Machine Learning Capstone

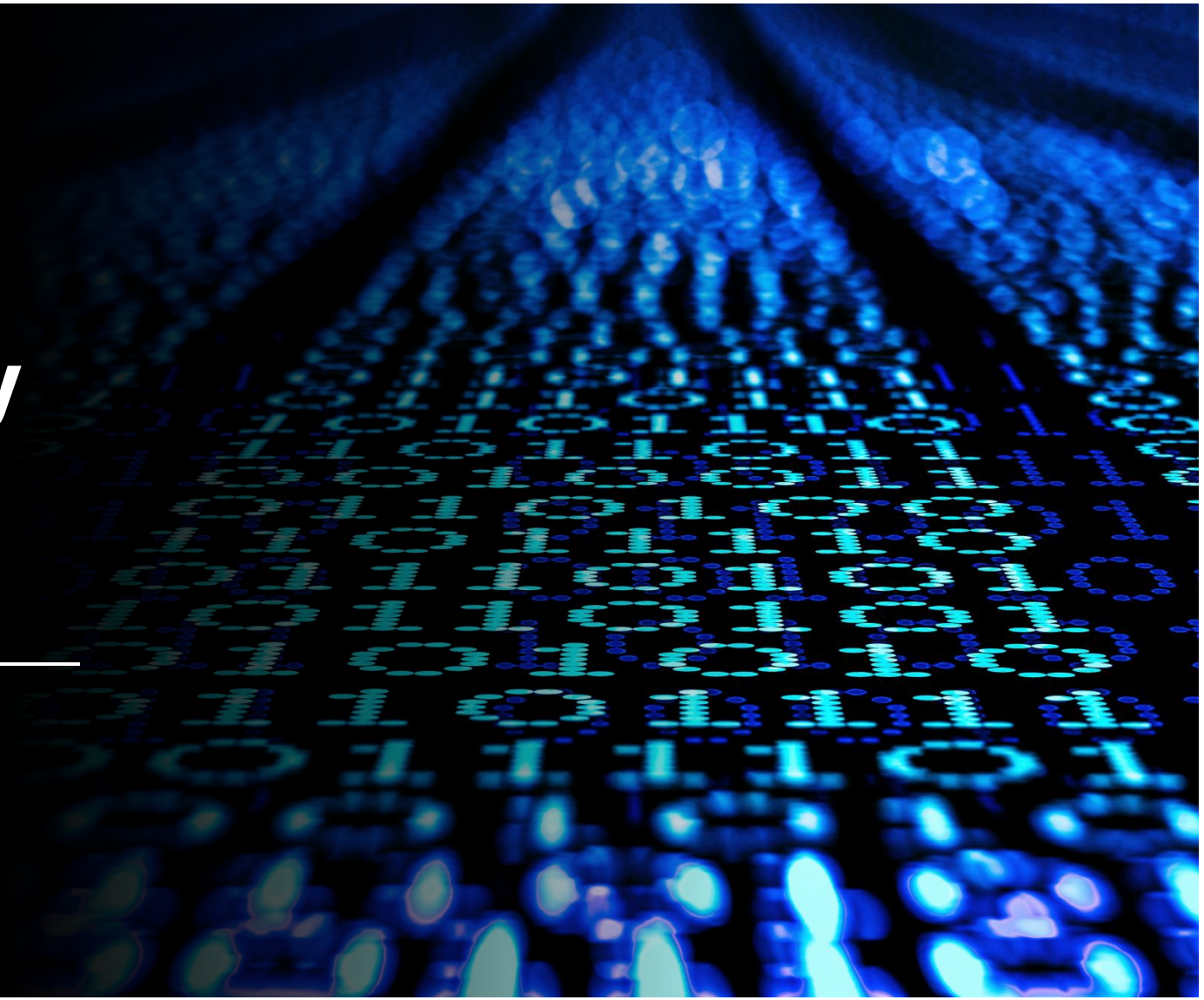
Ben Cottrell

Introduction and Background

The main objective of this project is to improve learners' learning experience via helping them find new and interesting courses, assisting with their learning paths.

To do this, we will explore and compare various machine learning models to find the model with the best performance

Exploratory Data Analysis



Course Counts per Genre

The dataset was examined to identify the popularity of various online course topics. By calculating course counts for each genre and visualizing the results through a bar chart and table, insights into the most sought-after subjects were revealed.

The analysis revealed that Backend Development, Machine Learning and Database courses are among the most prevalent, while Blockchain and Chatbot courses are less common.

This exploration provides valuable insights for learners and educators seeking to understand current trends in online education.

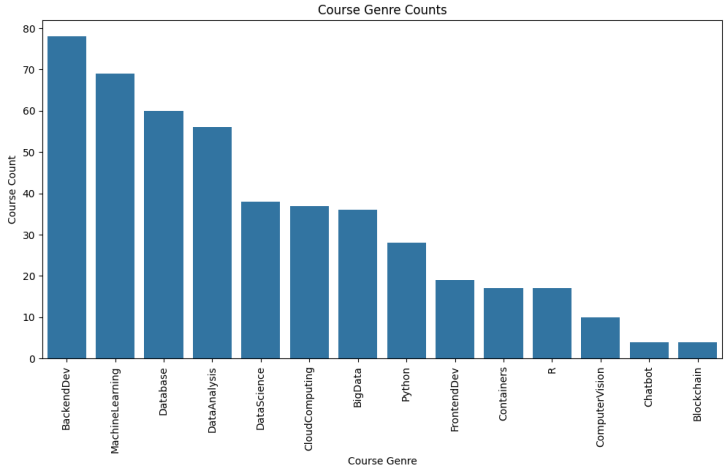
```
## WRITE YOUR CODE HERE

# Calculate the course count for each genre
genre_counts = course_df[genres].sum(axis=0)

# Convert the counts into a DataFrame
genre_counts_df = pd.DataFrame(genre_counts, columns=['Count'])

# Sort the genre counts
sorted_genre_counts = genre_counts_df.sort_values(by='Count', ascending=False)
```

▶ Click here for Hints

We can also visualize course genre counts using a bar chart:

TODO: Use seaborn barplot or other plot methods to plot course genre counts using a barchart. Tl

```
# WRITE YOUR CODE HERE
plt.figure(figsize=(12, 6))
sns.barplot(x=sorted_genre_counts.index, y='Count', data=sorted_genre_counts)
plt.xticks(rotation=90)
plt.xlabel('Course Genre')
plt.ylabel('Course Count')
plt.title('Course Genre Counts')
plt.show()
```

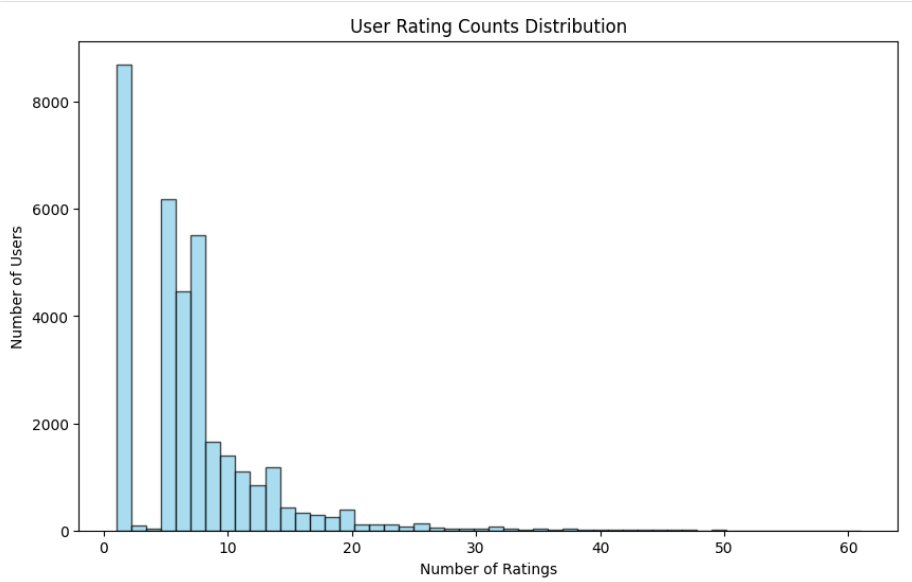


Course Enrolment Distribution

In the task of analysing course enrolments, the dataset was examined to gain insights into user engagement and interactions with online courses. By aggregating the rating counts for each user, it was found that the dataset comprises 233,306 enrolment records from 5,000 unique users.

The histogram distribution of user rating counts illustrates varying levels of engagement, with the majority of users giving relatively few ratings, with a smaller proportion giving a larger number of ratings. This analysis sheds light on the distribution of user interactions with online courses, providing valuable insights for optimizing course offerings and enhancing user experiences.

```
# WRITE YOUR CODE HERE
# Plot the histogram of user rating counts
plt.figure(figsize=(10, 6))
user_rating_counts.hist(bins=50, color='skyblue', edgecolor='black', alpha=0.7)
plt.xlabel('Number of Ratings')
plt.ylabel('Number of Users')
plt.title('User Rating Counts Distribution')
plt.grid(False)
plt.show()
```



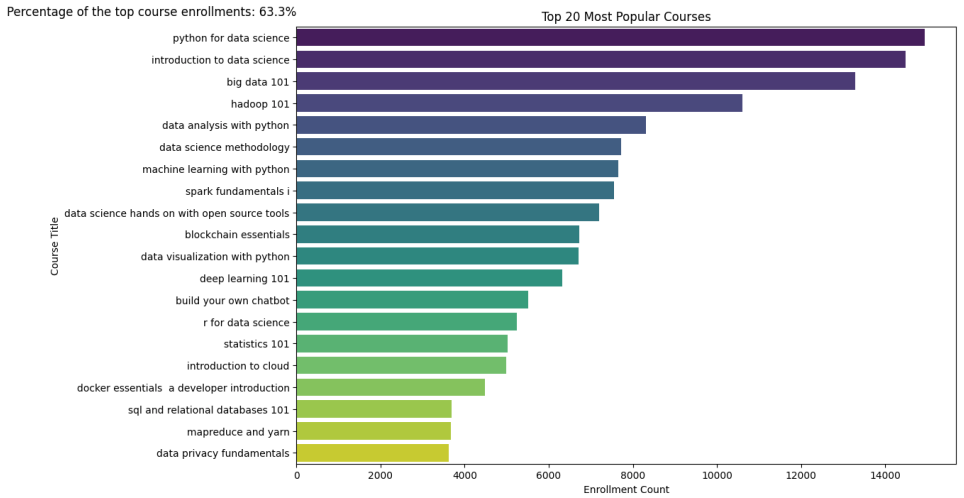
20 Most Popular Courses

In this task, the objective was to find the top 20 most popular courses based on enrolment counts. Using the enrolment data, the courses were aggregated, sorted and the top 20 courses were identified.

The resulting list showcases the titles and enrolment counts of these highly sought-after courses, providing insights into the preferences of learners in the online learning platform.

Looking at the graph, we see that ‘python for data science’ is the most popular course with 14,936 ratings.

Percentage of the top course enrollments: 63.3%



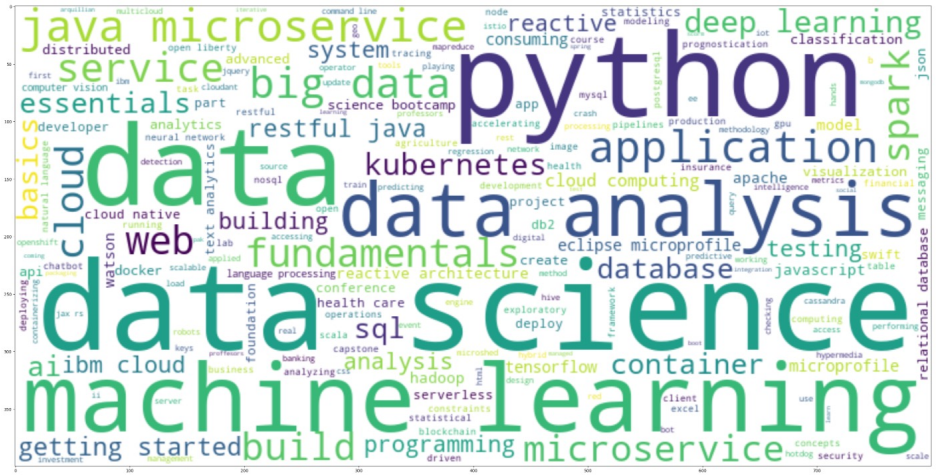
Top 20 Most Popular Courses:

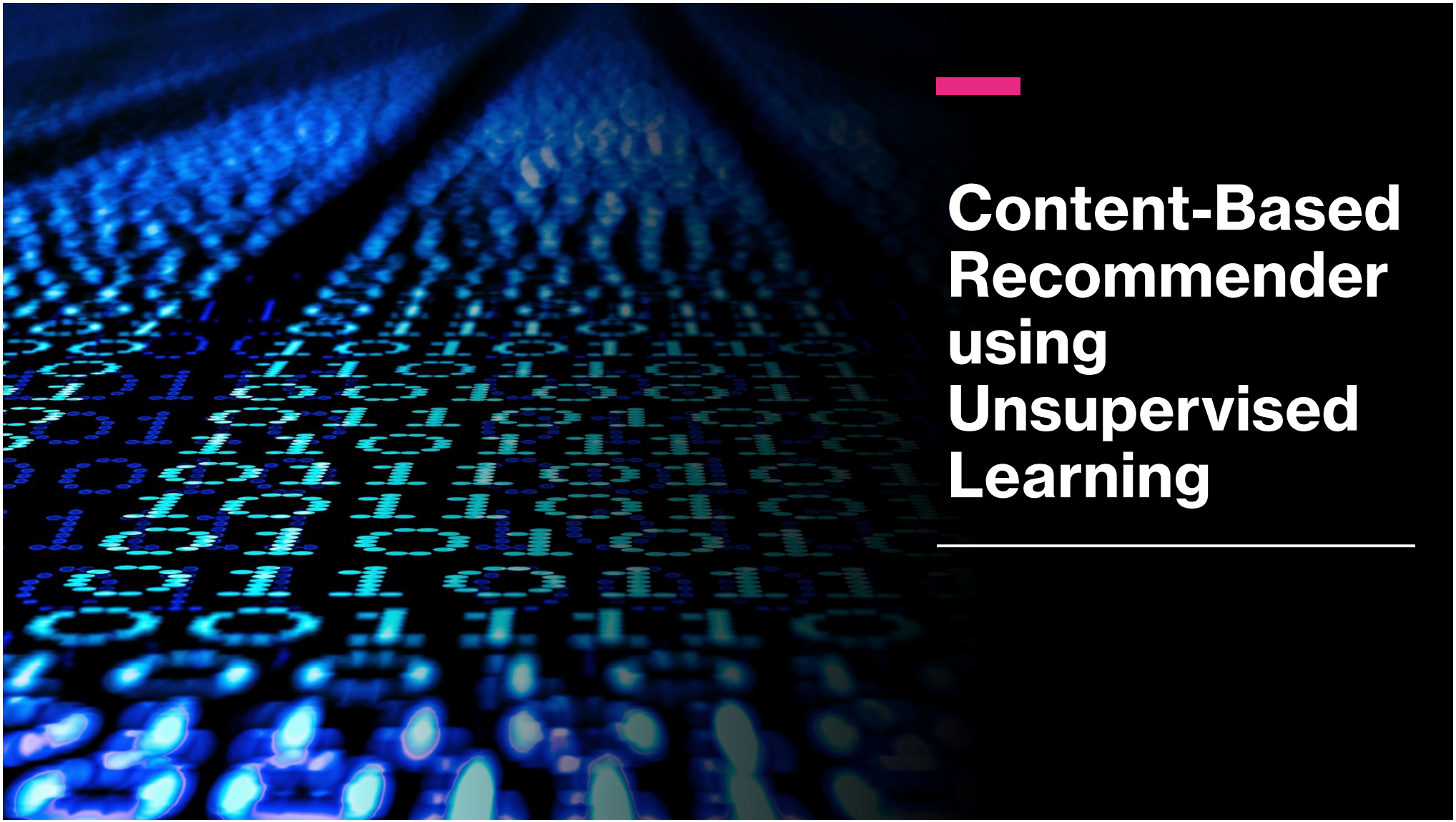
	COURSE_ID	Ratings	TITLE
0	PY0101EN	14936	python for data science
1	DS0101EN	14477	introduction to data science
2	BD0101EN	13291	big data 101
3	BD0111EN	10599	hadoop 101
4	DA0101EN	8303	data analysis with python
5	DS0103EN	7719	data science methodology
6	ML0101ENv3	7644	machine learning with python
7	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

Word Cloud of Course Titles

The utilization of WordCloud visualization unveils prevalent keywords within a dataset of online course titles. By aggregating and filtering course titles, dominant themes such as python, data science, machine learning, big data, AI, spark, deep learning, and cloud computing are identified.

This analysis provides valuable insights into the trending IT skills and subject areas covered by online courses, helping to guide learners and educators in understanding the current landscape of digital learning opportunities.





Content-Based Recommender 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 preferences. For the recommender system, the raw data includes user profiles, course genres and course ratings/interactions
2. **Data Processing** – This step is where the data is cleaned and preprocessed to prepare it for analysis. Tasks such as handling missing values, removal of duplicates and data transformation were performed so that the data was in a suitable format for further analysis
3. **Cleaned Dataset** – After processing, data is ready for feature engineering
4. **Feature Engineering** - 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** - 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:

For our user profile-based recommender system, a score threshold of 10.0 was used to filter out low-scoring recommendations. This threshold determines which courses are considered relevant enough to be recommended to users.

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. Here, the average number was approximately 60.82 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 USER column. These recommendations are generated by analysing user profiles and course genre vectors, with courses scoring higher in relevance to being recommended more frequently. The table shows that course 'TA0106EN' is the most frequently recommended course, with 17,390 recommendations.

```
# Calculate the average number of recommended courses per test user
average_courses_per_user = res_df.groupby('USER')['COURSE_ID'].nunique().mean()

# Print the result
print("Average number of new courses recommended per test user:", average_courses_per_user)
```

```
Average number of new courses recommended per test user: 60.82471217772012
```

COURSE_ID	USER
TA0106EN	17390
excourse21	15656
excourse22	15656
GPXX0IBEN	15644
ML0122EN	15603
excourse04	15062
excourse06	15062
GPXX0TY1EN	14689
excourse72	14464
excourse73	14464

Flowchart of Content-Based Recommender System Using Course Similarity

1. **Raw Data** – This refers to the initial dataset containing information about the courses, with data such as title, description etc.
2. **Data Processing** – Involves preprocessing raw data, which includes tokenization and lemmatization to break down text into individual words and convert them to their base form.
3. **Cleaned Dataset** – Once processed, the dataset is cleaned by removing stopwords – commonly used words that have little semantic value – and outliers (data points that provide a lot of 'noise')
4. **Feature Engineering** – transform the cleaned dataset into numerical features that represent the courses. Here Term Frequency-Inverse Document Frequency (TF-IDF) vectors are calculated for each course based on the words they contain as well as their importance in the dataset
5. **Features** – The final set of features used to represent each course, using the TF-IDF vectors from the feature engineering step. These features are used to calculate the similarities between courses and generate recommendations based on the similarity of the content.

Evaluation Results of Course Similarity Based Recommender System

Hyper-parameter Settings:

The similarity threshold used for the course similarity-based recommender system was set to 0.6. This threshold determines the level of similarity required between courses similarity-based 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 dataset was approximately 8.55 courses per user. This metric provides insight into the diversity of recommendations provided to users and helps to assess the system's effectiveness in its content suggestions.

Top-10 Most Frequently Recommended Courses:

The table represents the top 10 most frequently recommended courses. From looking at the table, we see that 'DS0110EN' is the most recommended course with 15,003 recommendations, followed by 'excourse22' & 'excourse62' which both had 14,937 recommendations. These results give a valuable insight into the performance of the course similarity-based recommender system, as well as helping to identify possible improvements for future iterations.

0	1
DS0110EN	15003
excourse22	14937
excourse62	14937
excourse63	14641
excourse65	14641
excourse68	13551
excourse72	13512
excourse74	13291
excourse67	13291
BD0145EN	12497

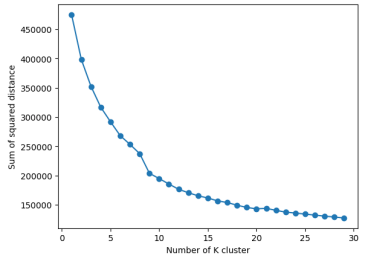
Flowchart of Clustering Based Recommender System

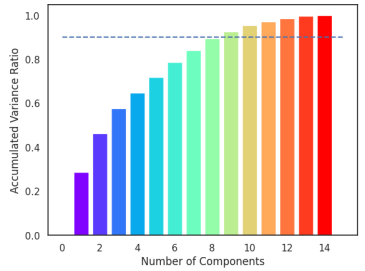
1. **Raw Data** – This refers to the original user profile vector features, which contained information about users' interests or preferences across different genres.
2. **Data Processing** – Raw data is preprocessed, to handle missing values, outliers or other data quality issues. We apply a normalization technique called StandardScaler to ensure that all features have a similar scale and distribution. This is crucial when using a clustering algorithm as without it, the model will not be able to perform effectively.
3. **Cleaned Dataset** – After processing, a cleaned dataset is created where the user profile features are standardized having been normalized using StandardScaler. Each feature has a mean of 0 and a standard deviation of 1, which is a common requirement for clustering algorithms.
4. **Feature Engineering** - Transforming the original user profile features into a new set of features that capture the most important information while reducing dimensionality. Principal Component Analysis (PCA) identifies the principal components that explain the maximum variance in the data and projects the original features onto these components.
5. **Features** - The final output of the process is the transformed feature set obtained after applying PCA. These features represent a lower-dimensional representation of the original user profile data, where each feature captures a combination of the original features' information.

Evaluation Results of Clustering Based Recommender System

Hyper-parameter Settings:

Employing the K-Means algorithm, we determined the ideal number of clusters using the elbow method. The top graph shows the ideal number of clusters for the recommender is between 10-15 clusters. For PCA, we selected the number of components that explained over 90% of the variance in the data. These parameters helped to ensure that the recommender system could group users based on their preferences whilst minimizing information loss through dimensionality reduction.





Average Number of New Courses Recommended per User:

The average number of new or unseen courses recommended per user in the test dataset was approximately 37.65 courses per user. This metric provides insight into the diversity of recommendations provided to users and helps to assess the system's effectiveness in its content suggestions.

Top-10 Most Frequently Recommended Courses:

The table represents the top 10 most frequently recommended courses. From looking at the table, we see that 'WA0101EN' is the most recommended course with 31,635 recommendations, followed by 'SC0101EN' with 31,162 & 'CL0101EN' which had 30,883 recommendations.

	0	1
45	WA0101EN	31635
38	SC0101EN	31162
15	CL0101EN	30883
71	DS0301EN	30258
1	BD0115EN	30215
52	DB0101EN	30189
50	CO0101EN	29408
28	ML0101EN	29138
64	CC0101EN	28906
41	ST0101EN	28869

Collaborative Filtering Recommender System using Supervised Learning

Flowchart of KNN Based Recommender System

1. **Raw Data** – This refers to the original dataset containing information about user-item interactions such as user ID, item ID (the courses) and ratings (the enrolments). Raw data consists of user-item pairs along with their corresponding ratings
2. **Data Processing** – Dataset is loaded, tasks such as handling missing values, removal of duplicates and converting data into a suitable format are performed.
3. **Cleaned Dataset** – After processing, a cleaned dataset is created which has had irrelevant or missing data removed.
4. **Feature Engineering** – Creating new features and transforming existing features to improve the performance of the recommendation system. Relevant information may be extracted at this stage, such as user demographics. Features may also involve encoding categorical variables, scaling numerical features or creating interaction terms
5. **Features** – The variables or attributes used by the KNN-based recommender system to make predictions. These features capture the relationship between the user and items, enabling the system to identify similarities and make personalised recommendations.

Flowchart of NMF Based Recommender System

1. **Raw Data** – This refers to the course ratings data.
2. **Data Processing** – Dataset is loaded, tasks such as handling missing values, removal of duplicates and converting data into a suitable format are performed.
3. **Cleaned Dataset** – After processing, a cleaned dataset is created which has had irrelevant or missing data removed.
4. **Feature Engineering** - Creating new features and transforming existing features to improve the performance of the recommendation system. Features may also include user-item interactions or latent factors generated by the NMF model.
5. **Features** - The variables or characteristics used by the machine learning model to make predictions. In the context of collaborative filtering, features might include user preferences, item attributes, or latent factors representing users and items in a lower-dimensional space.

Flowchart of Neural Network Embedding Based Recommender System

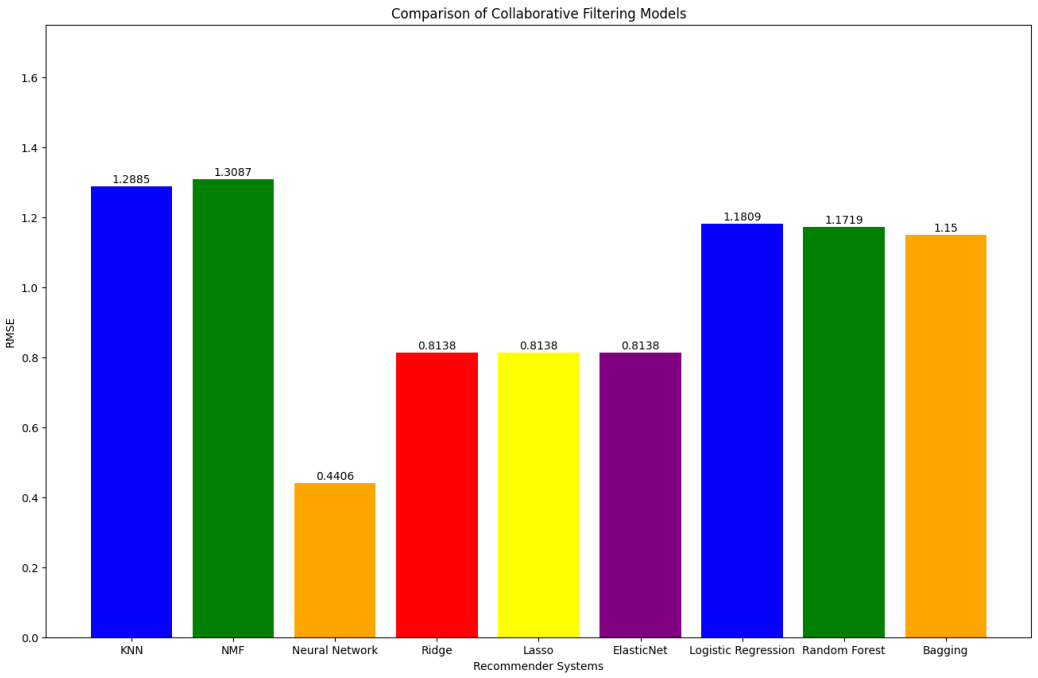
1. **Raw Data** – This refers to the course ratings data.
2. **Data Processing** – Dataset is loaded, tasks such as handling missing values, removal of duplicates and converting data into a suitable format are performed.
3. **Cleaned Dataset** – After processing, a cleaned dataset is created which has had irrelevant or missing data removed.
4. **Feature Engineering** - Creating new features and transforming existing features to improve the performance of the recommendation system. Features may also include user-item interactions or latent factors generated by the Neural Network Embedding model.
5. **Features** - The variables or characteristics used by the machine learning model to make predictions. In the context of collaborative filtering, features might include user preferences, item attributes, or latent factors representing users and items in a lower-dimensional space.

Compare the Performance of Collaborative Filtering Models

The bar chart on the slide shows the performance of the following collaborative filtering models:

- KNN
- NMF
- Neural Network
- Ridge
- Lasso
- ElasticNet
- Logistic Regression
- Random Forest
- Bagging

Based on the chart, the Neural Network Embedding based recommender system achieved the lowest root mean square error value of 0.4406, indicating the best performance across all the models in predicting user-item interactions. Therefore, this would be the model best suited for collaborative filtering in this scenario.



Conclusions

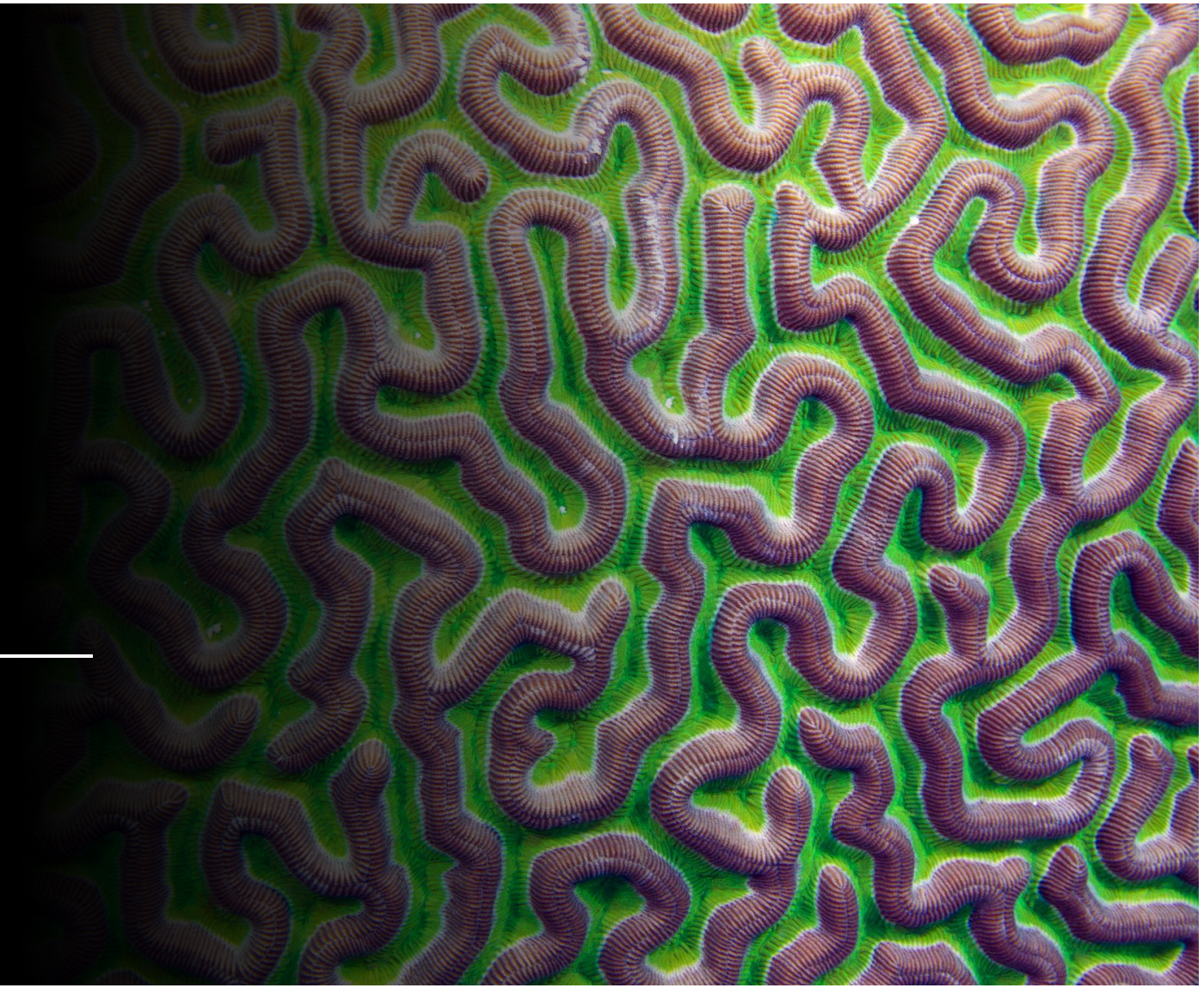
In this project, various machine learning techniques were implemented to enhance course recommendation systems. Starting with preprocessing techniques like Bag of Words (BoW) and stopword removal, we transformed raw textual data into numerical features for analysis.

Using cosine similarity, we recommended similar courses based on user enrolment data. We further enhanced the recommendations by applying clustering algorithms such as K-means to group users with similar learning interests, enabling personalized suggestions.

Collaborative filtering methods, such as K-Nearest Neighbour and NMF, were used to predict user-course interactions. These models were evaluated using root mean squared error (RMSE) to assess prediction accuracy. Neural networks, trained using TensorFlow, extracted latent features of users and courses, further refining the rating predictions. Moreover, we extended our models by building regression and classification methods, leveraging embedding vectors to predict course ratings. The regression models used regularization techniques (L1, L2, and ElasticNet), while classification models incorporated methods like Logistic Regression, Random Forest and Bagging. Upon evaluation of these models, we saw that the Neural Network Embedding based recommender system was the best performing model, as it's root mean squared error was lower than the other models tested. Therefore, this model would be the best suited for collaborated filtering for a user recommendation scenario.

Overall, the combination of traditional collaborative filtering techniques, deep learning models, and advanced regression/classification methods provided an effective, multi-layered approach to recommending personalized courses based on user preferences.

Appendix



Github: https://github.com/bcottrell93/ibm_machine_learning/tree/main/machine_learning_capstone