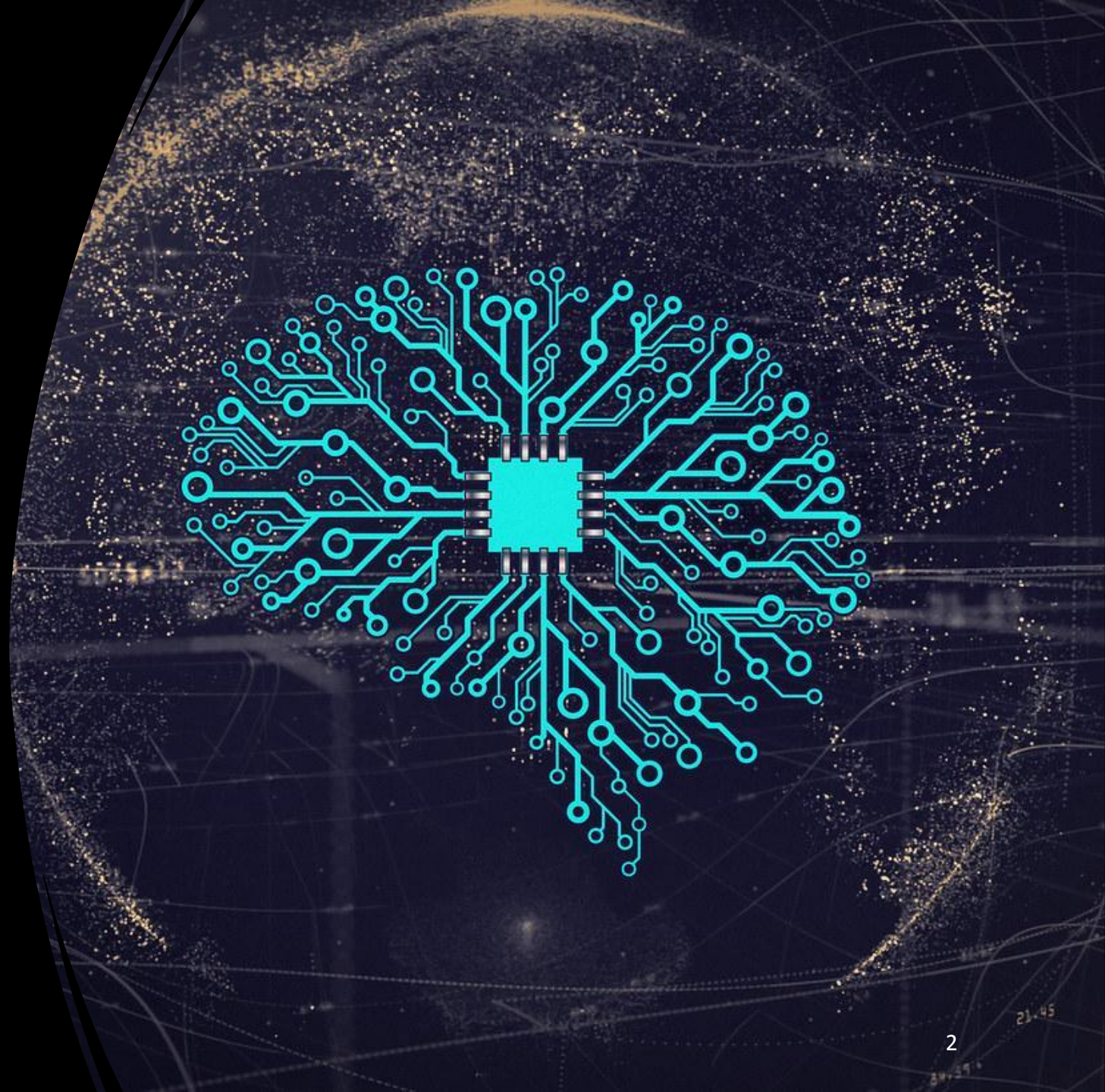
Machine Learning Capstone

Build a Personalized Online Course Recommender System with Machine Learning

María Belén Camandone 07-2024

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

In the contemporary digital era, online learning has gained immense popularity due to its flexibility, accessibility, and the wide range of courses available. However, the vast amount of content can overwhelm learners, making it challenging for them to identify courses that align with their interests and goals. This necessitates the development of personalized online course recommender systems, which can suggest relevant courses to users based on their preferences and past behavior.

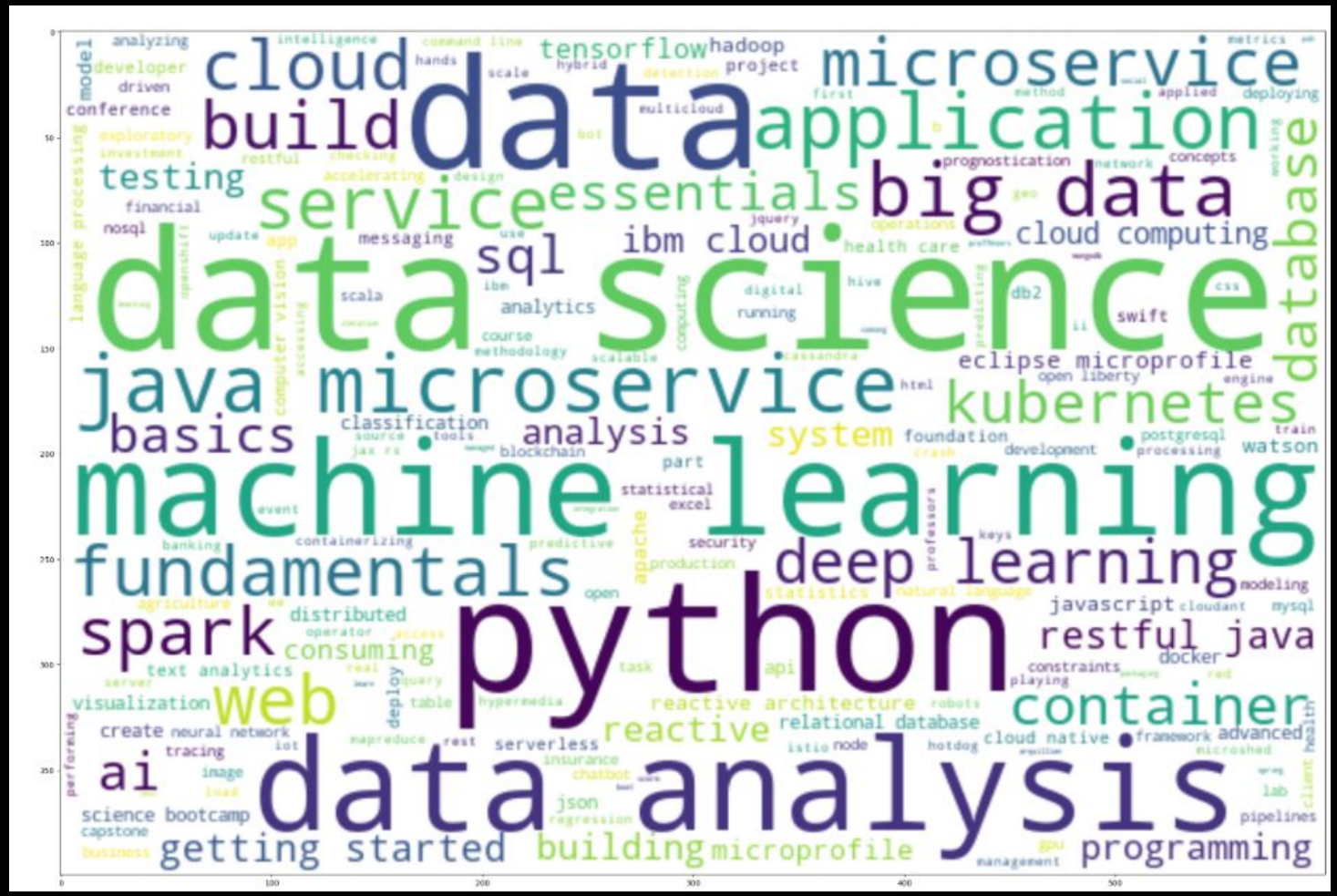
This project focuses on building a personalized online course recommender system using various machine learning algorithms. The project is structured into several modules, each employing a different recommendation method.

Exploratory Data Analysis

Word cloud of course titles

A word cloud is a graphical representation of text data where the size of each word indicates its frequency or importance within the dataset.

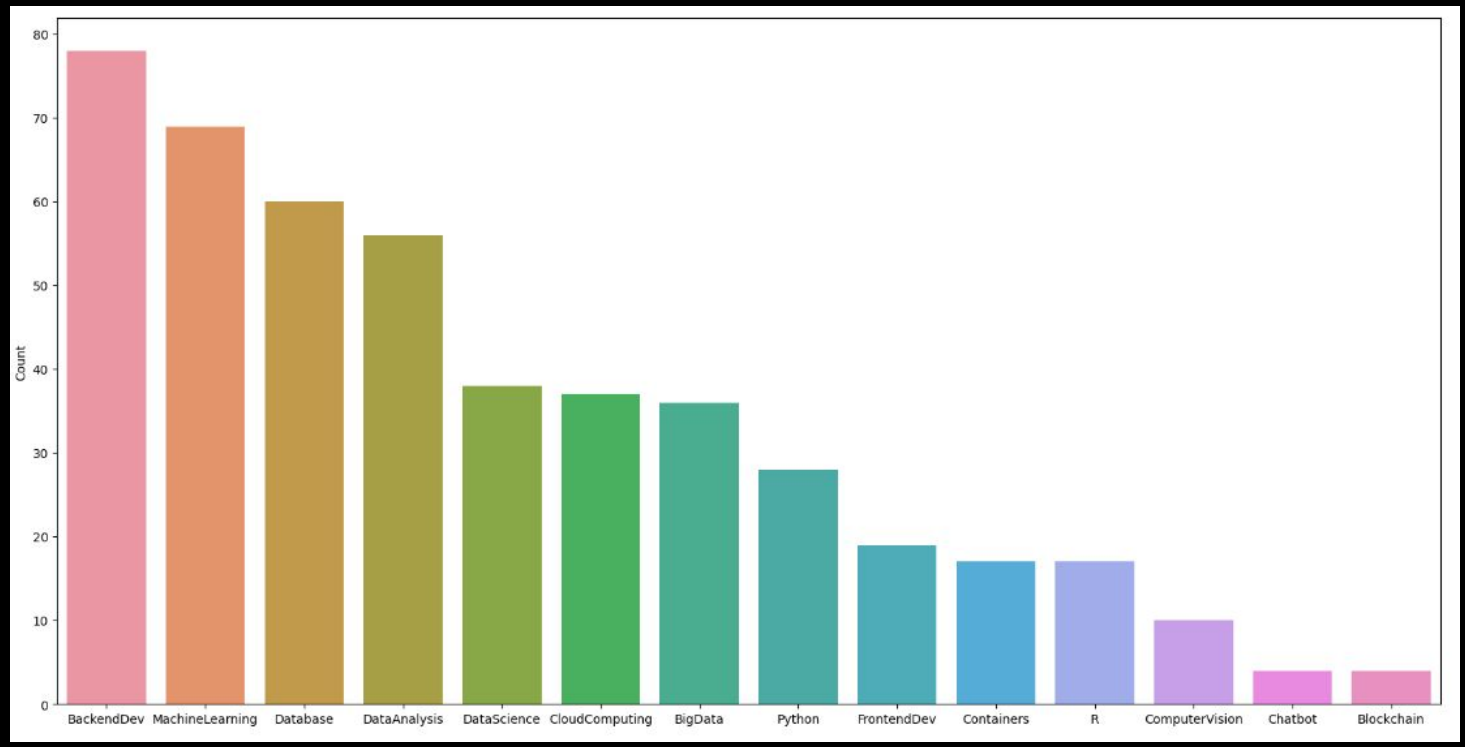
For this project, the word cloud was generated using the titles of all available courses. The most prominent words, such as "data," "science," "machine," and "learning," suggest that courses related to data science and machine learning are highly prevalent. By looking at these keywords, we should have a general understanding that the courses in the dataset are focused on demanding IT skills.



Course counts per genre

The bar chart illustrates the distribution of course offerings across various genres. The x-axis represents different course genres, while the y-axis indicates the number of courses available in each genre.

The distribution of courses by genre reveals a strong emphasis on backend development, machine learning, and data-related skills. This aligns with current industry trends where these skills are in high demand. Understanding this distribution helps in tailoring the recommendation system to prioritize popular and essential genres, thereby enhancing the relevance and value of course recommendations for users.



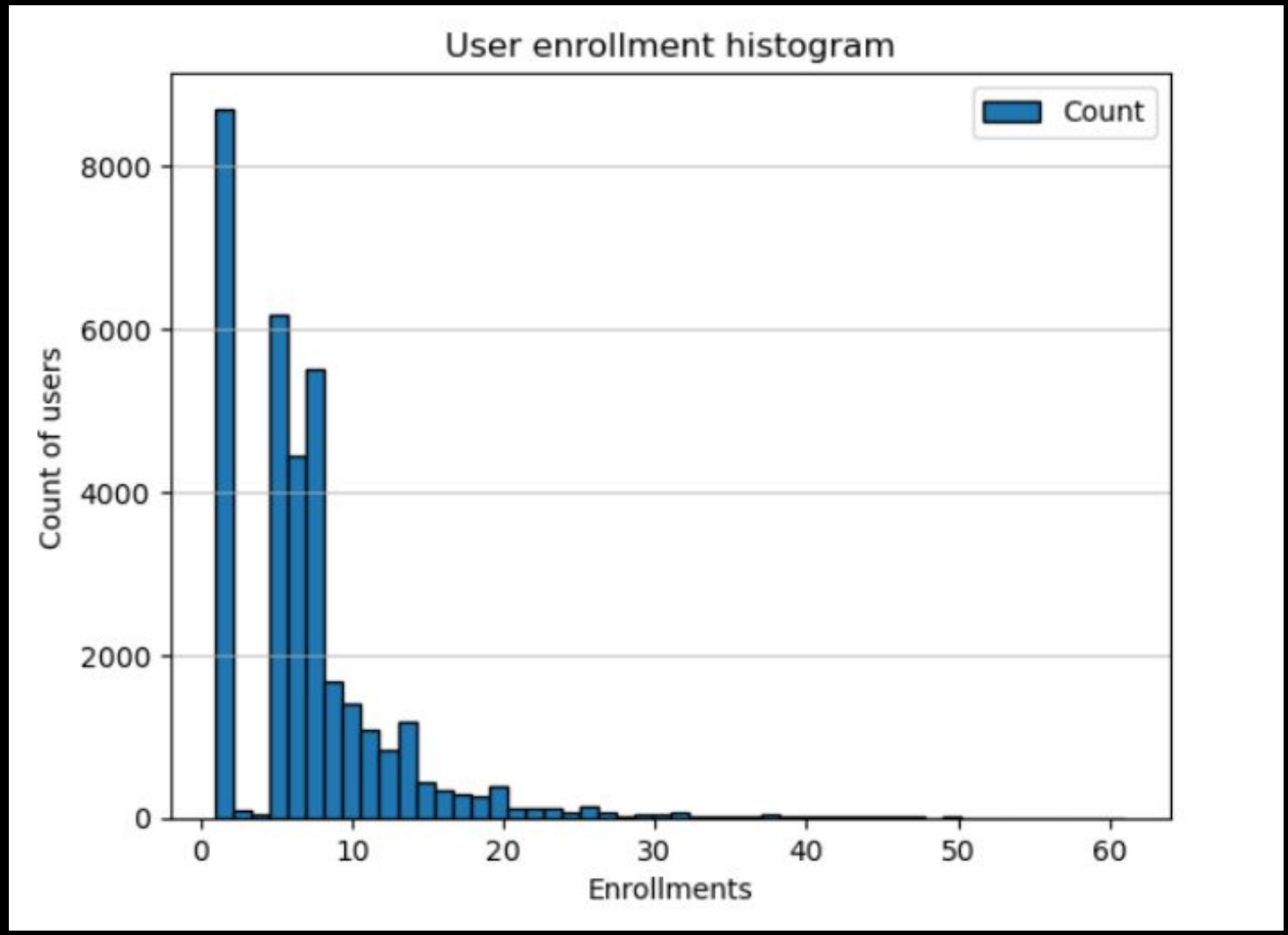
Course enrollment distribution

This histogram illustrates the distribution of user enrollments across various courses. The x-axis represents the number of enrollments per user, while the y-axis shows the count of users with a given number of enrollments.

The median is 6 courses, and the largest number of users enroll in a single course.

The distribution shows a long tail, indicating that while the majority of users enroll in a small number of courses, there are outliers who enroll in a significantly higher number of courses.

Understanding this distribution helps in personalizing recommendations to cater to the diverse engagement levels of users.

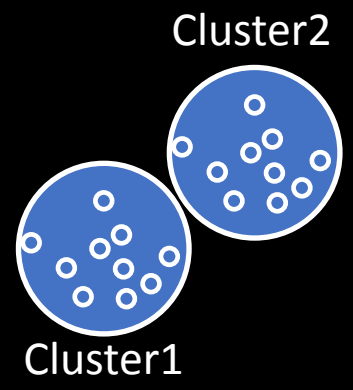


20 most popular courses

The distribution of enrollments among these top 20 courses underscores the prominence of data science and related fields such as machine learning, big data, and programming. Courses focusing on practical skills and cutting-edge technologies tend to attract more enrollments, reflecting the current trends and demands in the industry. This information is vital for developing a recommendation system that aligns with user interests and industry needs.

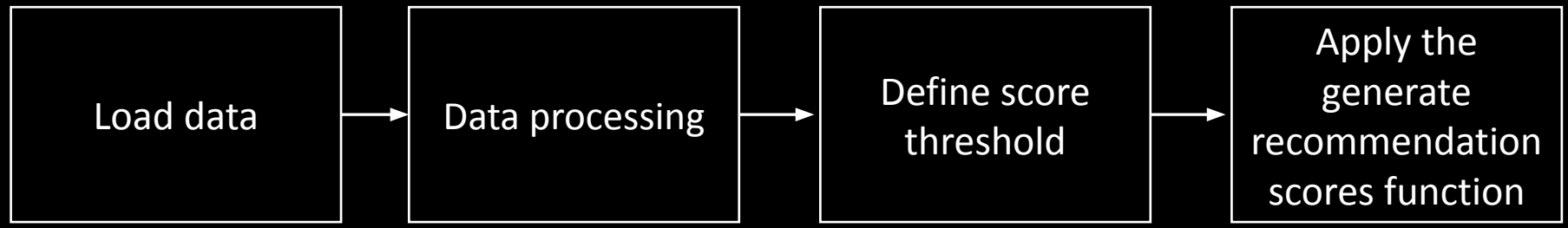
	TITLE	count
0	python for data science	14936
1	introduction to data science	14477
2	big data 101	13291
3	hadoop 101	10599
4	data analysis with python	8303
5	data science methodology	7719
6	machine learning with python	7644
7	spark fundamentals i	7551
8	data science hands on with open source tools	7199
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

Content-based Recommender System using Unsupervised Learning



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

The most common type of content-based recommendation system suggests items to users based on their individual profiles. A user's profile encapsulates their preferences and tastes, which are inferred from their interactions, such as the frequency of clicks or the items they have liked.
The recommendation process hinges on the similarity between items. This similarity is determined by comparing the content features of the items, such as category, tags, genre, and other attributes. In essence, the system analyzes these features to find items that closely match the user's profile.



Evaluation results of user profile-based recommender system

I chose a threshold score of 60, and I only keep the results with scores larger than the recommendation threshold.

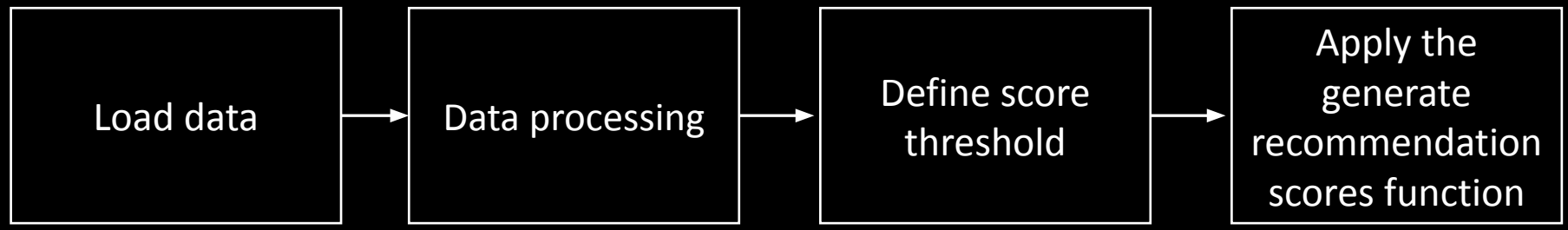
On average, 12 new/unseen courses have been recommended per user (in the test user dataset).

The top-10 commonly recommended courses are:

	TITLE	count
0	analyzing big data with sql	1044
1	foundations for big data analysis with sql	1044
2	getting started with the data apache spark ma...	926
3	cloud computing applications part 2 big data...	672
4	analyzing big data in r using apache spark	559
5	spark overview for scala analytics	558
6	\r\ndistributed computing with spark sql	546
7	using the cql shell to execute keyspace operat...	546
8	big data essentials hdfs mapreduce and spark...	546
9	big data capstone project	546

Flowchart of content-based recommender system using course similarity

In these recommendations we use the course similarity matrix to identify and recommend new courses that are similar to the ones in which users are already enrolled.



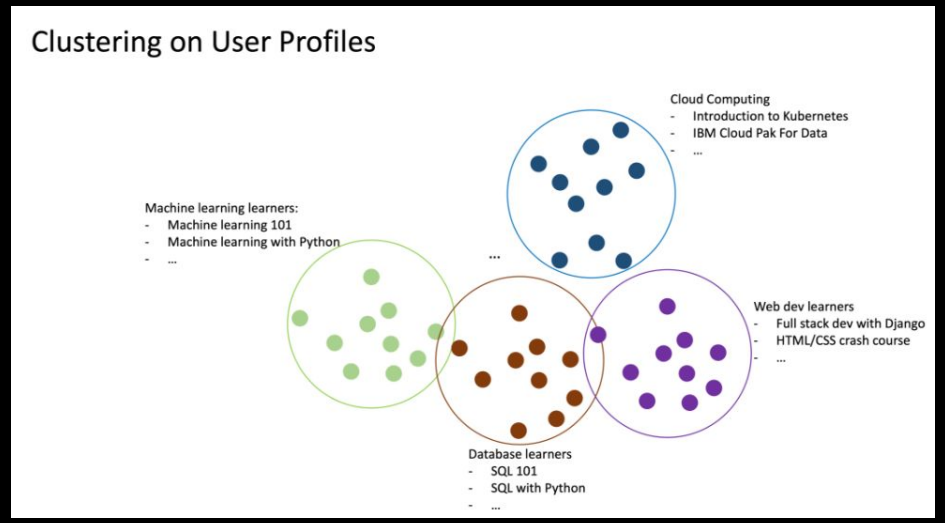
Evaluation results of course similarity based recommender system

I chose a threshold score of 55, and I only keep the results with scores larger than the recommendation threshold.

- On average, 2 new/unseen courses have been recommended per user (in the test user dataset).
- The top-10 commonly recommended courses are:

TITLE	count
introduction to data science in python	7427
introduction to data science in python	7427
introduction to data analytics	2511
the r programming environment	2208
data analysis with r programming	2208
data analysis with r programming	2208
a crash course in data science	1869
data science fundamentals for data analysts	1479
data analysis using python	1441
\nsql for data science	1441

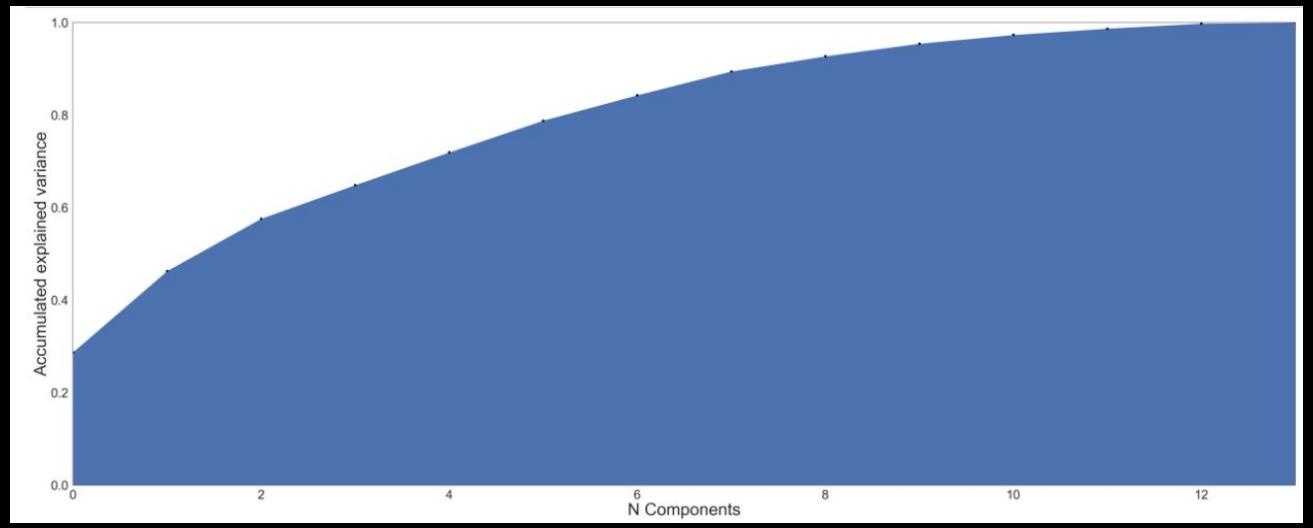
Flowchart of clustering-based recommender system

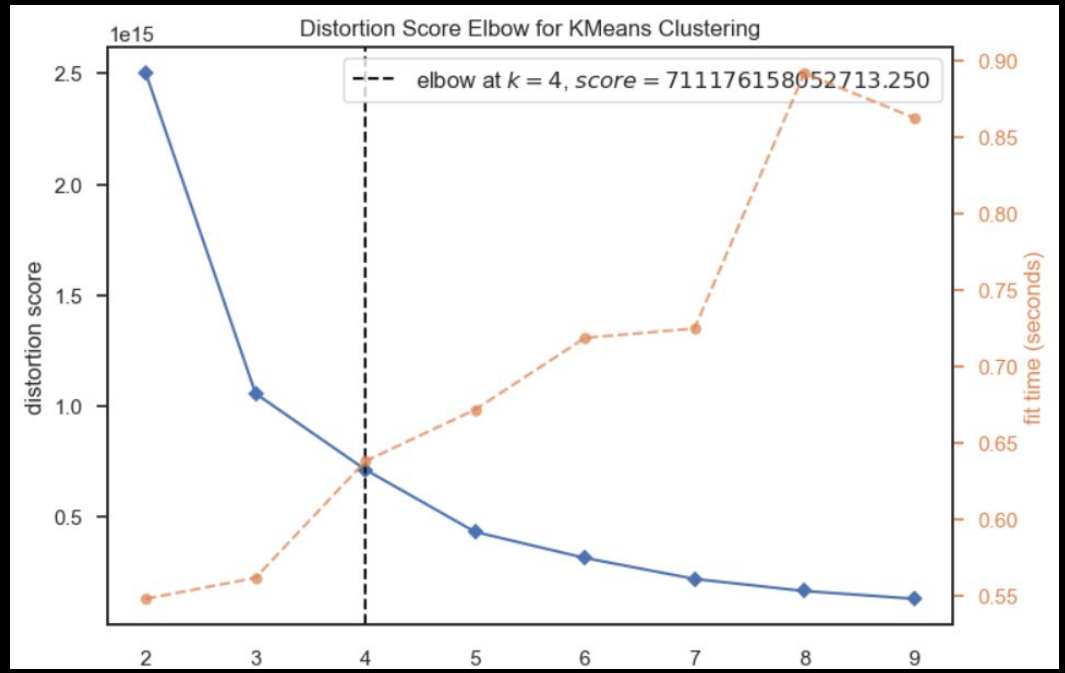


We can apply clustering algorithms such as K-means or DBSCAN to group users based on their similar learning interests. Then, generate course recommendations based on the popular courses in the same cluster.

Evaluation results of clustering-based recommender system

Keep 90% of explained variance, apply Elbow Method, grouping in 4 clusters:





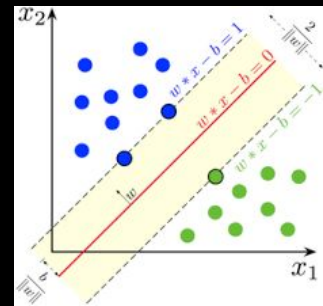
Evaluation results of clustering-based recommender system

- On average, 15 new/unseen courses have been recommended per user (in the test user dataset)
- The top-10 commonly recommended courses are:

Top-10 commonly recommended courses:

	item	recommendation_count	TITLE
0	ST0101EN	28886	statistics 101
4	RP0101EN	28664	r for data science
8	CB0103EN	28389	build your own chatbot
12	ML0115EN	27578	deep learning 101
16	DS0105EN	26702	data science hands on with open source tools
20	BD0211EN	26350	spark fundamentals i
24	DS0103EN	26182	data science methodology
28	DA0101EN	25598	data analysis with python
32	BD0111EN	23302	hadoop 101
36	CO0101EN	21864	docker essentials a developer introduction

Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system

Collaborative filtering is a technique that leverages user interactions and preferences to make recommendations. It is based on the idea that if two users have had similar preferences in the past, they are likely to continue to have similar tastes in the future. This approach can be divided into two main categories:

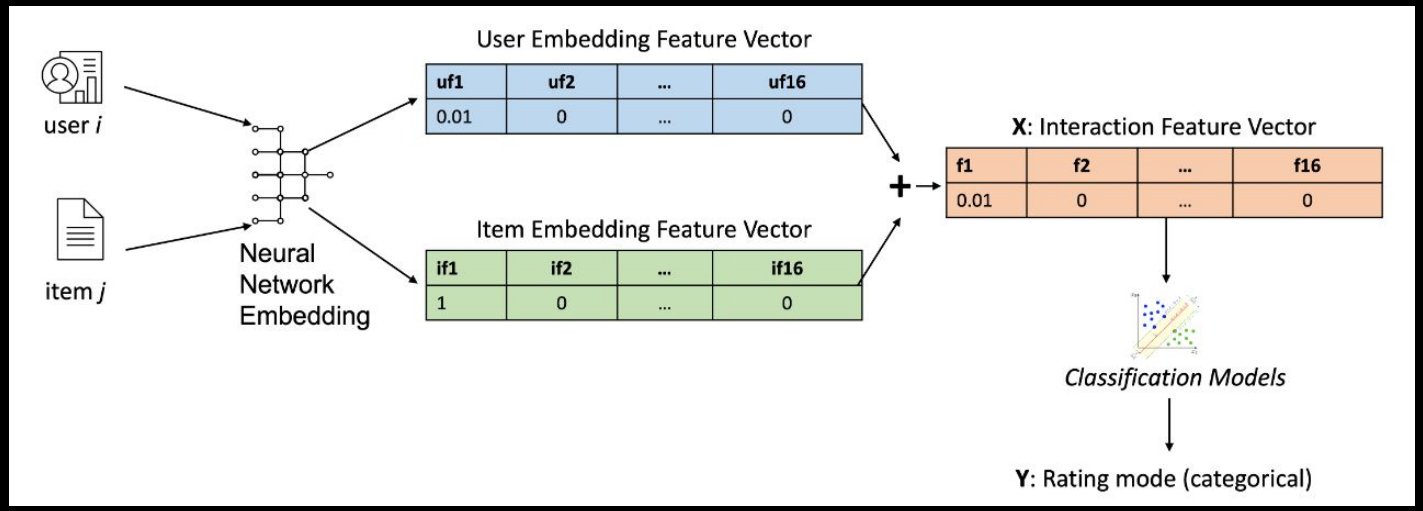
- User-based: collaborative filtering is based on the user similarity or neighborhood
- Item-based: collaborative filtering is based on similarity among items

In this project, I used the Surprise library and the KNN algorithm with item-based collaborative filtering. To calculate the similarities, I chose the cosine and Pearson correlation metrics.

Flowchart of NMF based recommender system

Non-negative matrix factorization (NMF) decomposes a big sparse matrix into two smaller and dense matrices.
Non-negative matrix factorization can be one solution to big matrix issues. The main idea is to decompose the big and sparse user-interaction into two smaller dense matrices, one represents the transformed user features and another represents the transformed item features.

Flowchart of Neural Network Embedding based recommender system

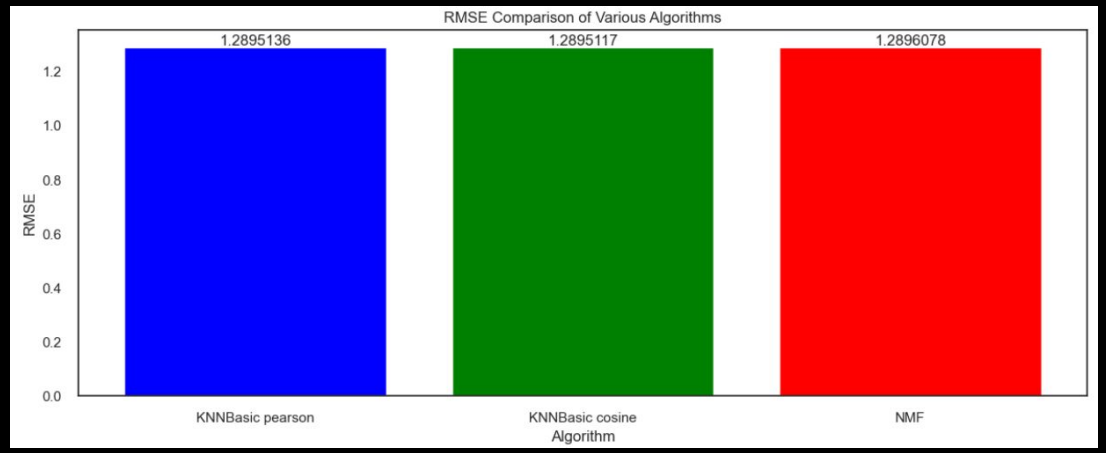


Built models to predict numerical course ratings using the embedding feature vectors extracted from neural networks.

Compare the performance of collaborative-filtering models (KNN,NMF)

Based on the results presented in the bar graph, we can draw the following conclusions about the performance of the three algorithms (KNNBasic with Pearson similarity, KNNBasic with Cosine similarity, and NMF:

- **Algorithm Selection:** Since all three algorithms produced similar values of RMSE, the choice among them may depend on other factors such as computational efficiency and interpretability.
- **Parameter Tuning:** The similar RMSE values suggest that further tuning of the parameters for each algorithm might be necessary to potentially uncover differences in performance.
- **Dataset Characteristics:** The specific characteristics of the dataset might have influenced the similar performance outcomes. It might be beneficial to analyze if certain inherent properties of the dataset (e.g., sparsity, distribution of ratings) contribute to this result.



Compare the performance of collaborative-filtering models using the embedding feature vectors

For this case, using the embedding feature vectors extracted from neural networks to predict numerical course ratings with a regression linear model it is not a viable option.
Linear regression is designed for continuous target variables, where the outcome can take any value within a range. Our target variable in this dataset is discrete, with only a few possible outcomes (3, 4, or 5), so, linear regression is not appropriate because can produce predictions that are not valid ratings. For example, it might predict a rating of 3.7 or 4.2.

We also consider the prediction problem as a classification problem also using embedding features.
We tested with Logistic Regression and Xgboost models but they both had a low overall accuracy around (33%) , indicating that they are not performing well in classifying the instances into the three classes.

Conclusions

In this project, we developed a personalized online course recommendation system using various machine learning techniques. The main objectives were to enhance user experience by providing course recommendations that align with their interests and past behaviors.

Specifically, we developed:

Content-based recommendation systems:

1. Using user profiles and course genres.
2. Based on course similarity.
3. Using user profile clustering with PCA and K-means.

Collaborative filtering systems:

4. Based on KNN (K-Nearest Neighbors).
5. Using NMF (Non-negative Matrix Factorization).
6. Using the embedding feature vectors

Conclusions

Limitations:

Limited Hyperparameter Optimization: Due to hardware constraints, we were unable to perform extensive hyperparameter optimization. This limitation may have affected the performance and accuracy of some models.

Results:

The content-based models are easy to interpret, but, can have a high computational cost if the dataset is large. However, applying PCA and K-means proved to be an efficient and quick strategy for clustering user profiles.

The collaborative filtering algorithms, implemented through the Surprise libraries, were simple and fast to apply. These methods provided efficient recommendations.

On the other hand, using the embedding feature vectors extracted from neural networks and then apply linear regression was not possible in this case, a target variable consisting of discrete ratings (3, 4, or 5) is not appropriate due to the continuous nature of linear regression predictions. Furthermore, the classification models not performing well in classifying the instances into the three classes.

Conclusions

Future Research Directions:

Exploring other hyperparameters: Conducting more comprehensive hyperparameter optimization using grid search, random search, or Bayesian optimization methods to enhance model performance.

Impact:

The implementation of these recommendation systems has great potential to improve user experience on online learning platforms, making it easier for users to find courses that best fit their needs and interests. This not only increases user satisfaction but can also enhance student retention and engagement on educational platforms.

Appendix

Complete code:

Final project : Recommender System - Part 1: https://bit.ly/3WpD2ob

Final project : Recommender System - Part 2: https://bit.ly/3SqHyBy