y or N**
Couse6	?
Couse7	?
Couse8	?
...	
CouseN	?

Evaluation results of user profile-based recommender system

Place your hyper-parameter settings, such as recommendation score or course similarity thresholds, etc.

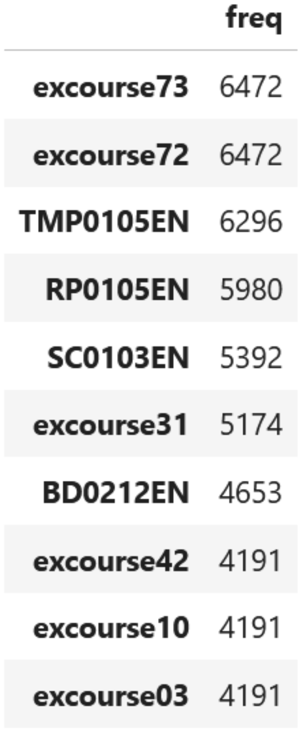
used score_threshold = 25 in the function generate_recommendation_scores()

On average, **23** new/unseen courses have been recommended per user.



USER	n_recom
1260722	211
1445103	210
746163	209
1653994	206
743823	206
...	...
429386	1
633297	1
1796443	1
1185558	1
1512471	1

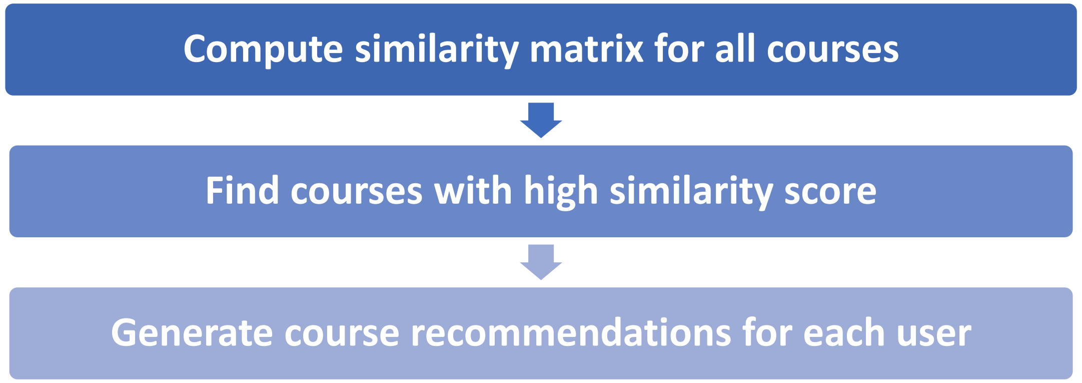
Top-10 most frequently recommended courses across all users.



	freq
excourse73	6472
excourse72	6472
TMP0105EN	6296
RP0105EN	5980
SC0103EN	5392
excourse31	5174
BD0212EN	4653
excourse42	4191
excourse10	4191
excourse03	4191

Flowchart of content-based recommender system using course similarity

- Compute similarity matrix for all courses based on bag of words.
- Find courses with similarity score above a certain threshold.
- Generate recommendations for each user based on the similarity of enrolled courses to unseen courses.



Compute similarity matrix for all courses

Find courses with high similarity score

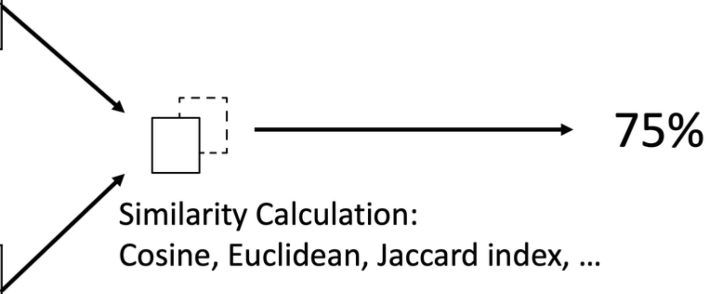
Generate course recommendations for each user

Course 1: “Machine Learning for Everyone”

	machine	learning	for	everyone	beginners
course1	1	1	1	1	0

Course 2: “Machine Learning for Beginners”

	machine	learning	for	everyone	beginners
course2	1	1	1	0	1



Similarity Calculation:
Cosine, Euclidean, Jaccard index, ...

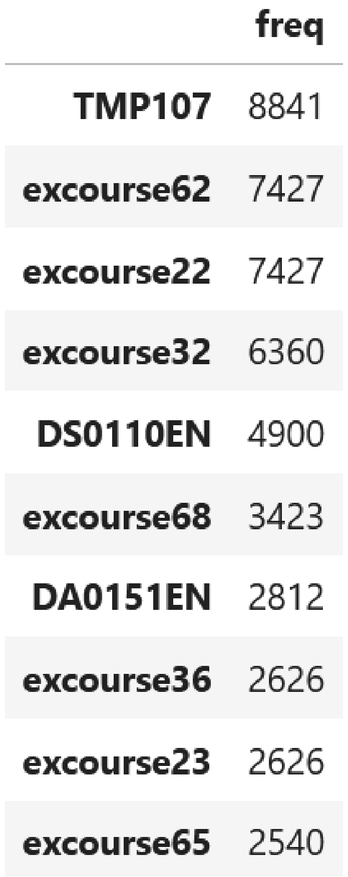
Evaluation results of course similarity based recommender system

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

used similarity threshold = 0.5 in the function generate_recommendations_for_one_user()

On average, **3.2** new/unseen courses have been recommended per user.

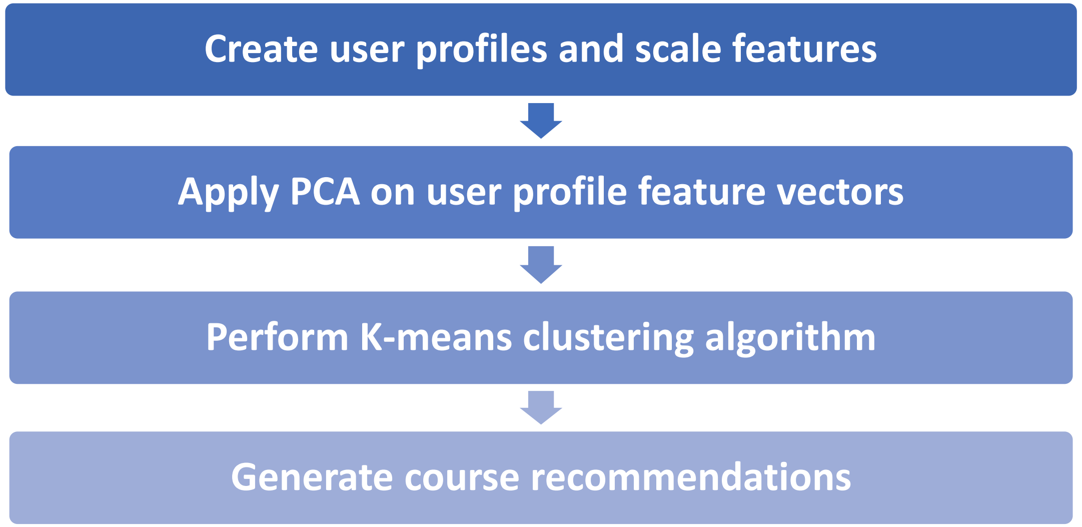
The table shows the top-10 most frequently recommended courses across all users.



	freq
TMP107	8841
excourse62	7427
excourse22	7427
excourse32	6360
DS0110EN	4900
excourse68	3423
DA0151EN	2812
excourse36	2626
excourse23	2626
excourse65	2540

Flowchart of clustering-based recommender system

- Create user profiles based on the genres of enrolled courses, and perform standard scaling to prepare the data.
- Apply PCA on user profile feature vectors to reduce dimensions
- Perform K-means clustering algorithm on the user profile feature vectors, selecting the number of cluster based on inertia and silhouette scores.
- Generate course recommendations based on the popular courses in the same cluster

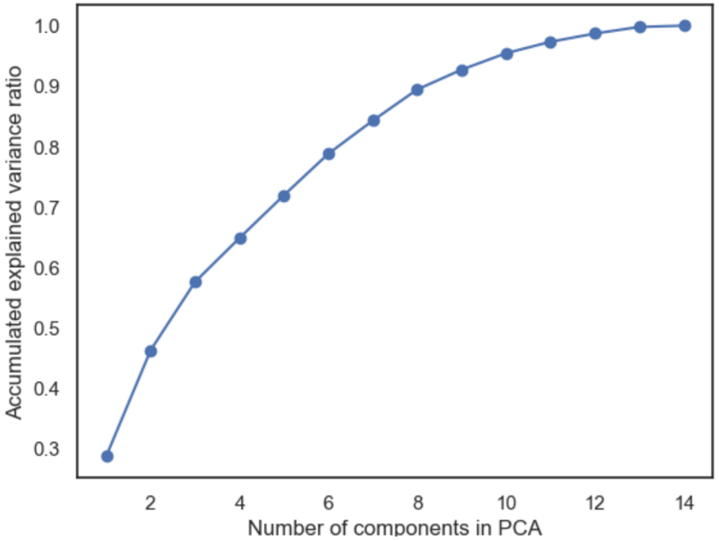


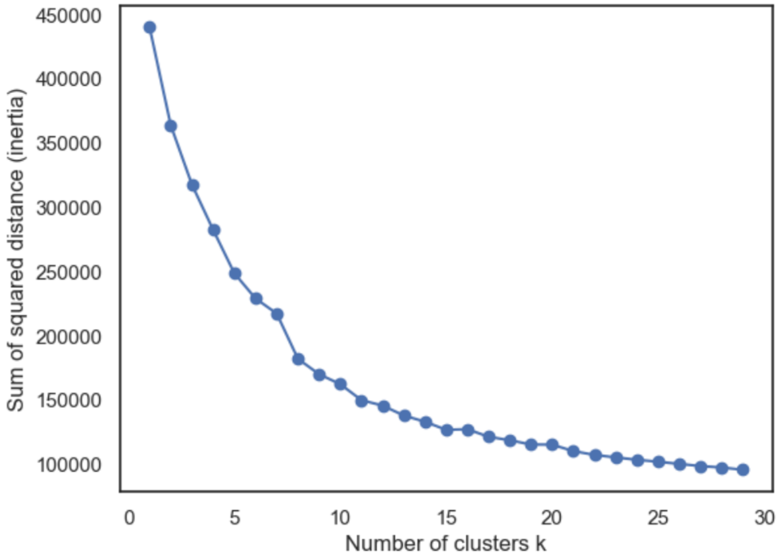
Create user profiles and scale features

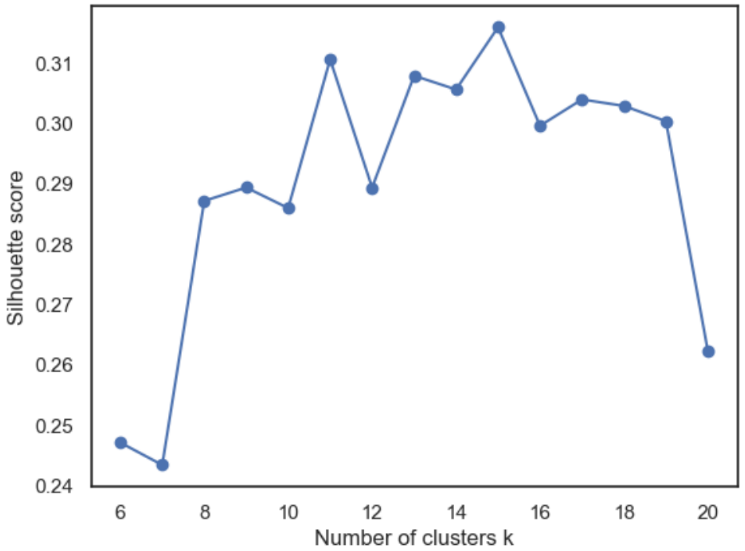
Apply PCA on user profile feature vectors

Perform K-means clustering algorithm

Generate course recommendations







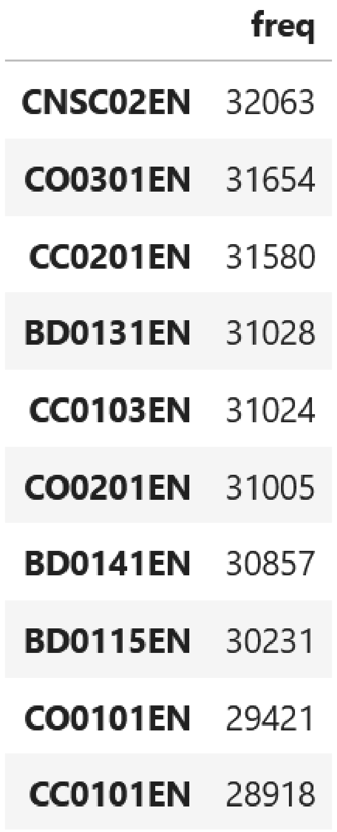
Evaluation results of clustering-based recommender system

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

- used **9** principal components in PCA, and **15** clusters in the KMeans algorithm
- the threshold for course popularity was chose to be at least **1000** enrollements

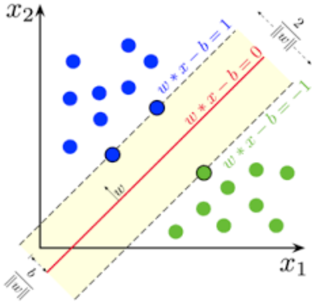
On average, **19.7** new/unseen courses have been recommended per user.

The table shows the top-10 most frequently recommended courses across all users.



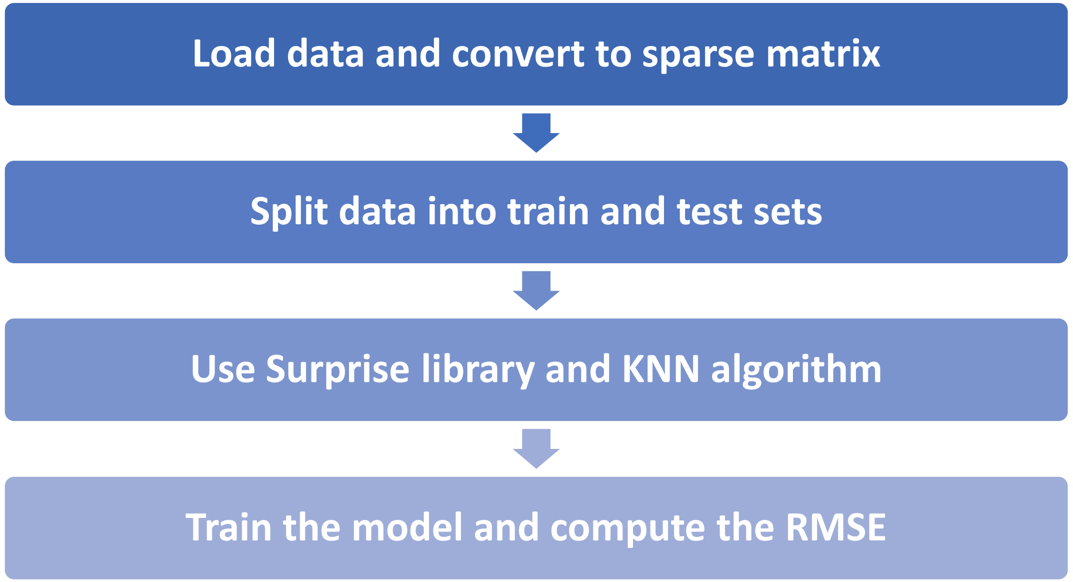
	freq
CNSC02EN	32063
CO0301EN	31654
CC0201EN	31580
BD0131EN	31028
CC0103EN	31024
CO0201EN	31005
BD0141EN	30857
BD0115EN	30231
CO0101EN	29421
CC0101EN	28918

Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system

- Load data and convert to sparse dataframe using pivot().
- Split the data into train and test sets
- Use Surprise library and K-nearest neighbor algorithm KNNBasic().
- Train the model and compute the RMSE (=1.29) on the predictions.



Load data and convert to sparse matrix

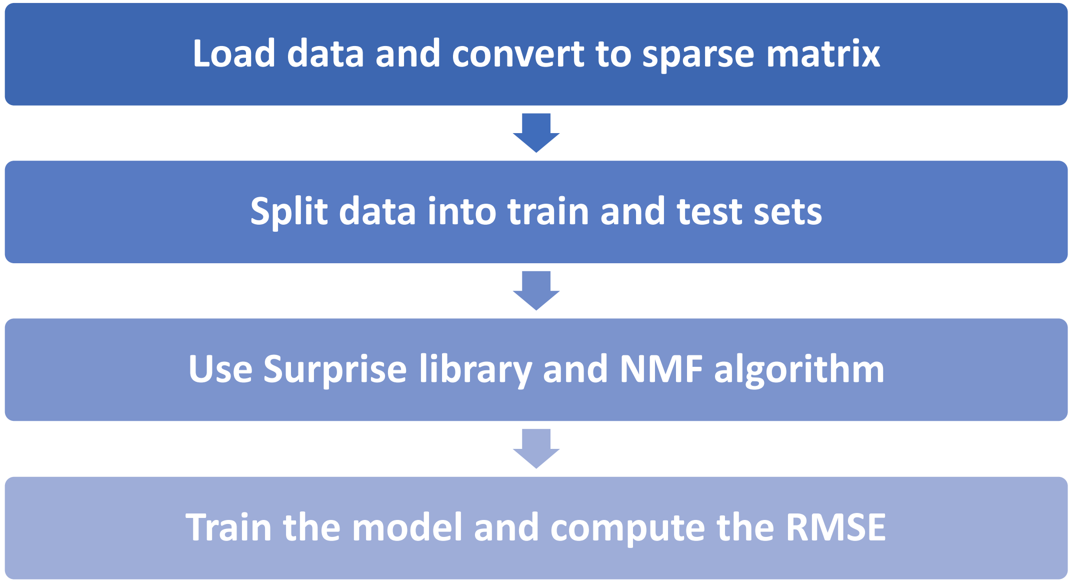
Split data into train and test sets

Use Surprise library and KNN algorithm

Train the model and compute the RMSE

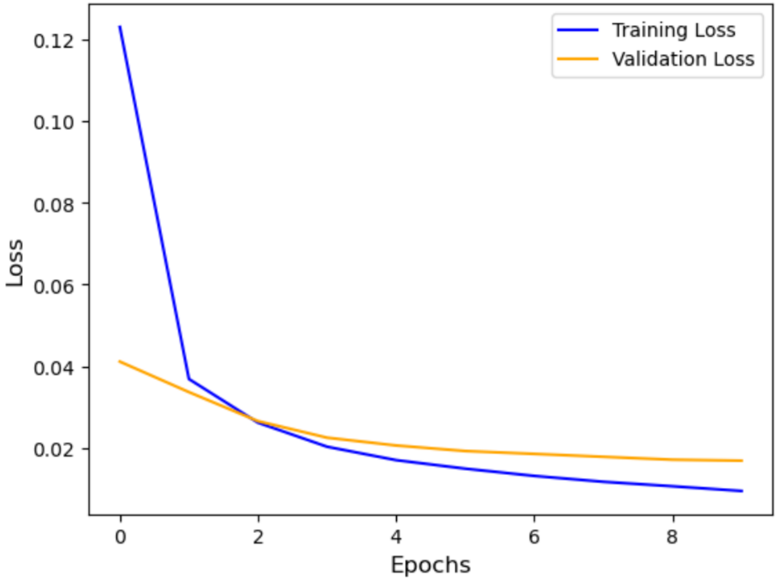
Flowchart of NMF based recommender system

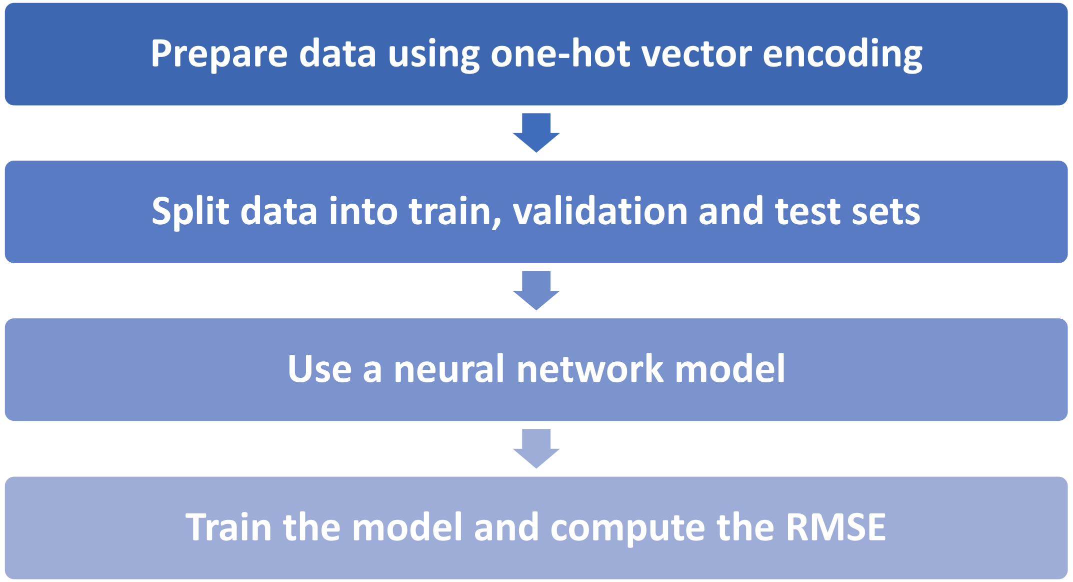
- Load data and convert to sparse dataframe using pivot().
- Split the data into train and test sets
- Use Surprise library and the non-negative matrix factorization algorithm NMF().
- Train the model and compute the RMSE (=1.29) on the predictions.


Load data and convert to sparse matrix
Split data into train and test sets
Use Surprise library and NMF algorithm
Train the model and compute the RMSE

Flowchart of Neural Network Embedding based recommender system

- Prepare the data using one-hot vector encoding.
- Split data into train, validation, and test sets.
- Use the provided neural networkd function RecommenderNet(), which is based on tensorflow and keras.
- Train the model using 16 embedding vector size, the Adam optimizer, and MeanSquareError loss.





Prepare data using one-hot vector encoding

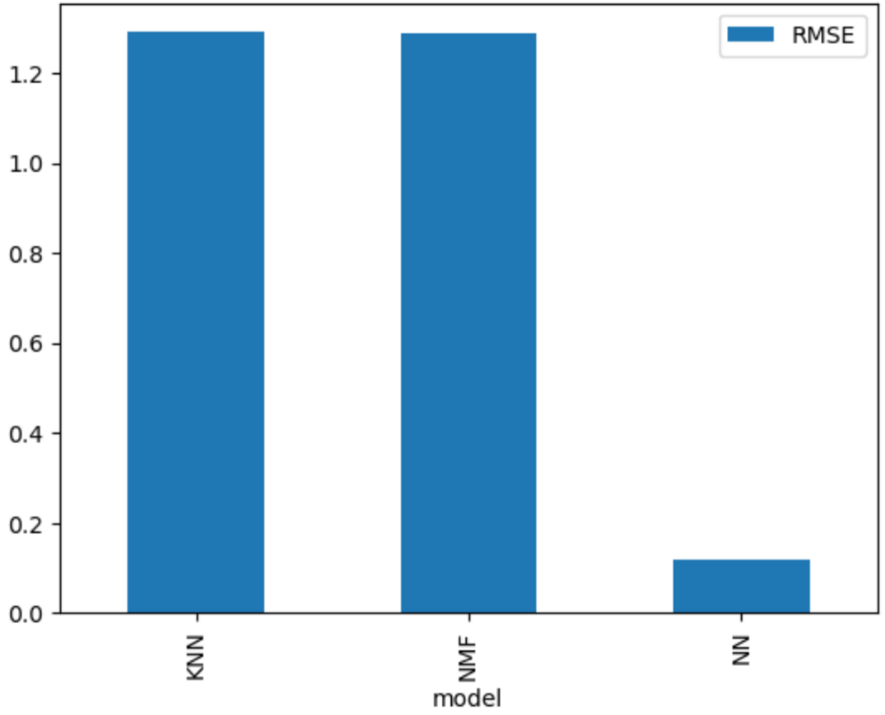
Split data into train, validation and test sets

Use a neural network model

Train the model and compute the RMSE

Compare the performance of collaborative-filtering models

- Barchart visualizing the RMSE of the three supervised-learning models considered.
- The neural networks model has the lowest error of 0.12



Conclusions

- Performed exploratory data analysis to find the most popular courses and their user enrollment distribution
- Developed content-based recommender systems using unsupervised learning (user profile-based, content-based, and clustering-based).
    - Clustering using the PCA (9 components) and k-means (15 clusters) algorithms produced best results.
- Developed collaborative-filtering recommender systems using supervised learning (KNN, NMF, and neural networks)
    - Neural networks produced the lowest RMSE.