Build a Personalized Online Course Recommender System with Machine Learning

Ignatius Evans Erlangga
22nd June, 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

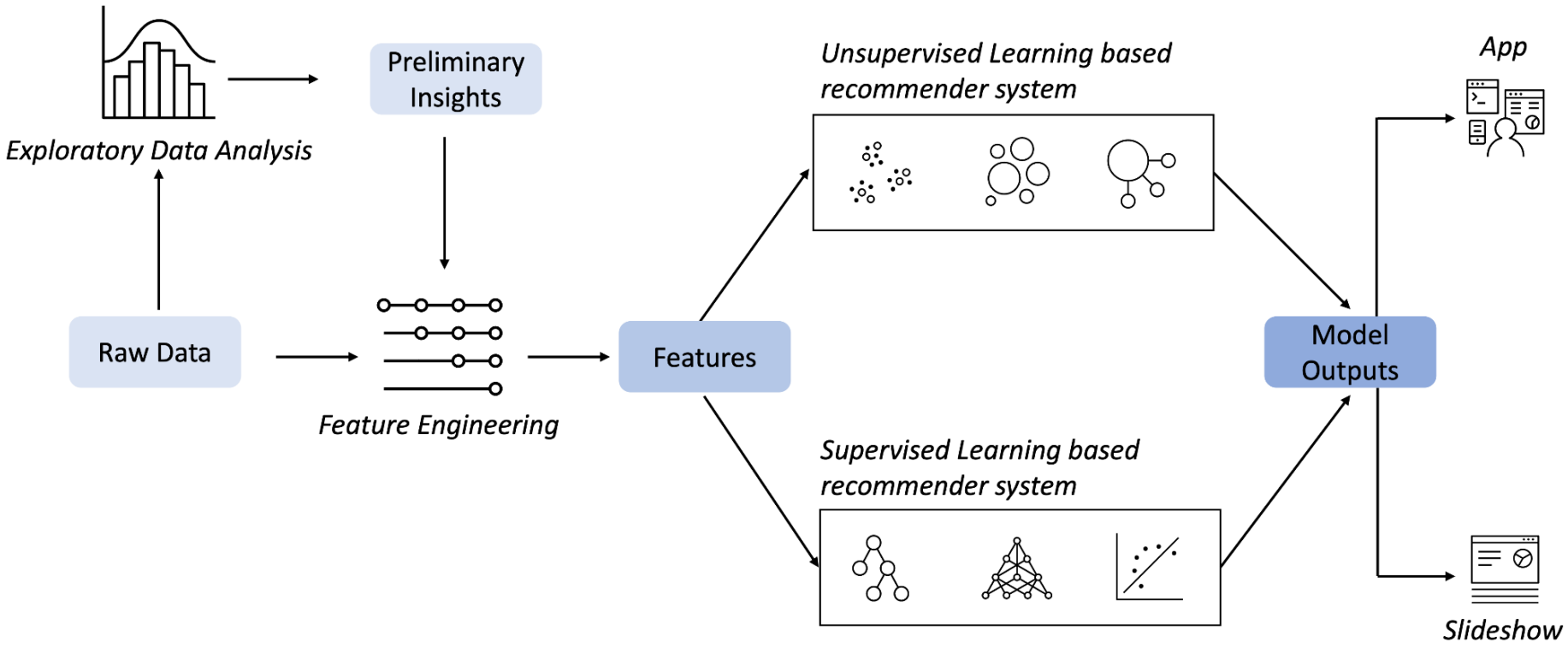
- The primary objective of this project is to enhance the learners' educational experience by assisting them in swiftly identifying new courses of interest and more effectively structuring their learning paths. Concurrently, by facilitating greater interaction between learners and courses through your recommender systems, your company's revenue is likely to increase.
- Currently, this project is in the Proof of Concept (PoC) phase, with the primary focus being the exploration and comparison of various machine learning and deep learning models to identify the one that performs best in off-line evaluations.

Problem Statement and Hypotheses :
By identifying new courses of interest and more effectively structuring Learner's learning path, will it be able to enhace their educational experience?
H0 : It will enhance their educational experience.
H1 : It will not enhance their educational experience.

Capstone Overview - Machine Learning pipeline

- Collecting and understanding data
- Performing exploratory data analysis on online course enrollments datasets
- Extracting Bag of Words (BoW) features from course textual content
- Calculating course similarity using BoW features
- Building content-based recommender systems using various unsupervised learning algorithms, such as:
  Distance/Similarity measurements, K-means, Principal Component Analysis (PCA), etc.
- Building collaborative-filtering recommender systems using various supervised learning algorithms
  K Nearest Neighbors, Non-negative Matrix Factorization (NMF), Neural Networks, Linear Regression, Logistic Regression, RandomForest, etc.
- Deploying and demonstrate modeling via a web app built with streamlit. Streamlit is an open-source app framework for Machine Learning and Data Science to quickly demonstration.
- Reporting in paper.

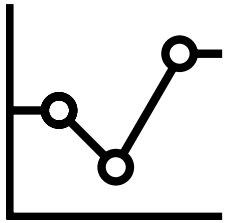
Machine Learning Pipeline



Exploratory Data Analysis

Pipeline:

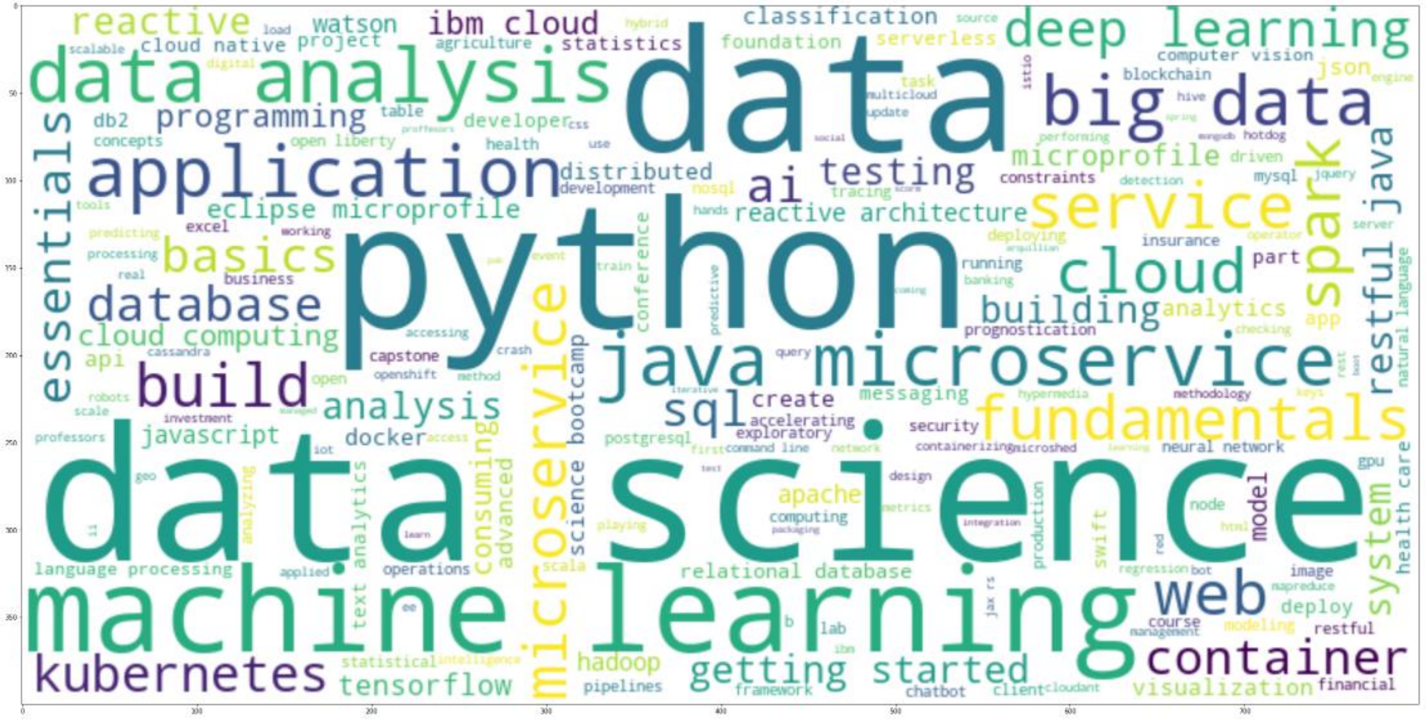
1. Describe the statistic of data columns
2. Identify keywords in course titles using a WordCloud
3. Determine popular course genres
4. Calculate the summary statistics and create visualizations of the online course enrollment dataset



EDA – Word Cloud on Course Titles

The omnipresences following by size respectively are:

1. data ->
2. data science ->
3. python ->
4. machine learning ->
5. data analysis ->
6. application ->
7. big data ->
8. java ->
9. microservice
10. web



Identify keywords in course titles using a Word Cloud

Exploratory Data Analysis – Course Count per Genre

	COURSE_ID	TITLE	Database	Python	CloudComputing	DataAnalysis	Containers	MachineLearning	ComputerVision	DataScience	BigData	Chatbot	R	BackendDev	FrontendDev	Blockchain
0	ML0201EN	robots are coming build iot apps with watson ...	0	0	0	0	0	0	0	0	0	0	0	1	1	0
1	ML0122EN	accelerating deep learning with gpu	0	1	0	0	0	1	0	1	0	0	0	0	0	0
2	GPXX0ZG0EN	consuming restful services using the reactive ...	0	0	0	0	0	0	0	0	0	0	0	1	1	0
3	RP0105EN	analyzing big data in r using apache spark	1	0	0	1	0	0	0	0	1	0	1	0	0	0
4	GPXX0Z2PEN	containerizing packaging and running a sprin...	0	0	0	0	1	0	0	0	0	0	0	1	0	0

```
COURSE_ID          object
TITLE              object
Database            int64
Python              int64
CloudComputing      int64
DataAnalysis        int64
Containers          int64
MachineLearning     int64
ComputerVision      int64
DataScience         int64
BigData             int64
Chatbot             int64
R                   int64
BackendDev          int64
FrontendDev         int64
Blockchain          int64
dtype: object
```

Describe the statistic of data columns

There are 307 courses at total.

Each course is a vector of genres 1x16.

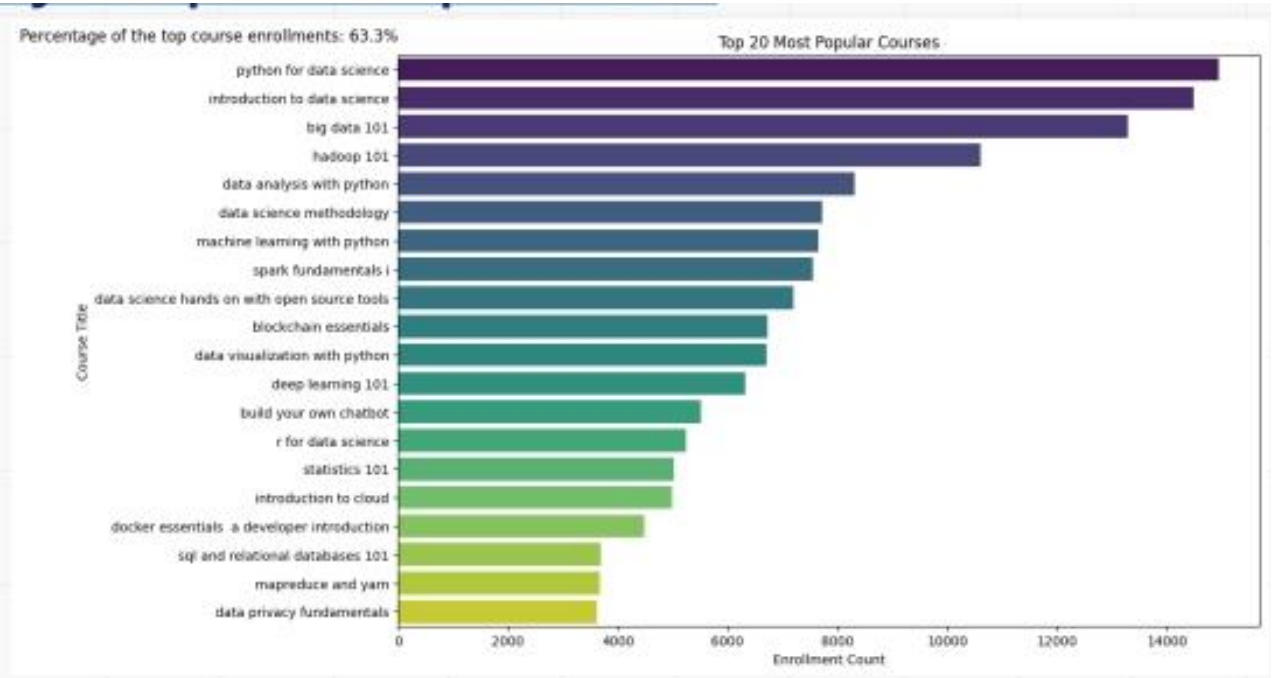
0 means this course is not related to this genre.

1 means this course is related to this genre.

EDA: Identifying the Top 20 Most Popular Courses

Top 20 Most Popular Courses:

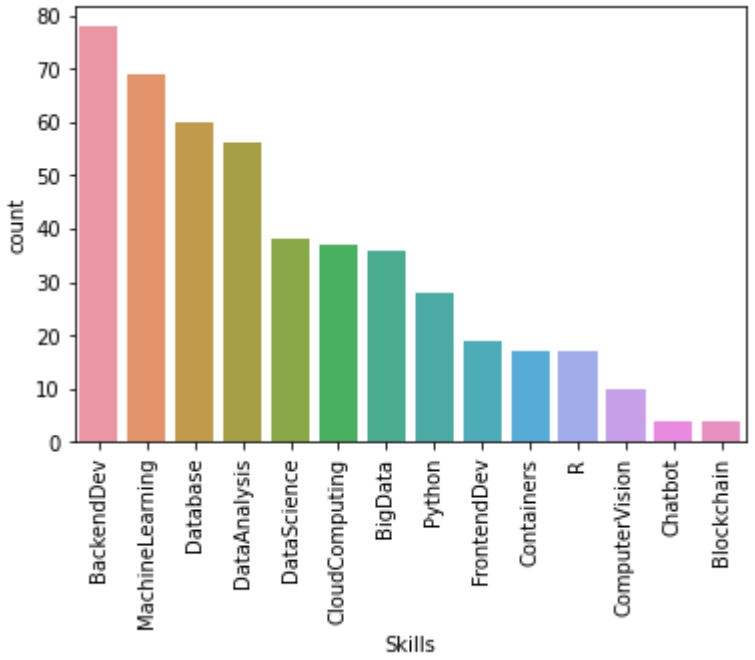
	COURSE_ID	Ratings	TITLE
0	PY0101EN	14936	python for data science
1	DS0101EN	14477	introduction to data science
2	BD0101EN	13291	big data 101
3	BD0111EN	10599	hadoop 101
4	DA0101EN	8303	data analysis with python
5	DS0103EN	7719	data science methodology
6	ML0101ENv3	7644	machine learning with python
7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



In this task, the objective was to determine the top 20 most popular courses based on enrollment counts. Using the enrollment data, the courses were aggregated, sorted, and the top 20 courses were identified. The resulting list showcases the titles and enrollment counts of these highly sought-after courses, providing insights into the preferences of learners in the online learning platform

EDA – Course Enrolment Distribution

	Skills	count
11	BackendDev	78
5	MachineLearning	69
0	Database	60
3	DataAnalysis	56
7	DataScience	38
2	CloudComputing	37
8	BigData	36
1	Python	28
12	FrontendDev	19
4	Containers	17
10	R	17
6	ComputerVision	10
9	Chatbot	4
13	Blockchain	4

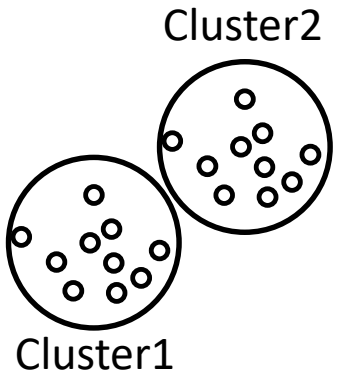


Determine popular course genres

BackendDev, MachineLearning, Database are the utmost popular genres.

While Blockchain, Chatbot, ComputerVision are the most less common ones.

Content-based Recommender System using Unsupervised Learning



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

1. **Raw Data:** This refers to the initial dataset containing information about users, courses, and their interactions or preferences. In our case, the raw data includes
user profiles, course genres, and possibly course ratings or interactions.

2. **Data Processing:**
• This step involves cleaning and preprocessing the raw data to prepare it for analysis.
• In our context, data processing includes tasks such as handling missing values, removing duplicates, and transforming data into a suitable format for further analysis.

3. **Cleaned Dataset:**
• After data processing, we obtain a cleaned dataset that is ready for feature engineering.
• This dataset contains all the relevant information about users and courses, with any inconsistencies or errors addressed.

4. **Feature Engineering:**
• Feature engineering involves creating new features or representations of the data that can be used to train a machine learning model.
• In our case, we create user profile vectors and course genre vectors as features. These vectors capture the interests or preferences of users and the characteristics of courses, respectively.

5. **Features:**
• The final output of feature engineering is a set of features, represented by user profile vectors and course genre vectors.
• These features serve as the input to the content-based recommender system. By comparing user profile vectors with course genre vectors, the system can generate recommendations personalized to each user's interests.

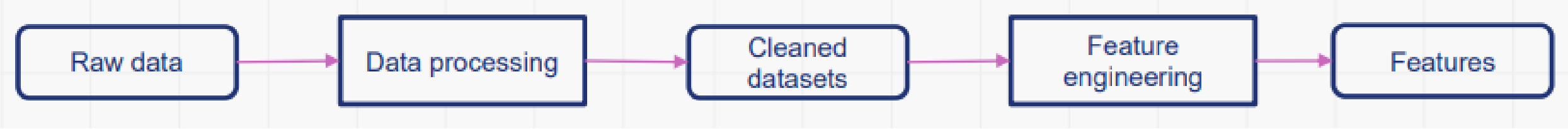
Evaluation results of user profile-based recommender system

Hyper-parameter Settings: In our discussion, we set a recommendation score threshold of 10.0 to filter out low-scoring recommendations. This threshold determines which courses are considered relevant enough to be recommended to users. Additionally, we may have adjusted other hyperparameters such as feature representation methods or similarity metrics during the implementation of the recommender system.

Average Number of New Courses Recommended per User: We calculated the average number of new courses recommended per user in the test user dataset. This metric helps evaluate the coverage and diversity of the recommender system. In our case, the average number was approximately **61.82 courses per user.**

Top-10 Most Frequently Recommended Courses: The table represents the top 10 most frequently recommended courses based on the user profile based recommender system. Each row corresponds to a course, identified by its COURSE_ID, and the number of times that course has been recommended to users, denoted by the RECOMMENDATION_COUNT column. These recommendations are generated by analyzing user profiles and course genre vectors, with courses scoring higher in relevance to a user's interests being recommended more frequently

Flowchart of content-based recommender system using course similarity



Raw Data: Initially, this phase involves the dataset comprising various course details such as titles, descriptions, and other pertinent attributes.

Data Processing: This stage focuses on preprocessing the raw data, including tokenization and lemmatization. Tokenization breaks down text into individual words, while lemmatization converts these words into their base forms.

Cleaned Dataset: Post-processing, the dataset undergoes cleaning procedures. This includes the removal of stopwords (commonly used words with minimal semantic value) and outliers (data points deemed irrelevant or noisy).

Feature Engineering: In this crucial step, the cleaned dataset is transformed into numerical features that effectively represent each course. Specifically, TF-IDF (Term Frequency-Inverse Document Frequency) vectors are computed for each course, reflecting the importance of words within each course description relative to the entire dataset.

Features: This term denotes the final set of features utilized to portray each course. These features consist of the TF-IDF vectors derived from the feature engineering process. They are subsequently employed to calculate similarities between courses and generate recommendations based on content similarity.

This breakdown illustrates the sequential stages involved in transforming raw data into a structured format suitable for building and implementing a course recommendation system based on machine learning models.

Evaluation results of course similarity based recommender system

Hyper-parameter Settings:
The hyper-parameter setting for generating recommendations in the course similarity based recommender system was a similarity threshold of 0.6. This threshold determined the level of similarity required between courses for them to be recommended to users.

Average Number of New Courses Recommended per User: The average number of new or unseen courses recommended per user in the test user dataset was approximately **0.987.** This metric provides insight into the diversity of recommendations provided to users and helps assess the system's effectiveness in suggesting novel content.

Top-10 Most Frequently Recommended Courses: Among all users, the top 10 most frequently recommended courses were identified. Notably, "excourse22" and "excourse62" were recommended the most, each appearing 257 times, followed by "WA0103EN" with 101 recommendations. Other courses like "TA0105" and "DS0110EN" were also frequently suggested, indicating their relevance and popularity within the user base. These evaluation results offer valuable insights into the performance and effectiveness of the course similarity based recommender system, informing potential improvements and optimizations for future iterations

Flowchart of clustering-based recommender system

1. Raw Data: This denotes the initial user profile feature vectors containing information about users' preferences across various course genres. Each vector encompasses features related to genres such as Machine Learning, Data Science, and Cloud Computing.

2. Data Processing (Normalization): At this stage, the raw data undergoes preprocessing to address missing values, outliers, and other data quality issues. Normalization techniques like StandardScaler are applied to ensure uniform scale and distribution across all features, crucial for optimizing the performance of clustering algorithms.

3. Cleaned Dataset (Standardization): Following data processing, a cleaned dataset is obtained where user profile features are standardized using methods like StandardScaler. Standardization ensures each feature has a mean of 0 and a standard deviation of 1, meeting the requirements of many machine learning algorithms.

4. Feature Engineering (PCA): This step involves transforming the original user profile features into a new feature set that retains essential information while reducing dimensionality. Principal Component Analysis (PCA) is utilized for this purpose, identifying eigenvectors that explain maximum variance in the data and projecting original features onto these components.

5. Features (PCA Transformed): The final outcome of this process is the transformed feature set obtained post-PCA application. These features represent a reduced dimensionality representation capturing the most significant aspects of the original user profile data.

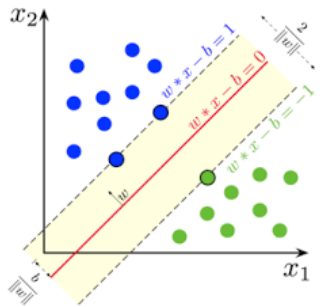
Evaluation results of clustering-based recommender system

We carefully tuned **hyperparameters** to optimize performance. Using the elbow method with the K-means algorithm, we determined the optimal number of clusters. Similarly, in PCA, we selected components explaining more than 90% of data variance. This method ensured our recommender system effectively groups users by preferences, minimizing information loss through dimensionality reduction.

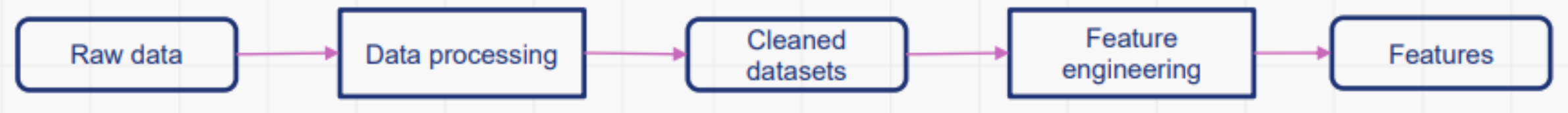
Average Number of New Courses Recommended per User: Upon assessing the system's efficacy, we discovered that, on average, it recommended approximately 36.587 new or unseen courses per user in the test dataset. This metric serves as a valuable indicator of the system's ability to diversify recommendations and introduce users to a broad range of learning opportunities beyond their previous engagements.

Top-10 Most Frequently Recommended Courses: Furthermore, our analysis of the most frequently recommended courses revealed compelling insights into the preferences of users within each cluster. Courses such as "WA0101EN," "DB0101EN," and "DS0301EN" emerged as the top recommendations, underscoring their popularity among users across different clusters. By leveraging these insights, we can further refine our recommendation strategies and tailor course offerings to better align with user preferences and learning objectives

Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system



1. Raw Data: The raw dataset contains user-item interactions, including user IDs, item IDs (courses), and ratings (enrollments). It consists of pairs of users and items with their corresponding ratings.

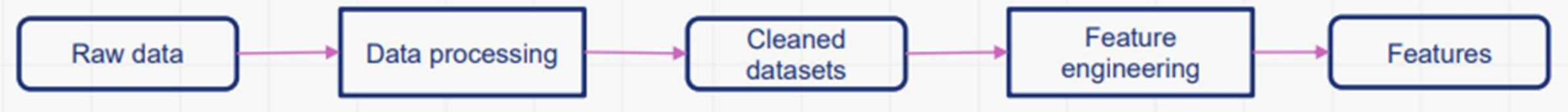
2. Data Processing: This step involves loading the dataset, handling missing values, removing duplicates, and formatting the data for analysis. It ensures the dataset is clean and prepared for further processing.

3. Cleaned Dataset: After processing, we obtain a structured dataset where irrelevant or erroneous data has been removed, missing values have been addressed, and the data is organized for model development.

4. Feature Engineering: Feature engineering enhances the recommendation system's performance by creating new features or transforming existing ones. It includes extracting relevant information from the dataset such as user-item interactions, timestamps, and demographics. This step also involves encoding categorical variables, scaling numerical features, and creating interaction terms.

5. Features (PCA Transformed): Features are the attributes used by the KNN-based recommender system for predictions. These features represent relationships between users and items, enabling the system to identify similarities and offer personalized recommendations. Examples include user-item interactions, user demographics, item characteristics, and contextual data.

Flowchart of NMF based recommender system



1. **Raw Data:** This refers to the initial dataset, which in our case includes course ratings. Raw data is typically unprocessed and may contain noise, missing values, or inconsistencies.

2. **Data Processing:** During this step, we preprocess the raw data to ready it for analysis. Tasks include handling missing values, eliminating duplicates, and formatting the data. Specifically, we utilized Pandas to pivot the data and construct a user-item matrix.

3. **Cleaned Dataset:** Following preprocessing, we obtain a refined dataset devoid of inconsistencies and suitable for analysis. This dataset typically features rows representing users, columns representing items, and cells containing ratings or interactions.

4. **Feature Engineering:** This step involves crafting new features or transforming existing ones to enhance machine learning model performance. For instance, features might encompass user-item interactions or latent factors derived from the NMF model.

5. **Features:** These variables or attributes serve as inputs for the machine learning model to generate predictions. In collaborative filtering, features may include user preferences, item characteristics, or latent factors representing users and items in a reduced-dimensional space.

Flowchart of Neural Network Embedding based recommender system

1. **Raw Data:** Initially, this refers to the dataset at hand, which includes course ratings data. Raw data is typically unprocessed and may contain noise, missing values, or inconsistencies.

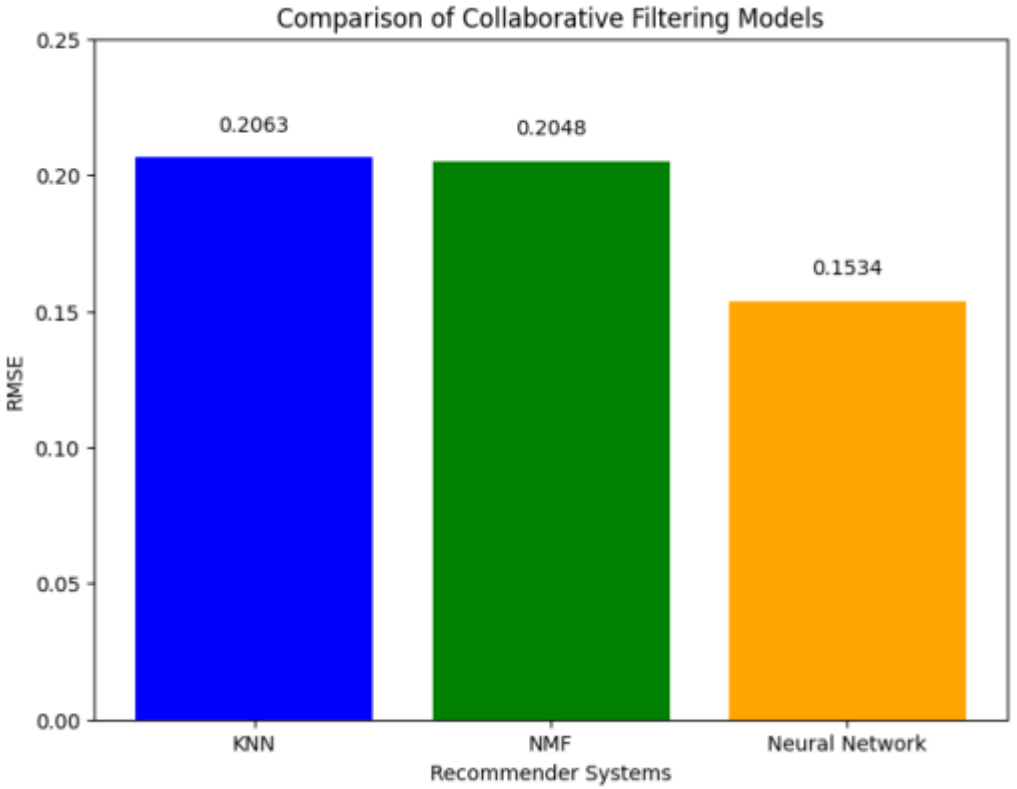
2. **Data Processing:** During this stage, we preprocess the raw data to ready it for analysis. This includes tasks such as handling missing values, removing duplicates, and formatting the data into a suitable structure. In our approach, we employed Pandas to pivot the data and construct a user-item matrix.

3. **Cleaned Dataset:** Following preprocessing, we derive a cleaned dataset that is devoid of inconsistencies and prepared for thorough analysis. Typically, this dataset organizes data with rows representing users, columns representing items, and cells containing ratings or interactions.

4. **Feature Engineering**: This critical step involves creating new features or transforming existing ones to enhance the performance of machine learning models. In our scenario, these features may encompass user-item interactions or latent factors generated using methods like the NMF model.

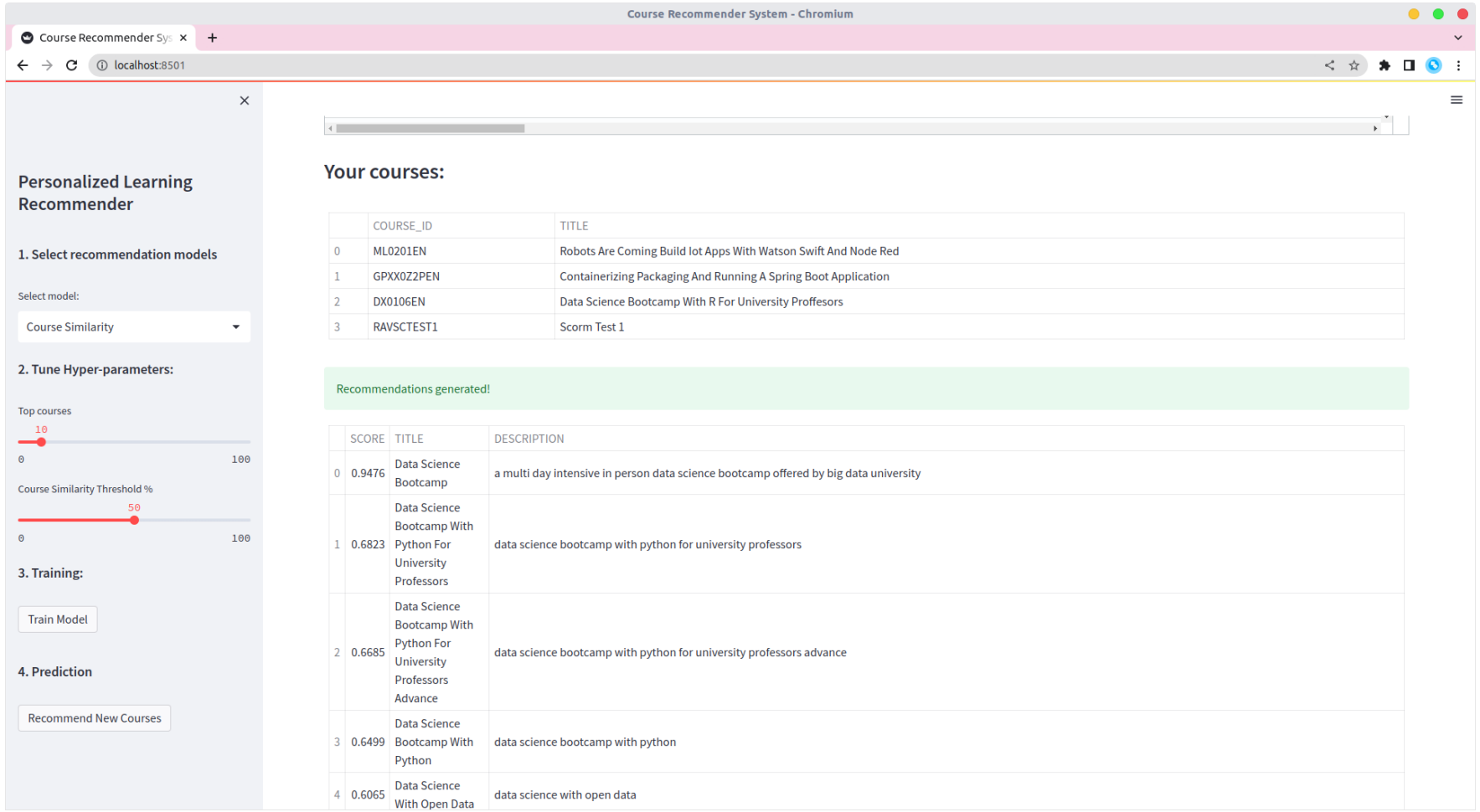
5. **Features:** These variables or attributes serve as inputs for the machine learning model to make predictions. Within the framework of collaborative filtering, features could include user preferences, item attributes, or latent factors that represent users and items in a lower-dimensional space.

Compare the performance of collaborative-filtering models

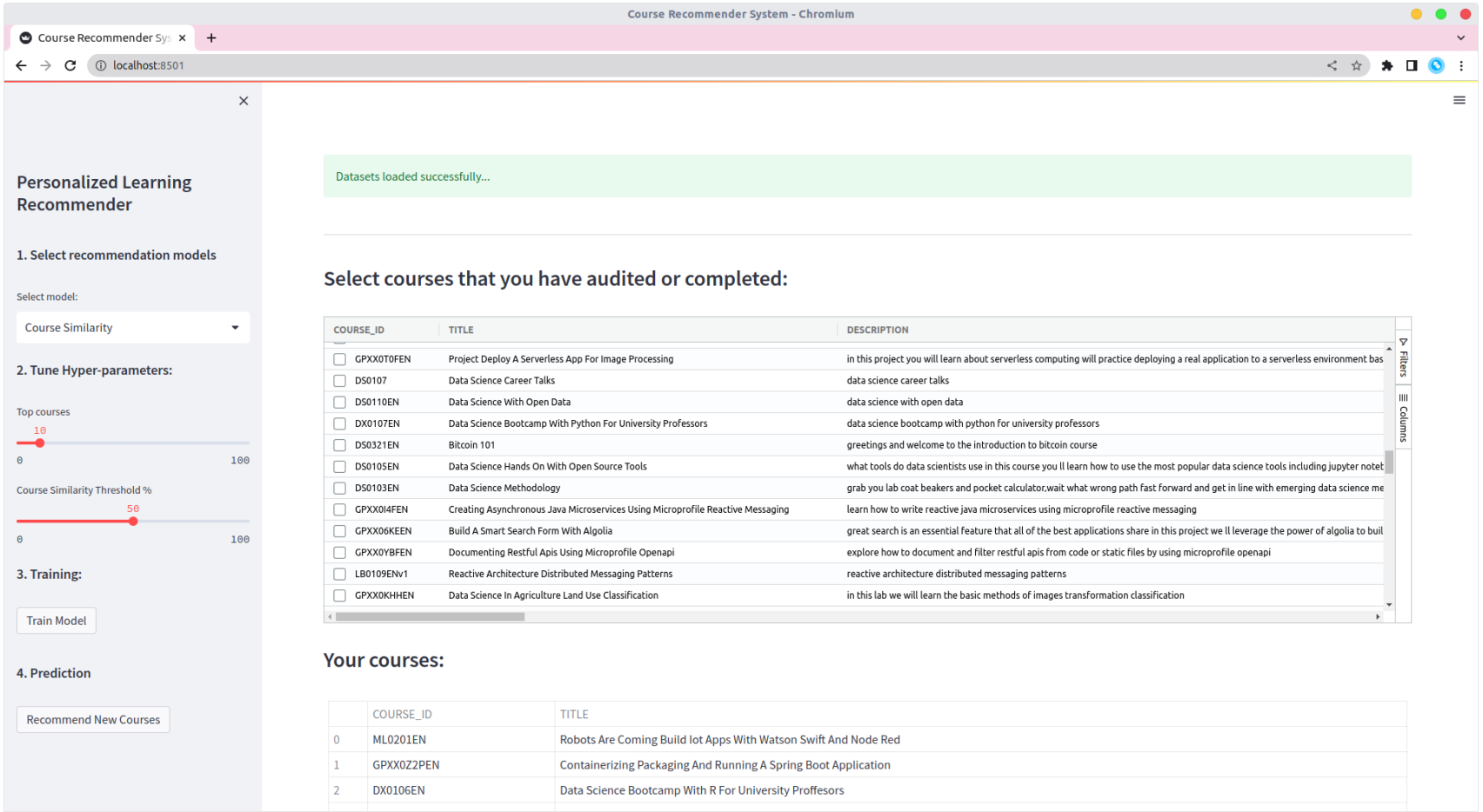


Based on the evaluation findings, the **Neural Network Embedding-based recommender system demonstrated the lowest RMSE value of 0.1534, signifying superior performance compared to the other two models in predicting user-item interactions.** Consequently, we identify the Neural Network Embedding-based recommender system as the most effective model for collaborative filtering within this particular context.

Optional: Build a course recommender system app with Streamlit(part 1)



Optional: Build a course recommender system app with Streamlit(part 2)



Conclusions (EDA)

From the comprehensive analyses conducted, several key insights have been gleaned concerning the course offerings and user engagement on the online learning platform. Firstly, an exploration of course genres revealed a wide spectrum of topics, with **backend development, machine learning, and databases emerging as the most popular genres based on enrollment numbers**. This diversity in course topics indicates **a broad interest among users**, catering to v**aried learning needs**.

Moreover, an analysis of course enrollments provided valuable insights into user behavior, showing that **a significant portion of users tend to complete courses** rather than merely audit them. This suggests a strong commitment to learning and a **high level of engagement** among the platform's users.

Additionally, an examination of the top 20 most popular courses highlighted a pronounced interest in data-related subjects. Courses such as "Python for Data Science," "Introduction to Data Science," and "Big Data 101" were particularly popular, reflecting the **increasing demand for skills in data analysis and machine learning**. This trend underscores the importance of offering courses that align with current industry demands and user interests.

These findings collectively emphasize the **necessity of providing a diverse range of courses** to cater to the varied interests of learners. They also highlight the critical role of data-driven insights in optimizing course offerings and enhancing user engagement on the platform. By leveraging these insights, the platform can better tailor its offerings to meet **the evolving needs** of its users, thereby fostering a more engaging and effective learning environment.

Conclusions (Content-based Recommender System Using User Profile And Course Genres)

The Content-based Recommender System Using User Profile and Course Genres begins by defining the task of generating course recommendations based on user profiles and course genre vectors. The procedure involves loading user profile and course genre dataframes, extracting user interests, identifying unfamiliar courses for each user, calculating recommendation scores, and filtering out courses that fall below a specified threshold. Once recommendation scores have been computed for all test users, the **system's performance is evaluated**. This evaluation includes determining the average number of recommended courses per user and identifying the top 10 most frequently recommended courses across all users. On average, approximately **61.82 courses are recommended per user**, indicating a substantial number of recommendations. The most frequently recommended courses include "TA0106EN," "GPXX0IBEN," and others, highlighting their popularity among the recommendations.

	USER	COURSE_ID	SCORE
0	37465	RP0105EN	27.0
1	37465	GPXX06RFEN	12.0
2	37465	CC0271EN	15.0
3	37465	BD0145EN	24.0
4	37465	DE0205EN	15.0
...	...	...	...
53406	2087663	excourse88	15.0
53407	2087663	excourse89	15.0
53408	2087663	excourse90	15.0
53409	2087663	excourse92	15.0
53410	2087663	excourse93	15.0

53411 rows × 3 columns

Top 10 most frequently recommended courses:

	COURSE_ID	RECOMMENDATION_COUNT
0	TA0106EN	608
1	GPXX0IBEN	548
2	excourse22	547
3	excourse21	547
4	ML0122EN	544
5	GPXX0TY1EN	533
6	excourse04	533
7	excourse06	533
8	excourse31	524
9	excourse73	516

Appendix

You can find everything related to this Project by visiting the given GitHub Repository URL below :

https://github.com/evanserlangga/IBMMLCP