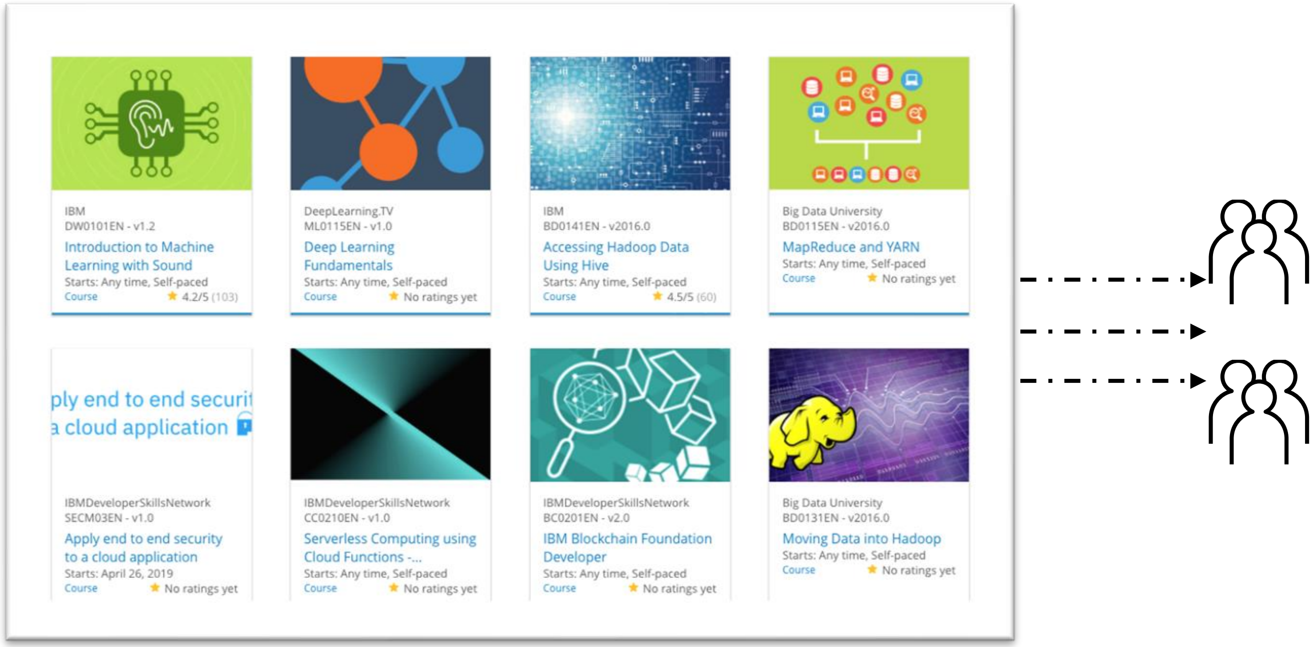
Build a Personalized Online Course Recommender System with Machine Learning

CHEE WEI HAN
6TH AUGUST 2024



IBM
DW0101EN - v1.2
Introduction to Machine Learning with Sound
Starts: Any time, Self-paced
Course ★ 4.2/5 (103)

DeepLearning.TV
ML0115EN - v1.0
Deep Learning Fundamentals
Starts: Any time, Self-paced
Course ★ No ratings yet

IBM
BD0141EN - v2016.0
Accessing Hadoop Data Using Hive
Starts: Any time, Self-paced
Course ★ 4.5/5 (60)

Big Data University
BD0115EN - v2016.0
MapReduce and YARN
Starts: Any time, Self-paced
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
SECM03EN - v1.0
Apply end to end security to a cloud application
Starts: April 26, 2019
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
CC0210EN - v1.0
Serverless Computing using Cloud Functions -...
Starts: Any time, Self-paced
Course ★ No ratings yet

IBMDeveloperSkillsNetwork
BC0201EN - v2.0
IBM Blockchain Foundation Developer
Starts: Any time, Self-paced
Course ★ No ratings yet

Big Data University
BD0131EN - v2016.0
Moving Data into Hadoop
Starts: Any time, Self-paced
Course ★ No ratings yet

Outline

- Introduction and Background
- Exploratory Data Analysis
- Content-based Recommender System using Unsupervised Learning
- Collaborative-filtering based Recommender System using Supervised learning
- Conclusion
- Appendix

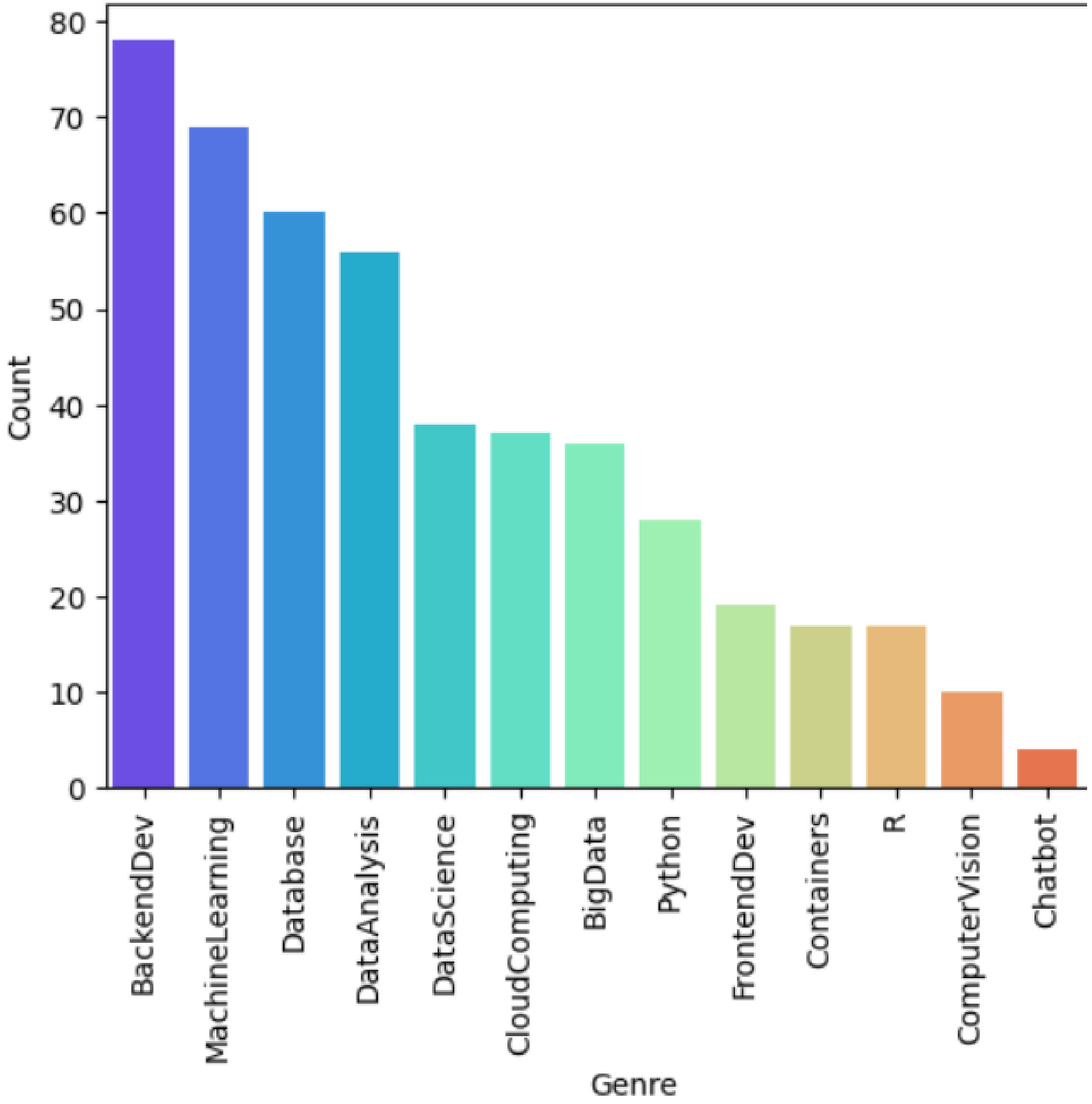
Introduction

- This project aims to build a courses recommender system to our users based on their preference and their neighbor action.
- We found that our user can be categorized to some group and recommend similar courses that belongs to the same groups. By doing this, they can enroll the new course and increase their satisfaction. Also, it end up increasing the revenue of our company.

Exploratory Data Analysis

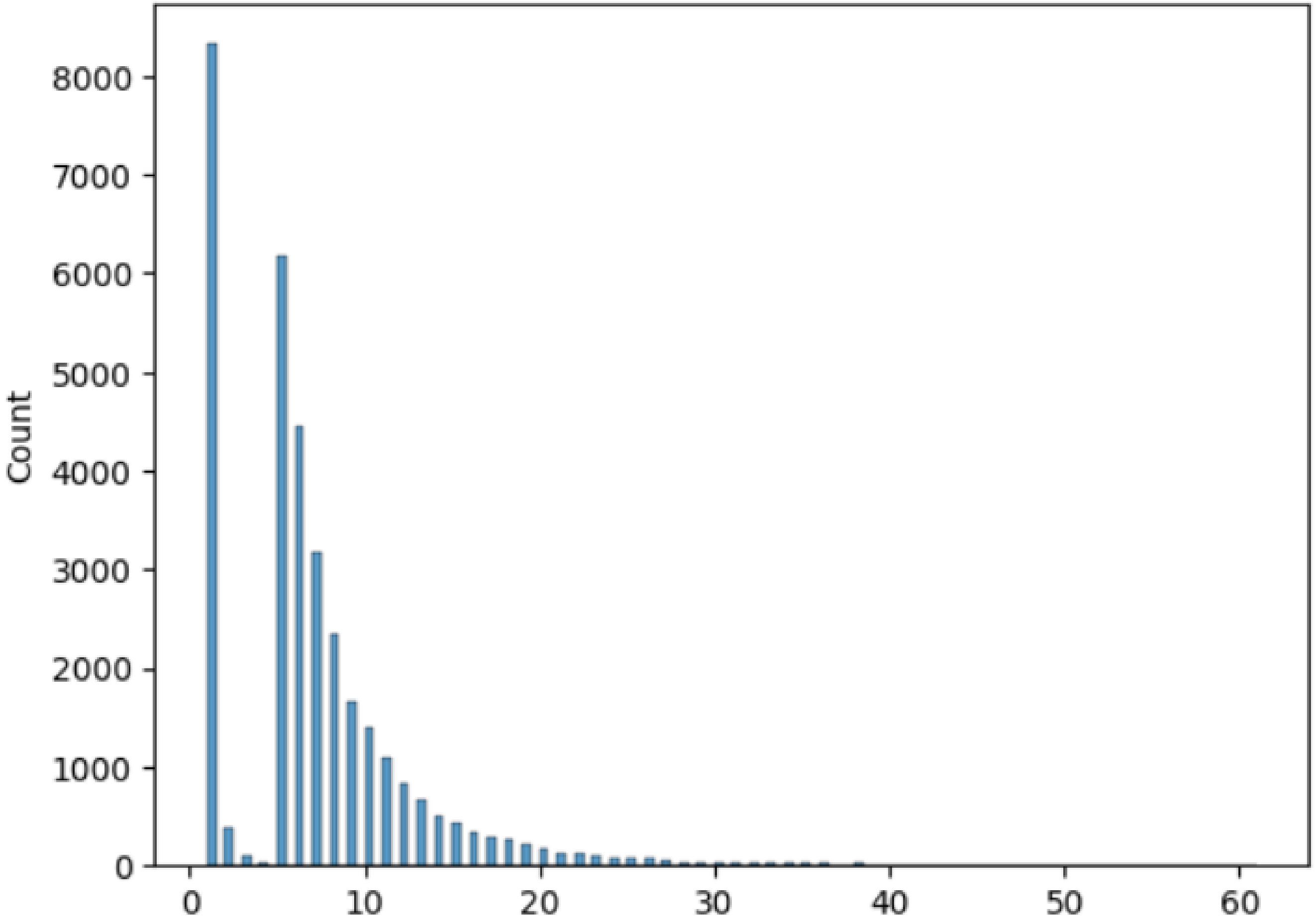
Course counts per genre

- Firstly, we would like to see the number of genre in courses.
- We used barchart to show the count of genre from all the courses.
- The most genre in the courses is backendDEV, machine learning and database.



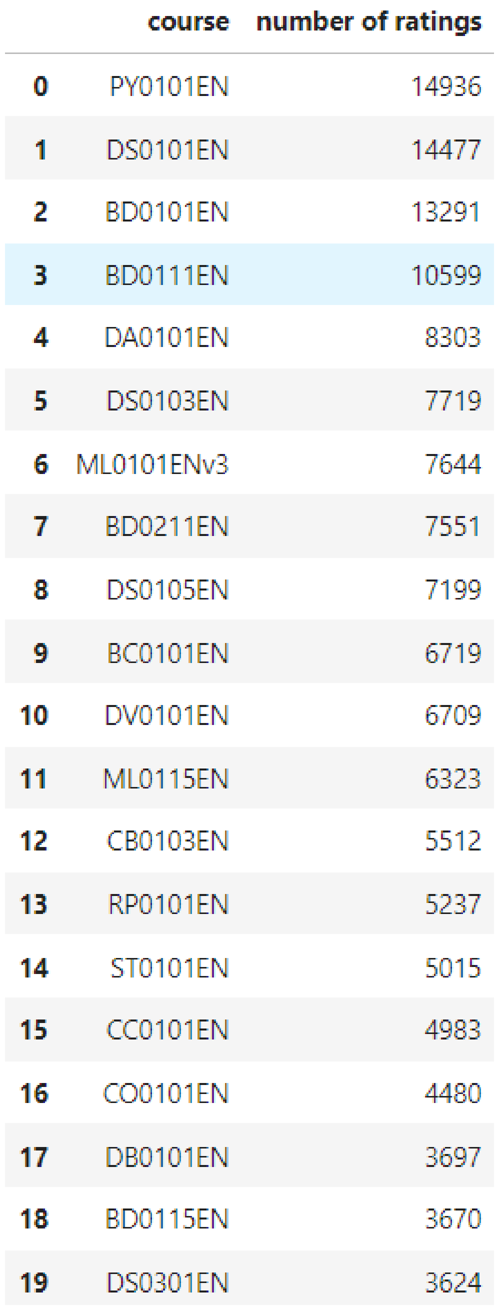
Course enrollment distribution

- We can also get a histogram showing the enrollment distributions, e.g., how many users rated just 1 item or how many rated 10 items, etc.
- Hence, we can see that, the highest number of enrollments is belongs to 0 to 10.

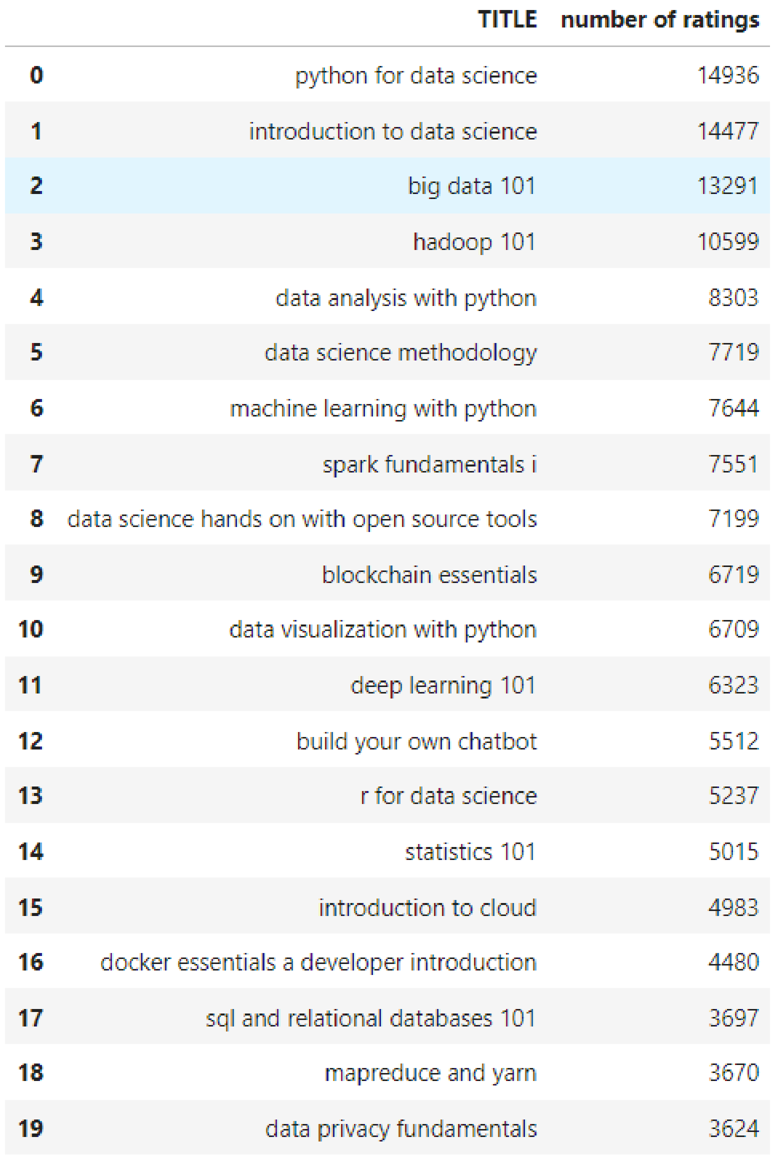


20 most popular courses

- The most popular 20 courses are here
- As the dataframe here, we notice that PY0101EN (python for data science) has the highest rating or enrollments. Then, the second highest will be 'DS0101EN' (introduction to data science) course.
- From here, we know that the courses related to the data science is popular current moment.



	course	number of ratings
0	PY0101EN	14936
1	DS0101EN	14477
2	BD0101EN	13291
3	BD0111EN	10599
4	DA0101EN	8303
5	DS0103EN	7719
6	ML0101ENv3	7644
7	BD0211EN	7551
8	DS0105EN	7199
9	BC0101EN	6719
10	DV0101EN	6709
11	ML0115EN	6323
12	CB0103EN	5512
13	RP0101EN	5237
14	ST0101EN	5015
15	CC0101EN	4983
16	CO0101EN	4480
17	DB0101EN	3697
18	BD0115EN	3670
19	DS0301EN	3624

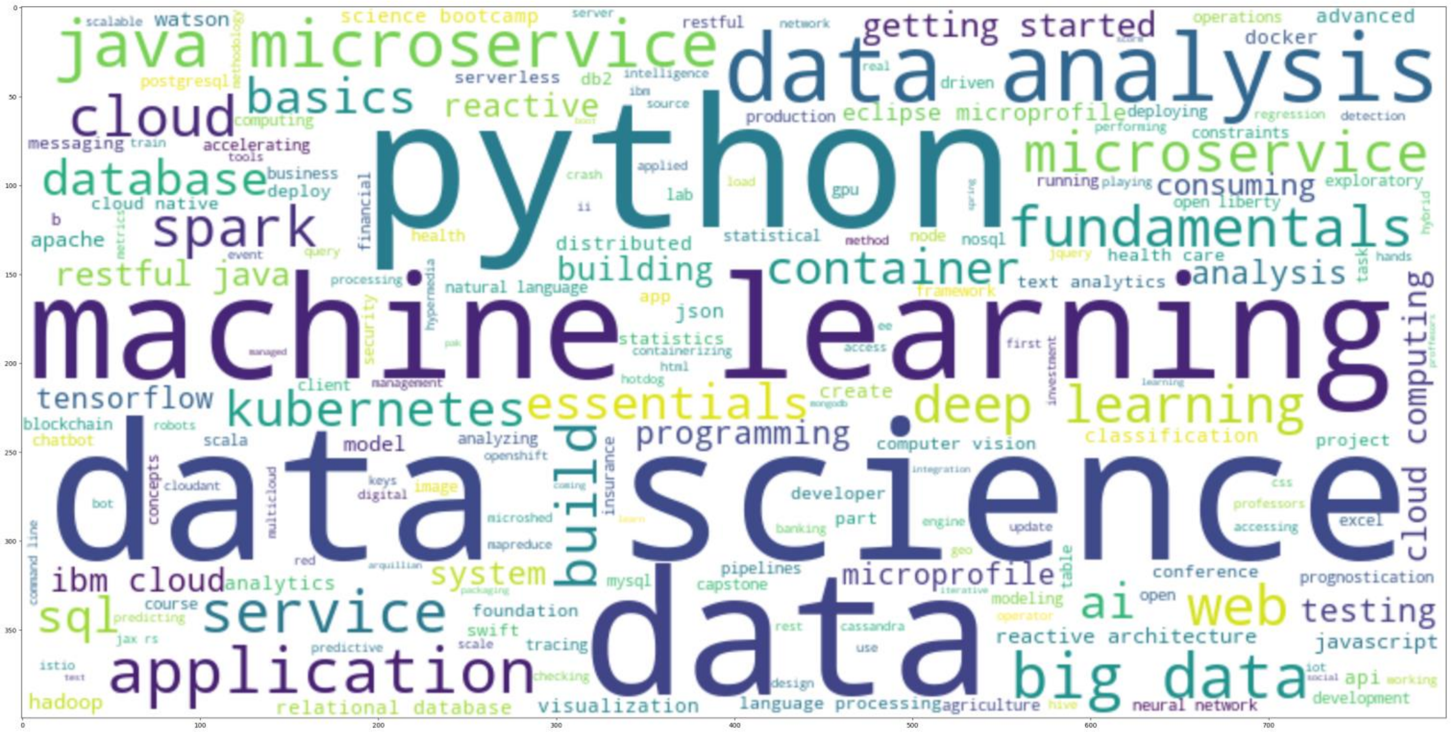


	TITLE	number of ratings
0	python for data science	14936
1	introduction to data science	14477
2	big data 101	13291
3	hadoop 101	10599
4	data analysis with python	8303
5	data science methodology	7719
6	machine learning with python	7644
7	spark fundamentals i	7551
8	data science hands on with open source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

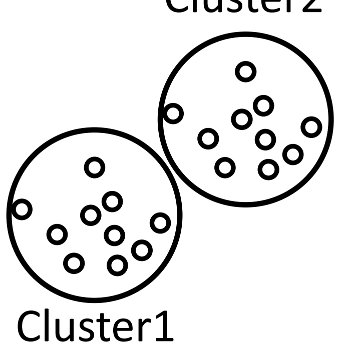
Word cloud of course titles

- We created WordCloud to visualize the dominant topic in the courses.

- As shown, the popular topics are python, machine learning and data science

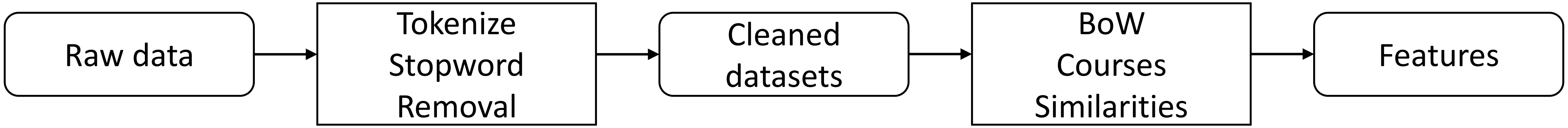


Content-based Recommender System using Unsupervised Learning



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

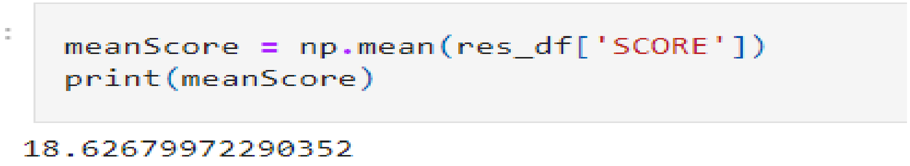
- Firstly, we can the raw textual data based on the tile and description of courses and preprocessed them into numeric value for machine learning to capture the pattern. From here, we used Bags of Words (BoW) method to calculate the frequency of each unique words in each courses. Then, we remove stopwords to reduce the dimensionality as these words are not provided much impact to our system.
- Then, we used this BoW to calculate the similarities of each courses using cosine distance to get the similarities score of each courses. We used this into our feature.
- Then, we iterate each user/enrolled course to finds their similar courses and try to recommend them similar courses based on the similarity score.



Evaluation results of user profile-based recommender system

We set our threshold score for recommendation is 10

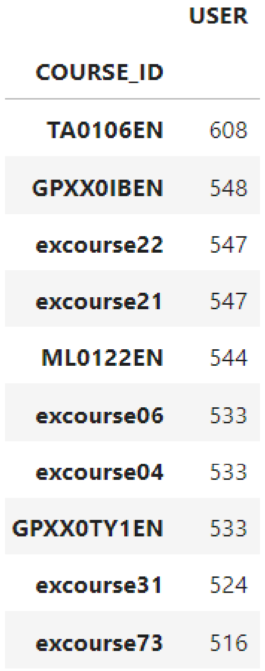
On average, there are around 18courses are recommended to users



```
meanScore = np.mean(res_df['SCORE'])
print(meanScore)
```

```
18.62679972290352
```

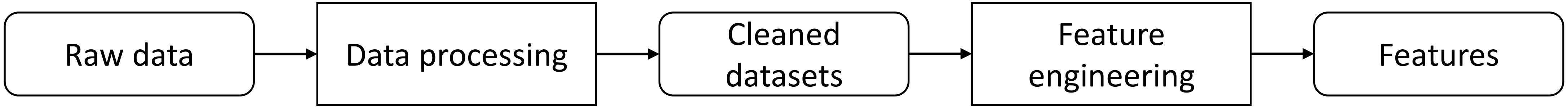
These list of course is the 10 most hot courses recommend across users.



	USER
COURSE_ID	
TA0106EN	608
GPXX0IBEN	548
excourse22	547
excourse21	547
ML0122EN	544
excourse06	533
excourse04	533
GPXX0TY1EN	533
excourse31	524
excourse73	516

Flowchart of content-based recommender system using course similarity

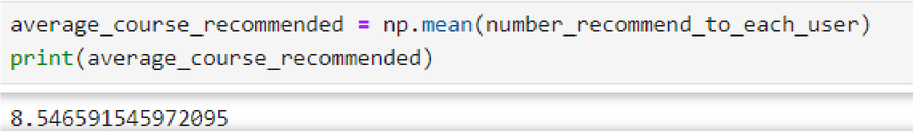
- We then applied the course similarities metric to recommend new courses which are similar to a user's presently enrolled courses.
- Firstly, we get BoW and assign the index to the courses for query purpose. With their course ids, we can use the id_idx_dict dictionary to query their row and column index on the similarity matrix.
- The we find courses which are similar enough to user enrolled courses.



Evaluation results of course similarity based recommender system

If the similarity is larger than a threshold such as 0.5 or 0.6, then add it to user course recommendation list

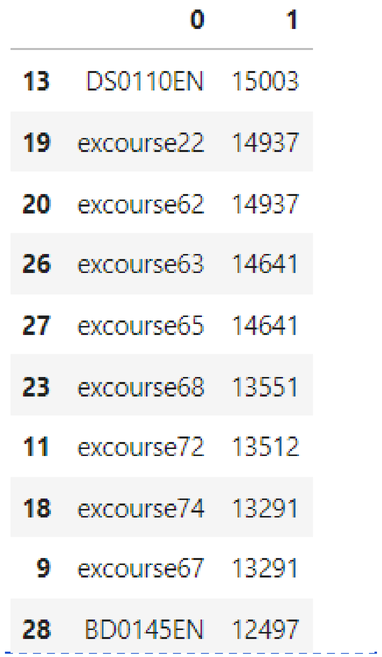
On average, there are around 8 to 9 courses are recommended to users



```
average_course_recommended = np.mean(number_recommend_to_each_user)
print(average_course_recommended)
```

```
8.546591545972095
```

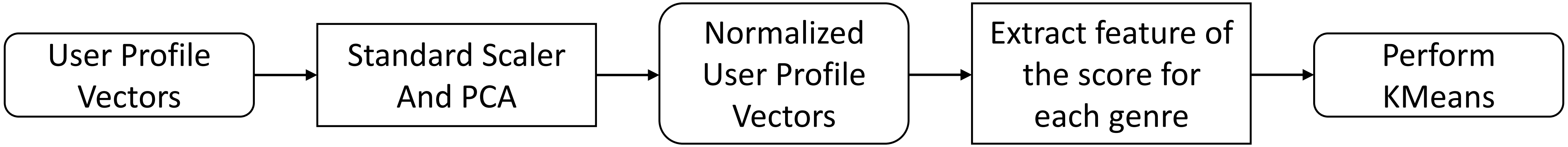
These list of course is the 10 most hot courses recommend across users.



	0	1
13	DS0110EN	15003
19	excourse22	14937
20	excourse62	14937
26	excourse63	14641
27	excourse65	14641
23	excourse68	13551
11	excourse72	13512
18	excourse74	13291
9	excourse67	13291
28	BD0145EN	12497

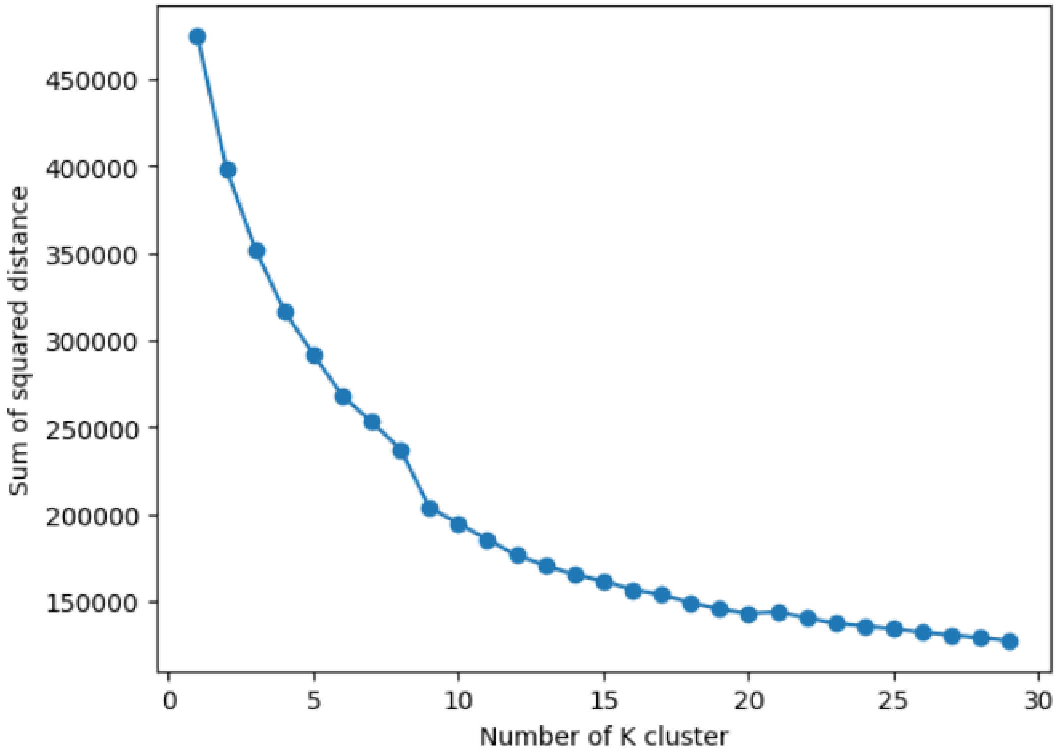
Flowchart of clustering-based recommender system

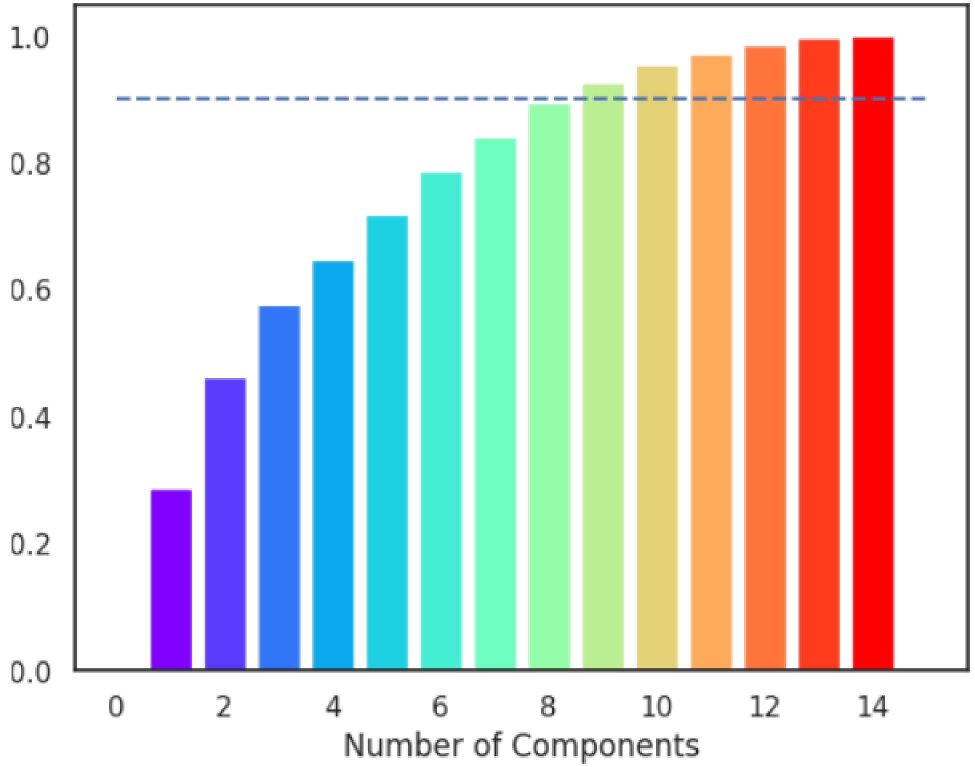
- With the user profile vectors generated, we can also easily compute the similarity among users based on their shared interests. We performed clustering algorithms such as K-means to group users with similar learning interests. For each user group, we can come up with a list of popular courses and recommend to the similar users.
- The dataset used is the user profile vectors that contains the scores of each genre for each user. We normalized the values and using PCA to perform the dimensionality reduction with new principal component that represent the features. Then, the new components are fit into KMeans cluster algorithm to find how many cluster can be formed.


User Profile Vectors
Standard Scaler And PCA
Normalized User Profile Vectors
Extract feature of the score for each genre
Perform KMeans

Result of clustering-based recommender system

We found that the 8 clusters and 9 principal component performed the best for clustering.

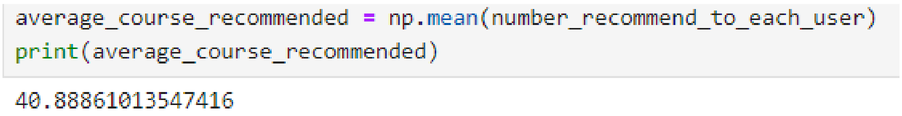




Evaluation results of clustering-based recommender system

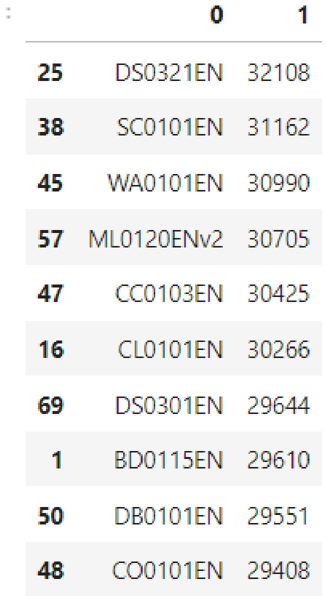
We found that the 8 clusters and 9 principal component performed the best for clustering.

On average, there are around 40 courses are recommended to users



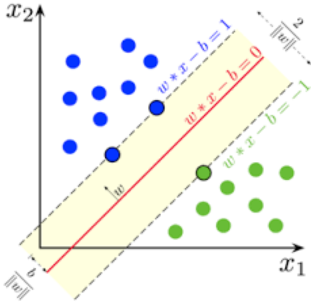
```
average_course_recommended = np.mean(number_recommend_to_each_user)
print(average_course_recommended)

40.88861013547416
```

These list of course is the 10 most hot courses recommend across users.



	0	1
25	DS0321EN	32108
38	SC0101EN	31162
45	WA0101EN	30990
57	ML0120ENv2	30705
47	CC0103EN	30425
16	CL0101EN	30266
69	DS0301EN	29644
1	BD0115EN	29610
50	DB0101EN	29551
48	CO0101EN	29408

Collaborative-filtering Recommender System using Supervised Learning

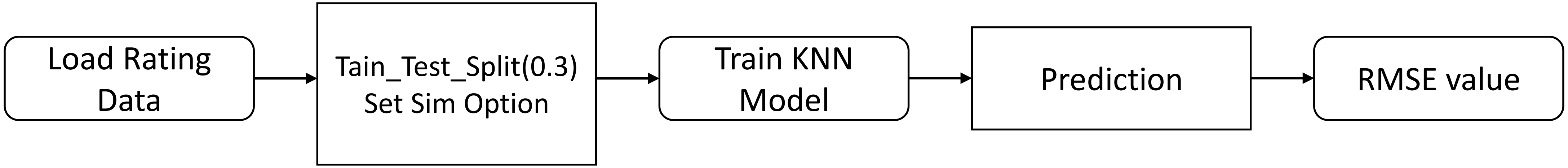


Flowchart of KNN based recommender system

- We performed KNN-based collaborative filtering on the user-item interaction matrix.

- We used surprise library to auto load the rating_df and train the KNN model. Then, we calculate the root mean squared error on the predictions of the model on testset data.

- After splitting the user course rating data, the KNN classification model is trained with the trainset. Predictions are made using the testset and the rmse will be evaluated.



Load Rating Data

Tain_Test_Split(0.3)
Set Sim Option

Train KNN Model

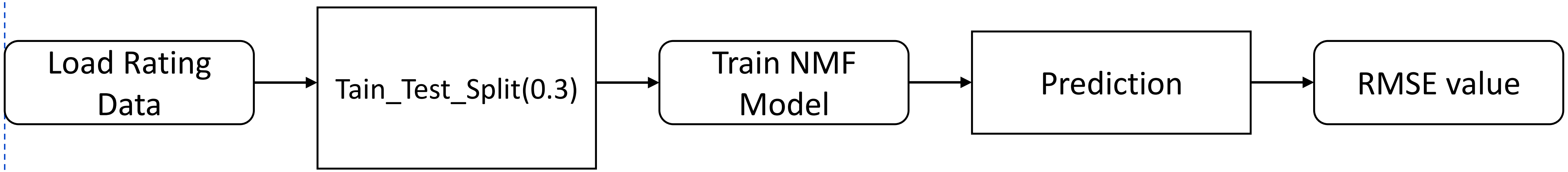
Prediction

RMSE value

Flowchart of NMF based recommender system

- We performed NMF-based collaborative filtering on the user-item matrix, which decomposes a big sparse matrix into two smaller and dense matrices.

- After train test splitting the user course rating data with 0.3 ratio, the NMF classification model is trained with the trainset. Predictions are made using the testset and the rmse will be evaluated.



Load Rating Data

Tain_Test_Split(0.3)

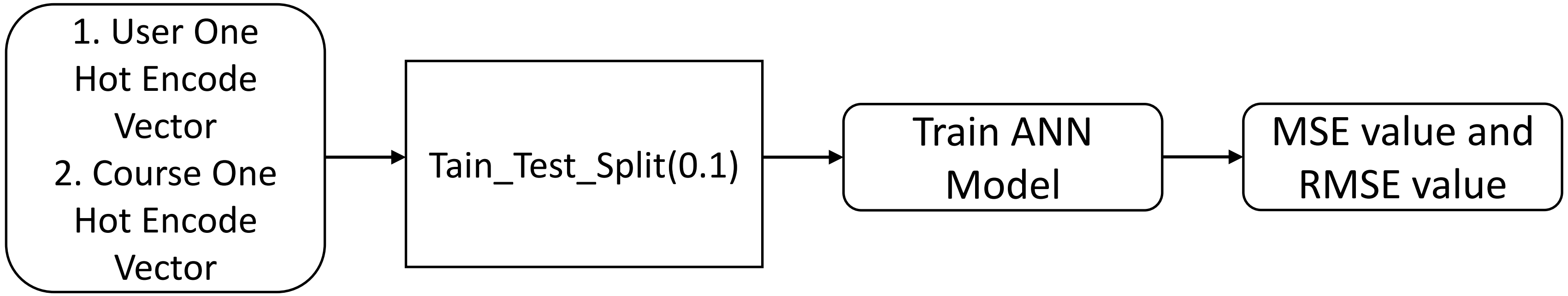
Train NMF Model

Prediction

RMSE value

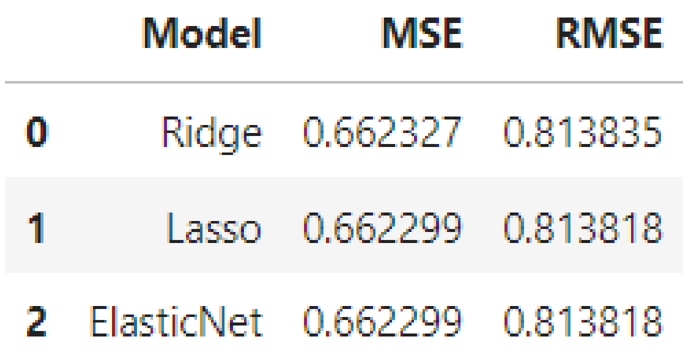
Flowchart of Neural Network Embedding based recommender system

- We then use tensorflow to train neural networks to extract the user and item latent features from the hidden's layers
- Predict course ratings with trained neural networks
- Neural networks can also be used to extract the latent user and item features. ANN performs rating prediction using dot product obtained through user one hot encoding and course one hot encoding, then dot product will be activated using relu.

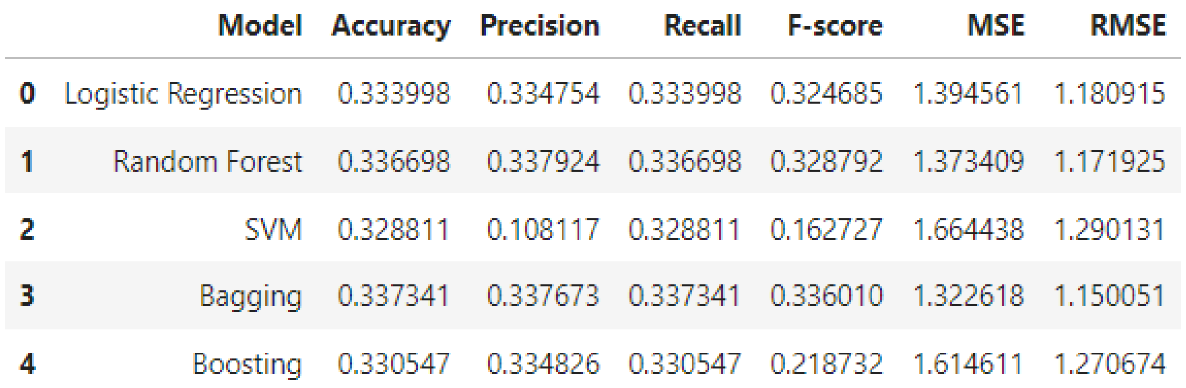

1. User One Hot Encode Vector
2. Course One Hot Encode Vector
Tain_Test_Split(0.1)
Train ANN Model
MSE value and RMSE value

Compare the performance of collaborative-filtering models

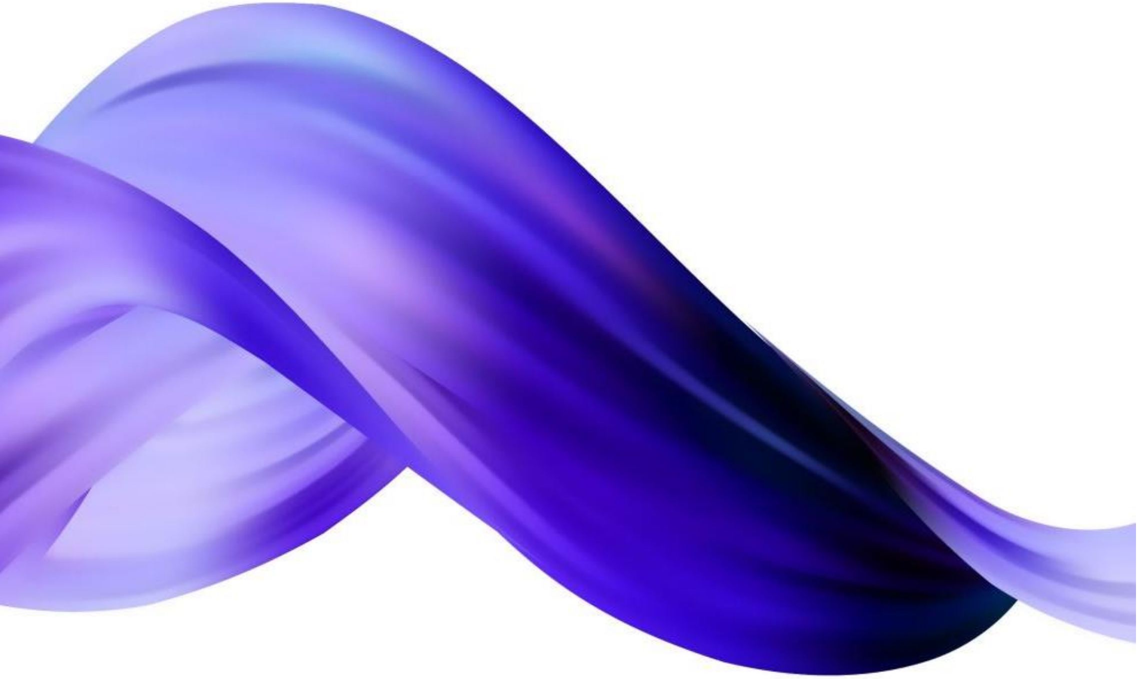
- Besides, we also build regression and classification method as extends ANN model by using two embedding vectors as an input into a Neural Network to predict the rating.
- We build regression and classification models to predict ratings using the combined embedding vectors.
- The regression model we used L1, L2 and ElasticNet regularization, while classification we used Logistic Regression, Random Forest, SVM, Bagging model and Boosting Model.
- We train and test the combined embedding vector and predict the score.



	Model	MSE	RMSE
0	Ridge	0.662327	0.813835
1	Lasso	0.662299	0.813818
2	ElasticNet	0.662299	0.813818



	Model	Accuracy	Precision	Recall	F-score	MSE	RMSE
0	Logistic Regression	0.333998	0.334754	0.333998	0.324685	1.394561	1.180915
1	Random Forest	0.336698	0.337924	0.336698	0.328792	1.373409	1.171925
2	SVM	0.328811	0.108117	0.328811	0.162727	1.664438	1.290131
3	Bagging	0.337341	0.337673	0.337341	0.336010	1.322618	1.150051
4	Boosting	0.330547	0.334826	0.330547	0.218732	1.614611	1.270674



Conclusions

- We have successful to build a courses recommender system from EDA and using different models to find the best model to improve our system to our users.
- To build a recommender system, there are 2 approach can be used which are content based and collaborative-filtering (User-based and item-based). Both approach is important to improve the system and recommend the course to our user accurately and also increase their satisfaction.

Appendix

- GitHub: https://github.com/WhanIsHere/IBM-Machine-Learning-Capstone