Build a Personalized Online Course Recommender System with Machine Learning

Isaac Abiola

August 05, 2025

Outline

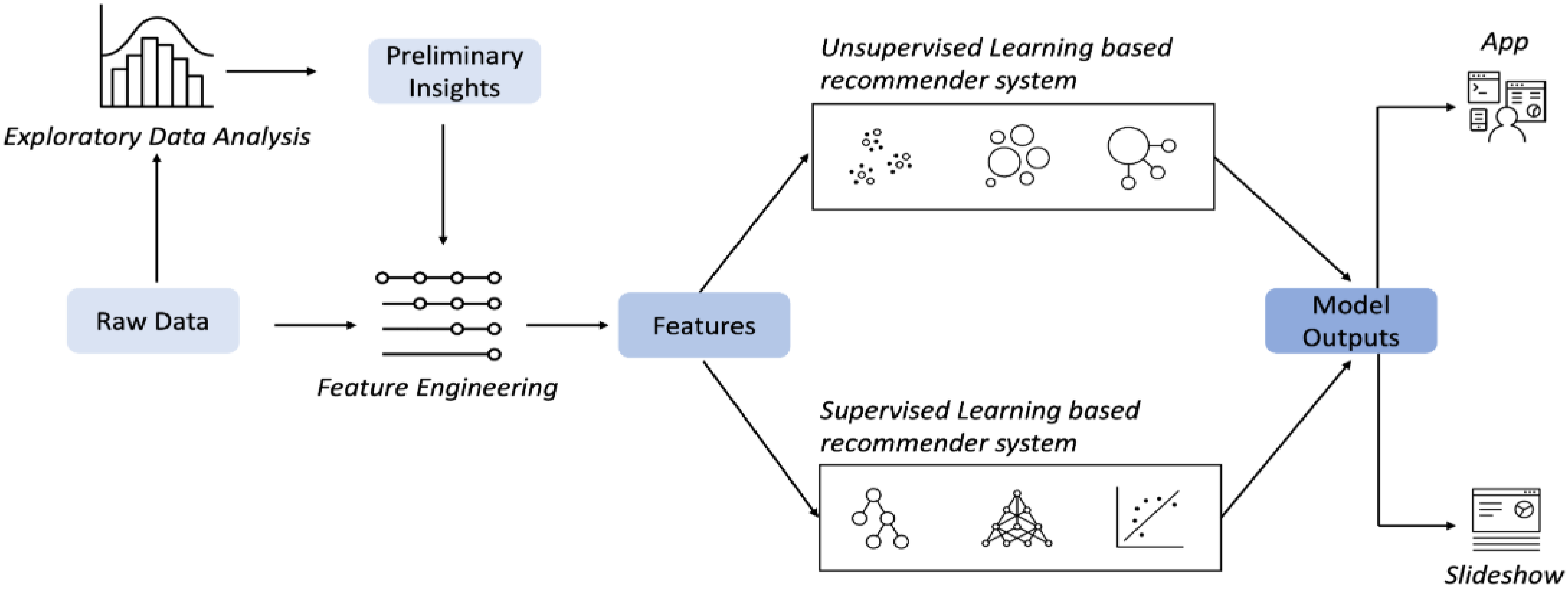
1. Introduction and Background
2. Exploratory Data Analysis
3. Content-based Recommender System using Unsupervised Learning
4. Collaborative-filtering based Recommender System using Supervised learning
5. Conclusion
6. Appendix

Introduction

- Project background and context
  - Our client, Coursera Inc, is a provider of online education/learning across several disciples. They are interested in increasing their subscribers base and deepening their relationships with them, by recommending courses that are most likely to be of interest to their subscribers. It is expected that this will result in increased subscriber satisfaction and loyalty thereby improving the viability and profitability of the institution.

- Problem states and hypotheses
  - In the project, we compare and contrast two alternative methods of making recommendation to subscribers. These are, (i) the collaborative based filtering method using supervised learning framework, and (ii) the content-based recommendation system that relies on unsupervised learning framework. The method that yields the best results will be recommended to the client.

Design Process Flow Map



Exploratory Data Analysis

Preliminary Insights

Raw Data

Feature Engineering

Features

Unsupervised Learning based recommender system

Supervised Learning based recommender system

Model Outputs

App

Slideshow

pecifically, you will undertake the tasks of
lling and understanding data

Exploratory Data Analysis & Features Engineering

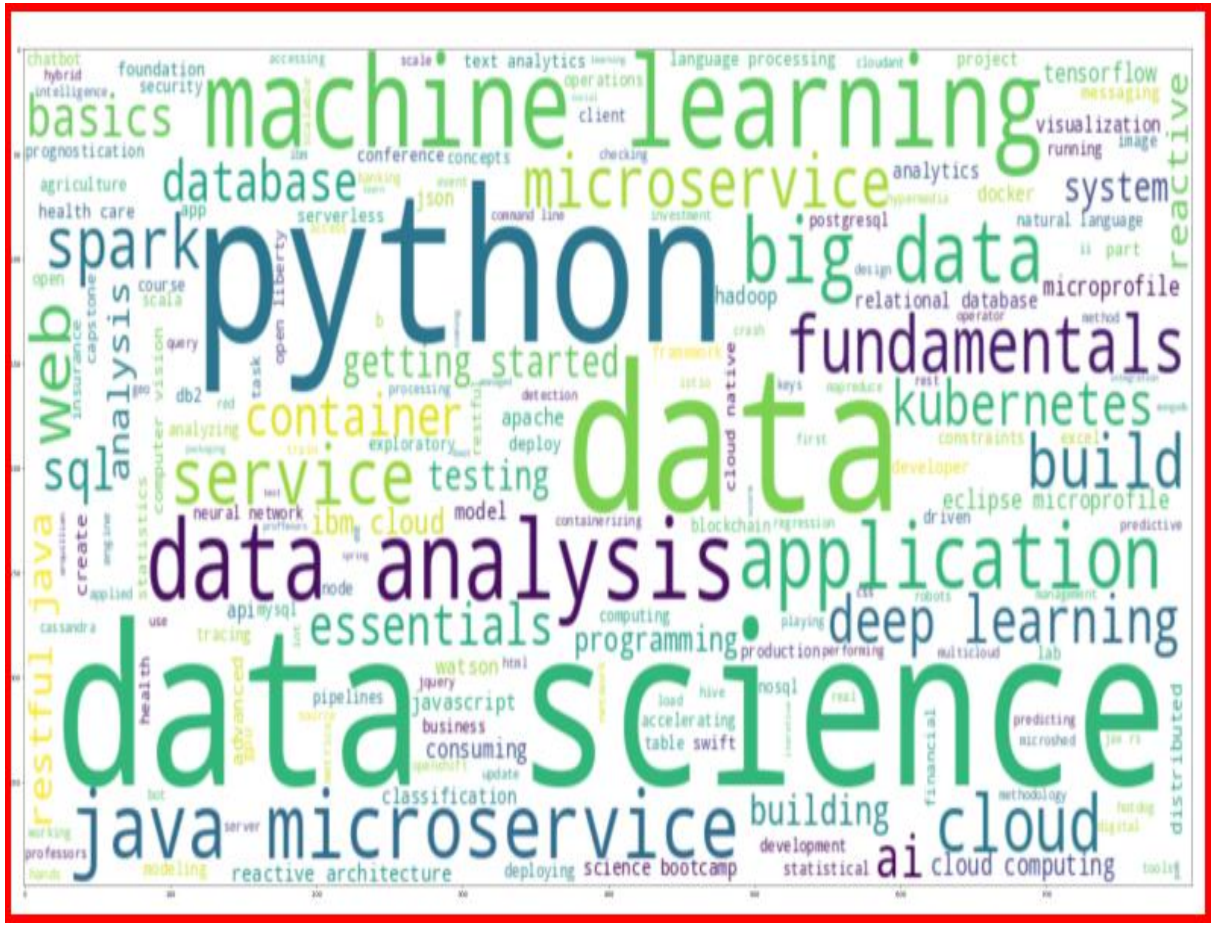
Data Source, Relevance and Quality

- The data was provided by our client, Coursera. The data was collected directly from subscribers interacting with the Coursera Inc platform. The recommendations and subscriber reactions are captured and stored in a secured server.
- The data is highly relevant for making recommendation as it is derived from previous recommendations suggested to subscribers and their responses to the recommendation.
- The data quality is reasonable and appropriate for the intended purpose.

Exploratory Data Analysis – Data Visualization

- This exploratory data analysis uses a Word Cloud to highlight the most common keywords in online course titles.
- It identifies themes such as Data Science, Python, data science, machine learning, big data, java microservice, cloud computing, among others as the dominant genre of interest
- The results help students and teachers understand the main IT skills currently in demand for digital learning.

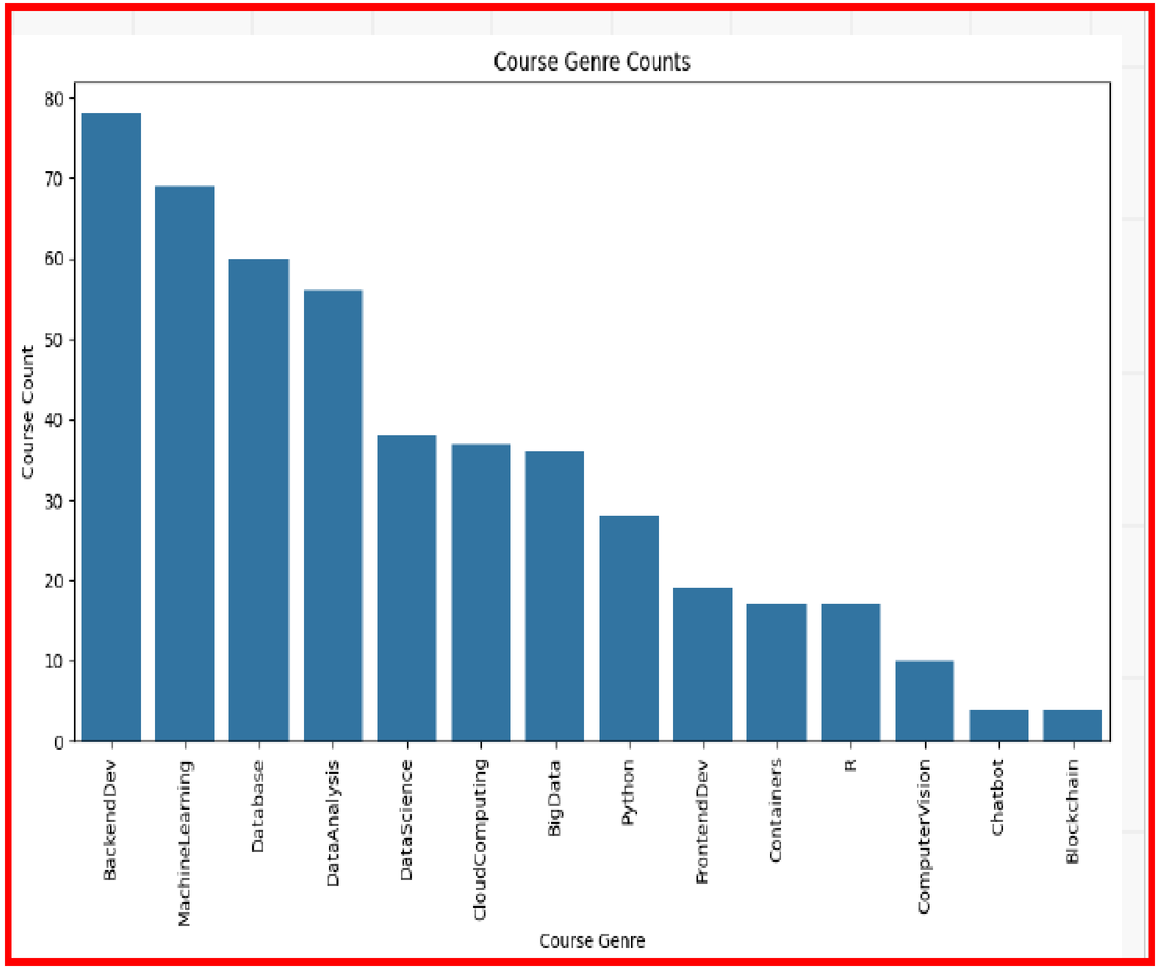
Plot a Word Cloud from Course Titles



Exploratory Data Analysis – Plots of Genres

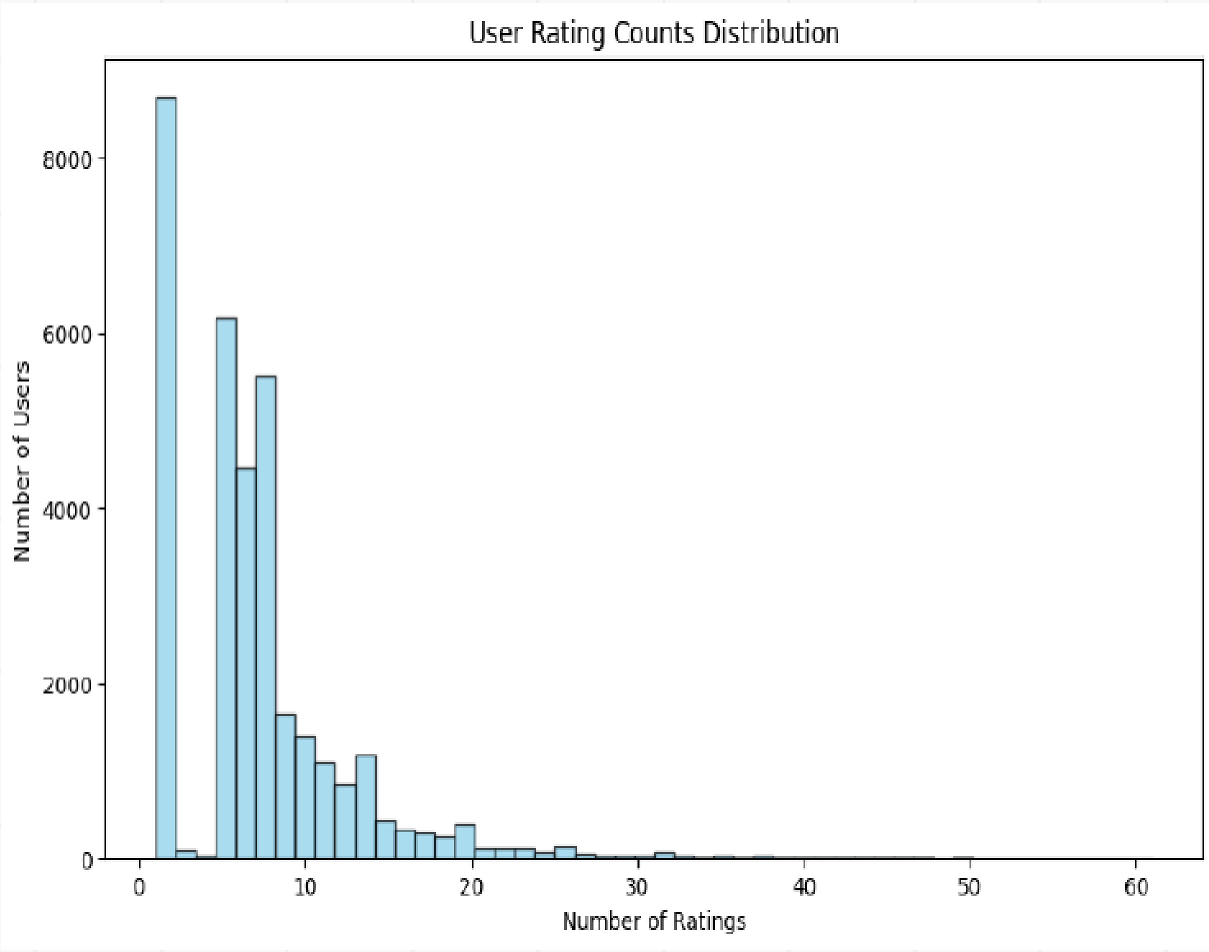
- An analysis of online course genres showed that Backend Development, Machine Learning, and Database topics are the most common, while Blockchain and Chatbot subjects are less prevalent.
- Visualized with a bar chart and table, these results offer useful insights into current trends for both learners and educators.

Analyze Course Genres

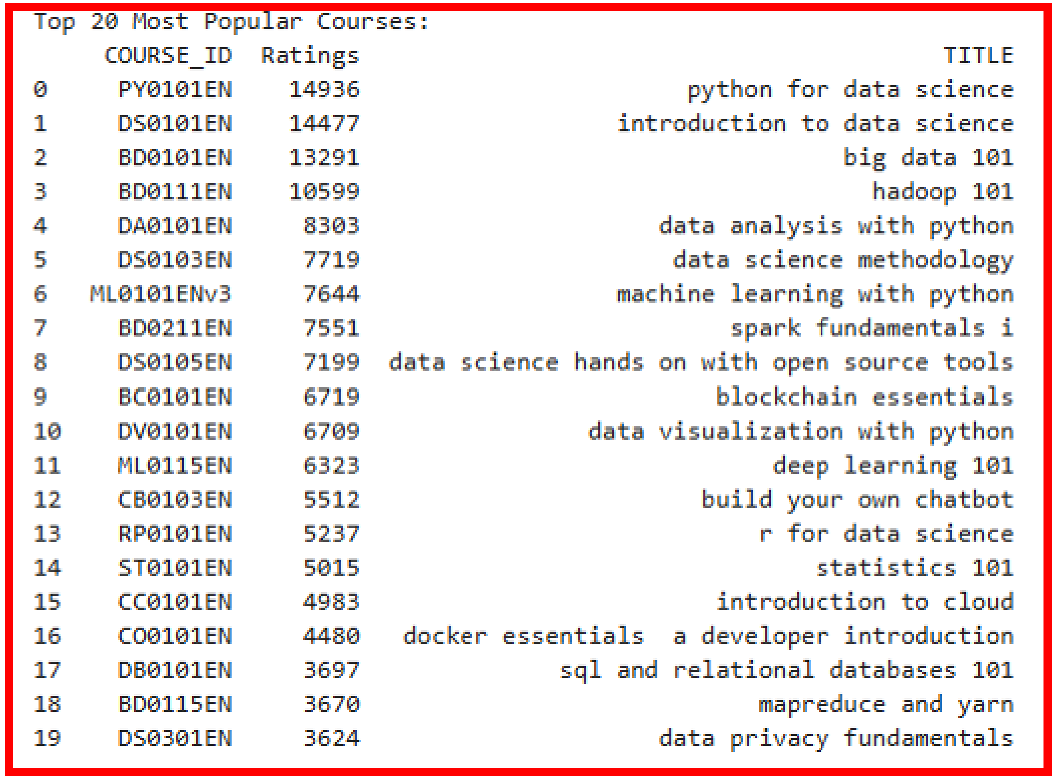


Exploratory Data Analysis – Subscriber Engagement

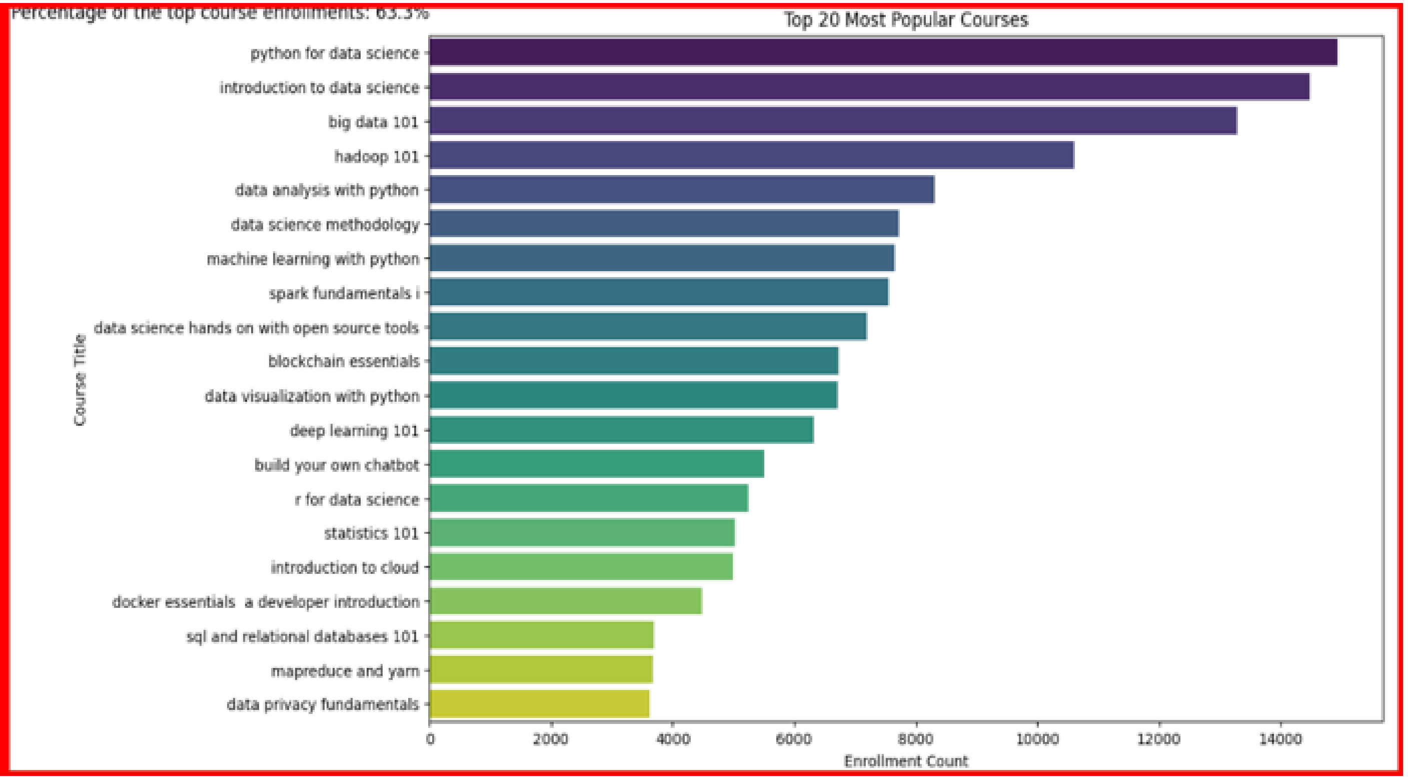
- An examination of course enrollment data identified 233,306 entries from 5,000 distinct users.
- he majority of users provided only a limited number of ratings, whereas a smaller subset contributed extensively.
- Understanding this distribution aids in developing strategies to enhance course offerings and overall user satisfaction.



Exploratory Data Analysis – 20 Most Popular Courses



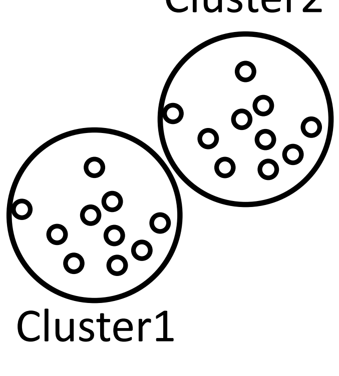
```
Top 20 Most Popular Courses:
     COURSE_ID  Ratings                                  TITLE
0     PY0101EN    14936                python for data science
1     DS0101EN    14477           introduction to data science
2     BD0101EN    13291                           big data 101
3     BD0111EN    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
```



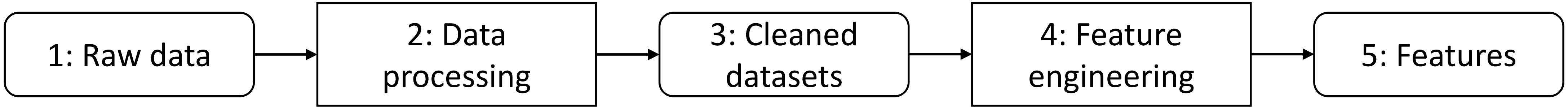
Percentage of the top course enrollments: 63.3%

- This task identified the top 20 courses by enrollment.
- Enrollment data was used to aggregate and rank courses, resulting in a list of course titles and their enrollment counts.
- This provides insight into learner preferences on the platform.
- The three most popular in descending order are as follows: Python for data science, Introduction to data science and big data 101.

Content-based Recommender System using Unsupervised Learning



Flowchart of Content-Based Recommender System Using User Profile and Course Genres



In the sequential transformation we leveraged user profiles and course genres to make recommendations based on the features derived from raw data.

1. **Raw Data: I**nitial dataset that captures all relevant information about users, courses, and their interactions. The key elements include data fields such as user demographic details, explicitly stated preferences, course metadata, and any available ratings or interaction logs.

2. **Data Processing:** In this step we handled missing values by imputation or removal, removed duplicate records to ensure consistency, transformed categorical fields (e.g., genres) into a machine-readable format, and normalized numerical attributes to a common scale.

3. **Cleaned Dataset:** Post-processing, we obtained a unified dataset that is free of inconsistencies. Each record now reliably represents a user, a course, and any interaction history necessary for downstream analysis.

4. **Feature Engineering:** We construct user profile vectors by aggregating preferences across genres and past ratings, generate course genre vectors via one-hot encoding, TF-IDF weighting, or embeddings, and incorporate additional features such as course difficulty or duration.

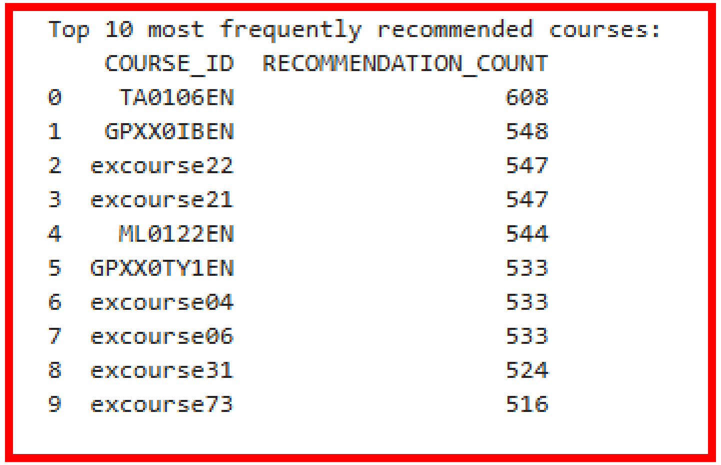
5. **Features:** In this stage, **t**he final feature set consists of numerical vectors for both users and courses. These vectors act as inputs to the content-based recommender engine, where similarity computations drive personalized course suggestions.

Evaluation results of user profile-based recommender system

- **Hyper parameter Settings:** We set a recommendation score threshold of 10.00 to filter out low scoring recommendations. This threshold determines which courses are considered relevant enough to be recommended to users. Besides, we adjusted other hyperparameters such as feature representation methods or similarity metrics during the implementation of the recommender system

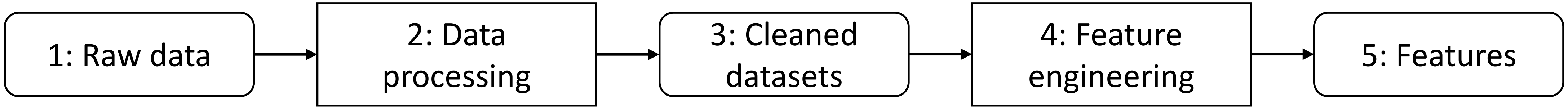
- **Average Number of New Courses Recommended per User:**
  The average number recommended is 62 (or 61.82) courses per user.

- Top 10 Most Frequently Recommended Courses:
  - The lists of ten most recommended courses is shown in the table to the right.
  - The recommended course is identified by its COURSE_ID and the RECOMMENDATION_COUNT indicates how many times it was recommended to users.
  - Recommendations are based on how well each course matches users' interests.



```
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
```

Flowchart of Content-Based Recommender System Using Course Similarity



- The flowchart outlines the sequential transformation from raw input data to actionable recommendations. In the analysis, we leveraged course similarity to tailor course suggestions based on feature similarities.

1. **Raw Data:** The initial dataset containing information about various courses, including their titles, descriptions, and other relevant attributes.

2. **Data Processing:** Preprocessing of raw data which includes tokenization and lemmatization to break down the text into individual words and convert them to their base forms. The stop words, which typically have little semantic value, are removed, and so are outliers (irrelevant or noisy data points).

3. **Cleaned Dataset:** Following data processing, the dataset undergoes cleaning, resulting in a unified dataset free of inconsistencies. Each record now reliably represents a course and its attributes necessary for downstream analysis.

4. **Feature Engineering:** Transform the cleaned dataset into numerical features that represent the courses. For example, calculate TF-IDF (Term Frequency-Inverse Document Frequency) vectors for each course based on the words they contain and their importance in the dataset.

5. **Features:** The final set of features are used to represent each course. These are the TF-IDF vectors obtained from the feature engineering step. They are then used to calculate similarities between courses and generate recommendations based on content similarity.

Evaluation results of course similarity based recommender system

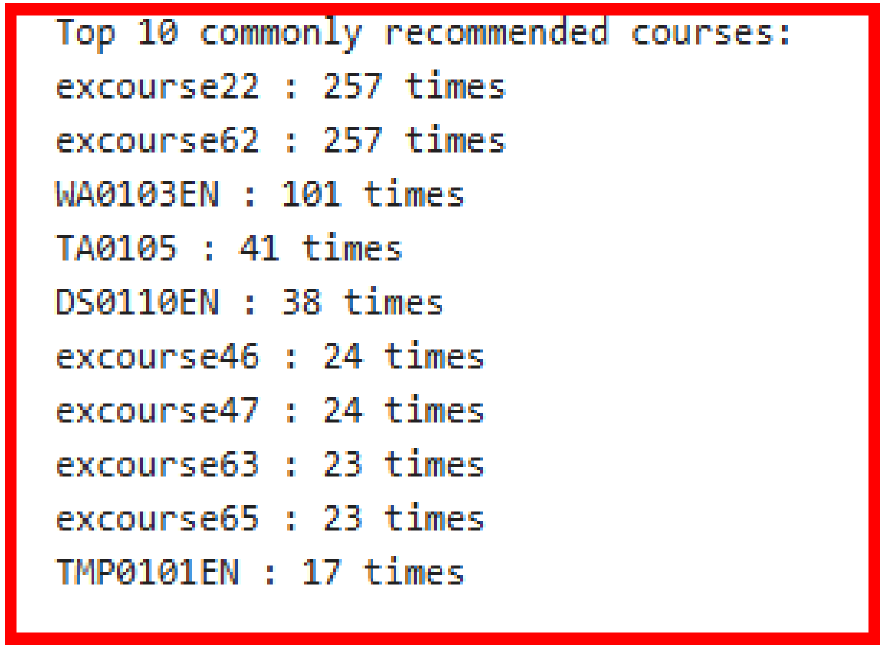
- **Hyper-parameter Settings:** The hyper-parameter selected for course similarity-based recommender system involved setting a similarity threshold of 0.6. This threshold established the minimum level of similarity required between courses to be recommended to users.

- The average number of new courses recommended per user one (or 0.987).

- The ten most frequently recommended courses are listed in descending order in the table on the right hand side.. The top three most recommended are "excourse22" and "excourse62" with received the highest number of recommendations, each appearing 257 times, followed by "WA0103EN" with 101 recommendations.

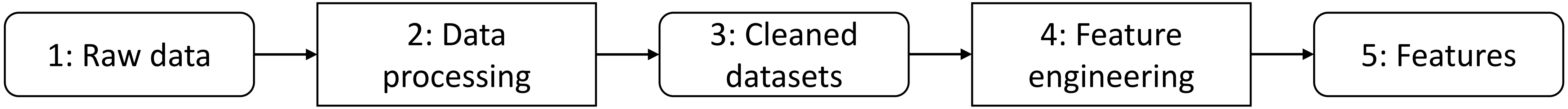
- Other frequently suggested courses, include "TA0105" and "DS0110EN," further indicate their relevance and popularity within the user base.

- The results provide meaningful insights into the performance and effectiveness of the course similarity-based recommender system, informing potential enhancements and optimizations for future iterations.



```
Top 10 commonly recommended courses:
excourse22 : 257 times
excourse62 : 257 times
WA0103EN : 101 times
TA0105 : 41 times
DS0110EN : 38 times
excourse46 : 24 times
excourse47 : 24 times
excourse63 : 23 times
excourse65 : 23 times
TMP0101EN : 17 times
```

Flowchart of clustering-based recommender system



- The flowchart outlines the sequential steps involved in transforming raw input data to features used in assigning clusters/segments.

1. **Raw Data:** Original user profile feature vectors. The features capture preferences across course genres (e.g., Machine Learning, Data Science, Cloud Computing),

2. **Data Processing:** Handle missing values, outliers, and data quality issues. Next, apply normalization (e.g., StandardScaler) to align feature scales.

3. **Cleaned Dataset:** The resulting dataset after preprocessing is standardized features with a mean = 0 and standard deviation = 1.

4. **Feature Engineering:** Apply Principal Component Analysis (PCA) to reduce dimensionality of data. The PCA identifies components explaining maximum variance

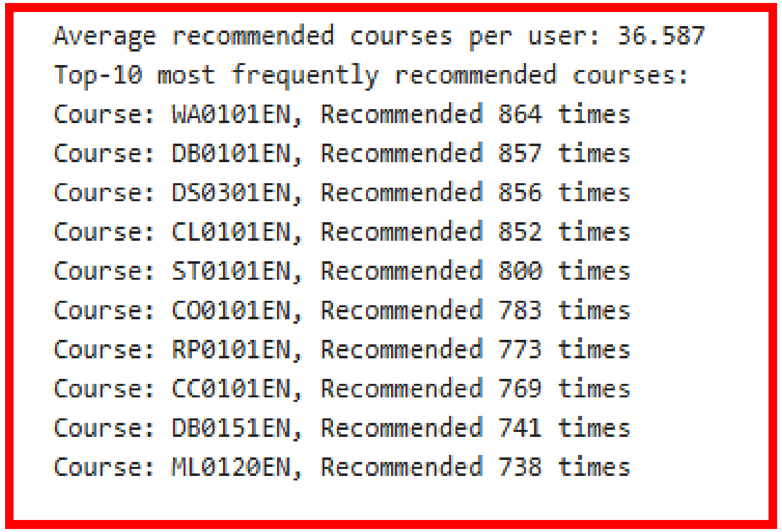
5. **Features: The PCA Transformed series, a l**ower-dimensional representation of user profiles. Each PCA feature encapsulates combined information from original genres.

Evaluation results of clustering-based recommender system

- **Hyper-parameter Settings:** Hyperparameters were adjusted to improve system performance. The ideal number of clusters was identified using the K-means algorithm with the elbow method. For principal component analysis (PCA), the number of components selected accounted for over 90% of data variance. This method allowed for user grouping based on preferences while reducing dimensionality and limiting information loss.

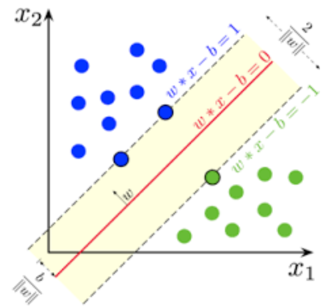
- **The average number of new cou**rses recommended per user is 37 (or 36.587).

- **Top-10 Most Frequently Recommended Courses**:
    - The commonly recommended courses provided insights into user preferences within individual clusters.
    - Courses including "WA0101EN," "DB0101EN," and "DS0301EN" emerged as frequent recommendations, reflecting their popularity across clusters.
    - These findings can be used to further develop recommendation strategies and better match course offerings to user preferences and educational objectives.

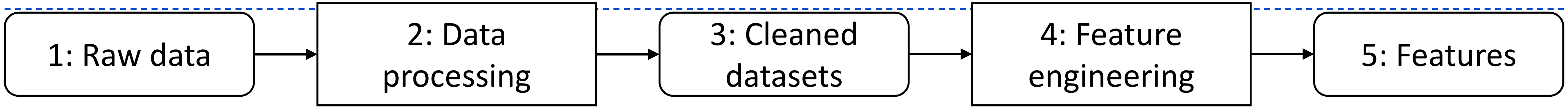


```
Average recommended courses per user: 36.587
Top-10 most frequently recommended courses:
Course: WA0101EN, Recommended 864 times
Course: DB0101EN, Recommended 857 times
Course: DS0301EN, Recommended 856 times
Course: CL0101EN, Recommended 852 times
Course: ST0101EN, Recommended 800 times
Course: CO0101EN, Recommended 783 times
Course: RP0101EN, Recommended 773 times
Course: CC0101EN, Recommended 769 times
Course: DB0151EN, Recommended 741 times
Course: ML0120EN, Recommended 738 times
```

Collaborative-filtering Recommender System using Supervised Learning



Flowchart of KNN based recommender system



1: Raw data

2: Data processing

3: Cleaned datasets

4: Feature engineering

5: Features

- The flowchart outlines the sequential steps involved in transforming raw input data to features used in assigning nearest neighbors.

1. **Raw Data:** Original interactions between users and items, typically a matrix of user IDs, item IDs, and ratings or implicit feedback.

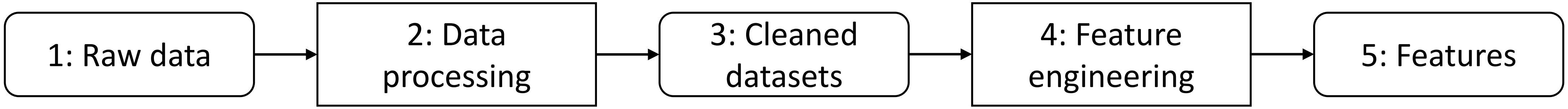
2. **Data Processing:** Load the raw dataset, handle missing values, remove duplicates, and format the data into a consistent structure for analysis.

3. **Cleaned Dataset:** The result of processing: a filtered and validated table of user–item pairs, with no gaps or inconsistencies, ready for feature work.

4. **Feature Engineering:** Create or transform variables that capture meaningful patterns—examples include user demographics, item metadata, or aggregated statistics like “average rating per item.” Use PCA transformation -- to reduce dimensionality by applying Principal Component Analysis (PCA) on the feature matrix, retaining the components that explain the most variance in user–item behavior.

5. **Features:** Compute for each target user, the distances (e.g., Euclidean, cosine) to all other users (or items) and select the top k nearest neighbors. Next generate recommendations based on aggregate preferences of the nearest neighbors (such as weighted ratings) to predict scores for unseen items, then rank and recommend the top candidates.

Flowchart of NMF based recommender system



- The flowchart outlines the sequential steps involved in transforming raw input data to features used in assigning nearest neighbors.

1. **Raw Data:** The original data on each user and items. It typically contains the user IDs, item IDs, and ratings or implicit feedback. This data may contain missing values, outliers and inconsistent data.

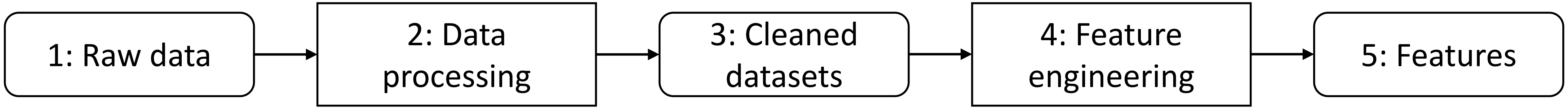
2. **Data Processing:** Load the raw dataset then handle missing values, remove duplicates, and format the data into a consistent structure for analysis.

3. **Cleaned Dataset:** The result of processing: a filtered and validated table of user–item pairs, with no gaps or inconsistencies, ready for feature work.

4. **Feature Engineering:** Create or transform variables that capture meaningful patterns—examples include user demographics, item metadata, or aggregated statistics like "average rating per item." Use PCA transformation -- to reduce dimensionality by applying Principal Component Analysis (PCA) on the feature matrix, retaining the components that explain the most variance in user–item behavior.

5. **Features: T**he variables or characteristics utilized by machine learning models for predictive purposes. In collaborative filtering, features may encompass user preferences, item attributes, or latent factors that represent users and items within a lower-dimensional space.

Flowchart of Neural Network Embedding based recommender system



- The flowchart outlines the sequential steps involved in transforming raw input data to features used in scores in neural network models.

1. **Raw Data:** The original data on each user and items. It typically contains the user IDs, item IDs, and ratings or implicit feedback. This data may contain missing values, outliers and inconsistent data.

2. **Data Processing:** Load the raw dataset then handle missing values, remove duplicates, and format the data into a consistent structure for analysis.

3. **Cleaned Dataset:** The result of processing: a filtered and validated table of user–item pairs, with no gaps or inconsistencies, ready for feature work.

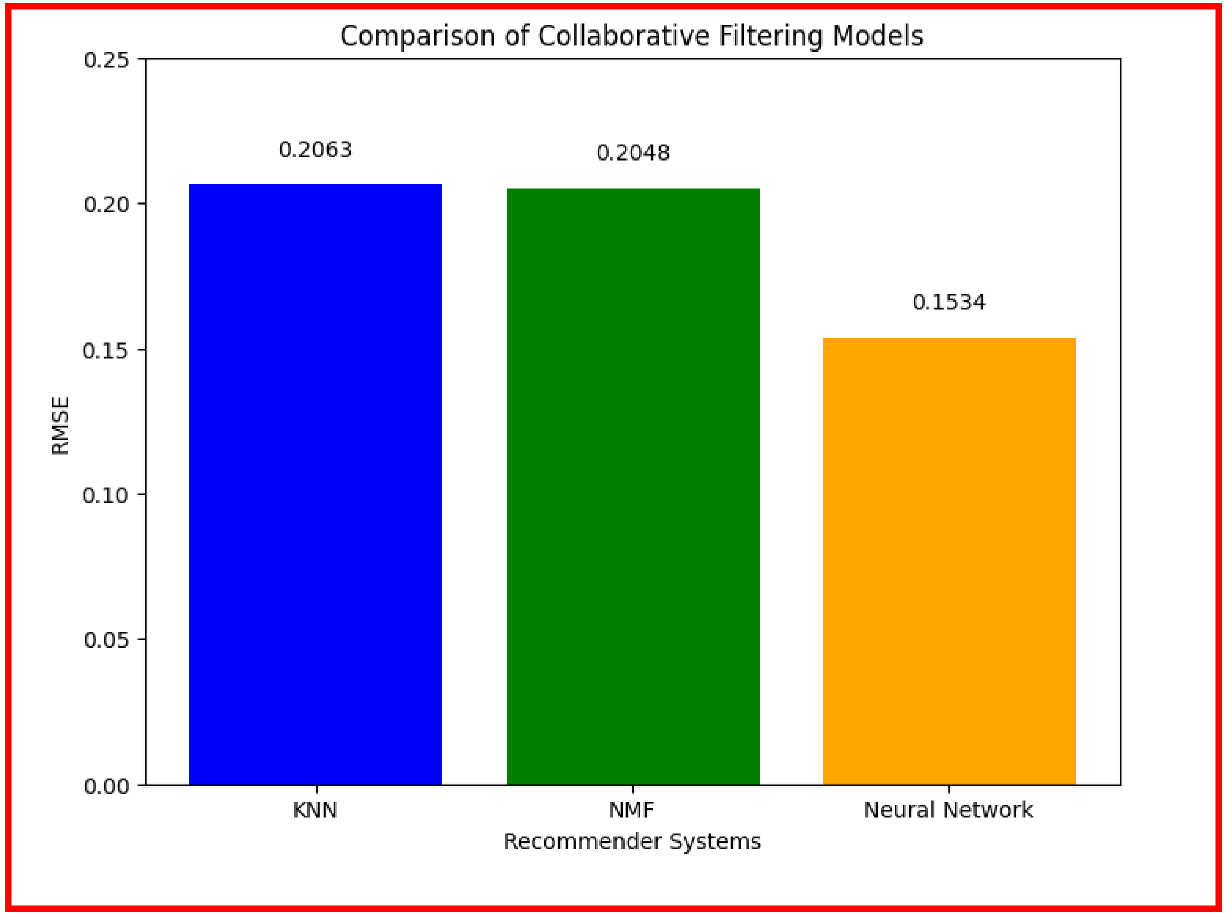
4. **Feature Engineering:** Create or transform variables that capture meaningful patterns—examples include user demographics, item metadata, or aggregated statistics like "average rating per item." Use PCA transformation -- to reduce dimensionality by applying Principal Component Analysis (PCA) on the feature matrix, retaining the components that explain the most variance in user–item behavior. The variables can also include interaction variable sets. The variables are also one-hot encoded before embedding

5. **Features:** The variables or characteristics utilized by machine learning models for predictive purposes. In collaborative filtering, features may encompass user preferences, item attributes, or latent factors that represent users and items within a lower-dimensional space.

Compare the performance of collaborative-filtering models

- The bar chart on the right-hand side shows that the **Neural Network Embedding-based recommender system** achieved the lowest RMSE of 0.1534.

- It is the most effective collaborative filtering model in this scenario



Conclusions –1 of 2

1. **The exploratory data analysis section conveys the following:**

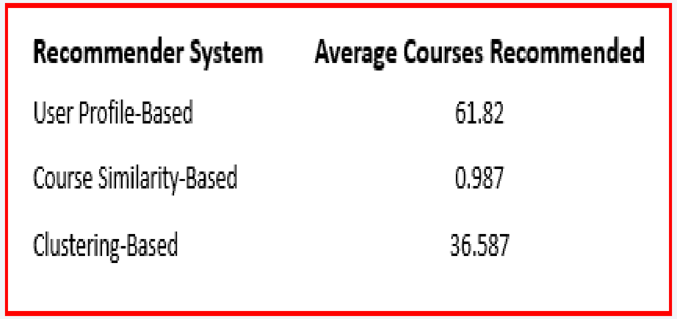
The word cloud analysis provides a visual representation of the most prominent IT skills and emerging trends. It reveals a strong user interest in pursuing learning opportunities across several high-demand domains, including:

- ○**Programming Languages**: Python and Java, with a particular emphasis on microservice architecture
- ○**Data-Centric Disciplines**: Data science, data analysis, and data engineering
- ○**Artificial Intelligence**: Machine learning and deep learning technologies

This insight underscores a growing demand for technical proficiency in scalable software development, data-driven decision-making, and intelligent automation. These trends align with industry-wide shifts toward cloud-native applications, predictive analytics, and AI-powered solutions.

2. **Comparison of Content-Based Recommender Systems Using Unsupervised Learning:**

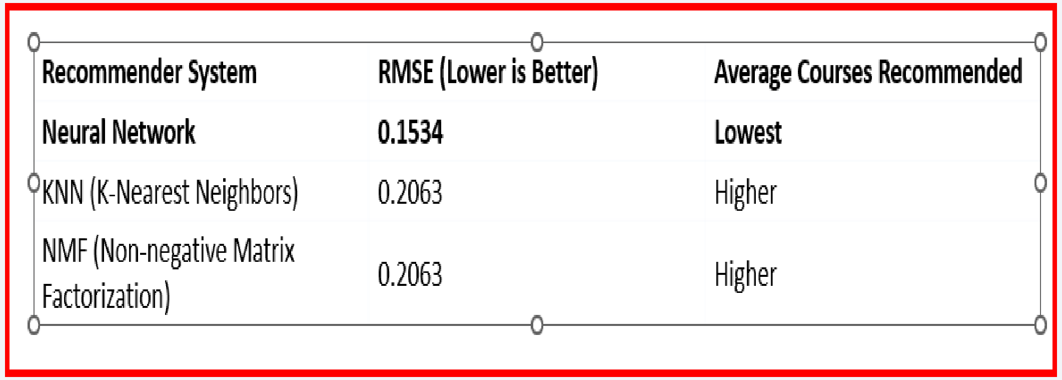
- In an evaluation of three unsupervised learning methodologies for content-based recommendation systems, **the user profile-based recommender system** emerged as the most effective in terms of volume of recommendations.
- This suggests the following:
  - **User Profile-Based Recommender:** Delivers the highest number of recommendations per user, suggesting it captures user preferences more comprehensively.
  - Course Similarity-Based Recommender: Offers very few recommendations, possibly due to strict similarity thresholds or sparse feature overlap.
  - Clustering-Based Recommender: Performs moderately well, grouping users or courses into clusters to generate relevant suggestions.



Recommender System	Average Courses Recommended
User Profile-Based	61.82
Course Similarity-Based	0.987
Clustering-Based	36.587

Conclusions –2 of 2

3. **Comparison of Content-Based Recommender Systems Using Supervised Learning:**
   - In an evaluation of three supervised learning methodologies for collaborative-filtering Recommender System, the neural network-based recommender system. It emerged as the most effective in terms of root mean square error **RMSE**.
   - This suggests the following:
     - **User Profile-Based Recommender:** Delivers the highest number of recommendations per user, suggesting it captures user preferences more comprehensively.
     - Course Similarity-Based Recommender: Offers very few recommendations, possibly due to strict similarity thresholds or sparse feature overlap.
     - Clustering-Based Recommender: Performs moderately well, grouping users or courses into clusters to generate relevant suggestions.



Recommender System	RMSE (Lower is Better)	Average Courses Recommended
Neural Network	**0.1534**	**Lowest**
KNN (K-Nearest Neighbors)	0.2063	Higher
NMF (Non-negative Matrix Factorization)	0.2063	Higher

- **RMSE (Root Mean Square Error)** measures the difference between predicted and actual ratings. A lower RMSE indicates better predictive accuracy.
- The Neural Network model had the lowest RMSE, meaning it made the most accurate predictions of user preferences.
- **Why Neural Networks Performed Best:**
  - Non-linear modeling -- Neural networks can capture complex, non-linear relationships between users and items that simpler models like KNN and NMF might miss.
  - Feature learning -- They can automatically learn latent features from user-item interactions, improving personalization.
  - Scalability -- With enough data and tuning, neural networks scale well and adapt to diverse user behaviors.
- **Next Steps or Considerations:**
  - It would be interesting to see how precision, recall, or diversity compare across models—not just RMSE.

Appendix

https://github.com/iabiola1979/iabiola1979-IBM-Machine-Learning-Prof.-Certs