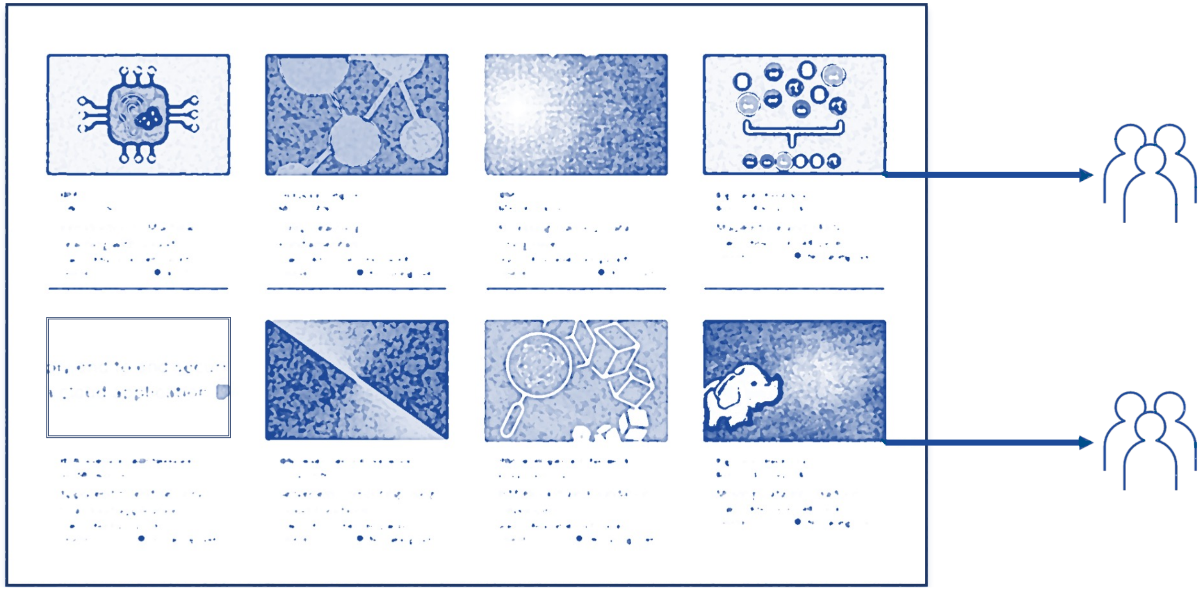
Building a Personalised Online Course Recommender System with **Machine Learning** Technologies

Dan Stollenwerk
24/1/24



Outline

- Introduction
- Exploratory Data Analysis
- **Content-Based** Recommender System
  > *Unsupervised Learning*
- **Collaborative Filtering-Based** Recommender System
  > *Supervised Learning*
- Observations
- Appendix

Introduction

- An **online course recommender system** enhances personalised learning experiences by suggesting relevant courses based on **user preferences** as reflected by past **activity**
  - Promotes continuous **skill development**
  - Improves **user reach** of educational resources

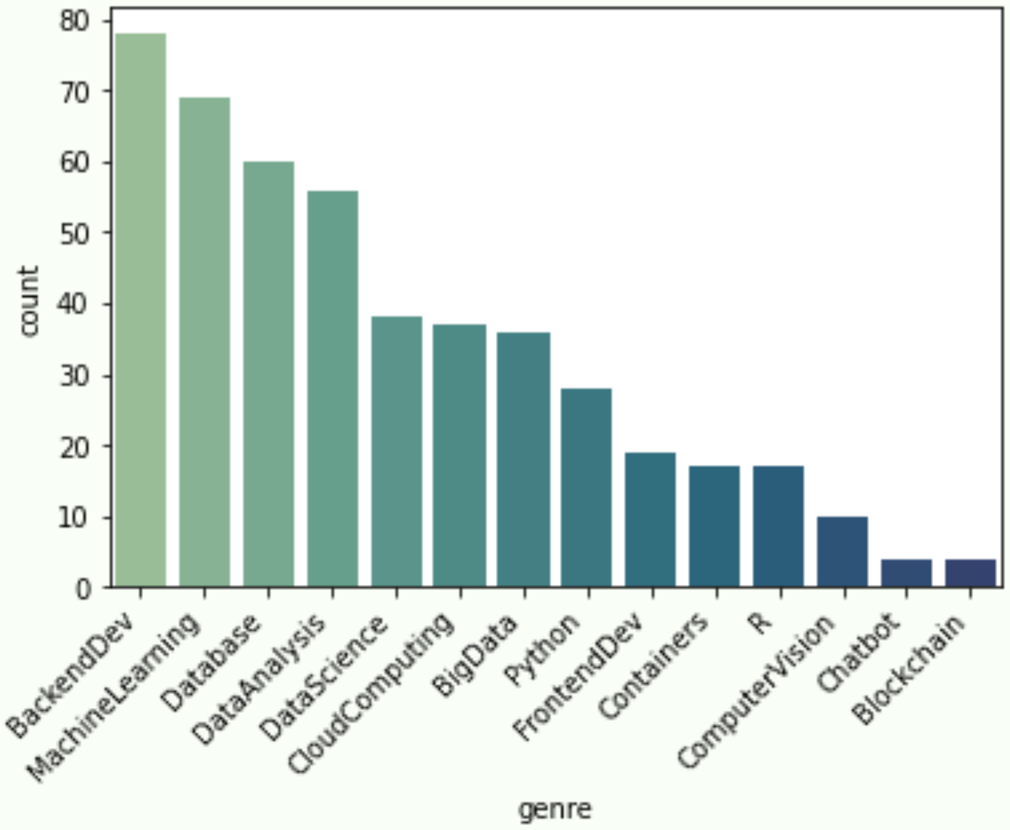
- **Problem**: test **machine learning** technologies using online course data to build optimised recommender system

- **Hypothesis**: implementation of recommender system will improve **completion rate** of online courses

Exploratory Data Analysis

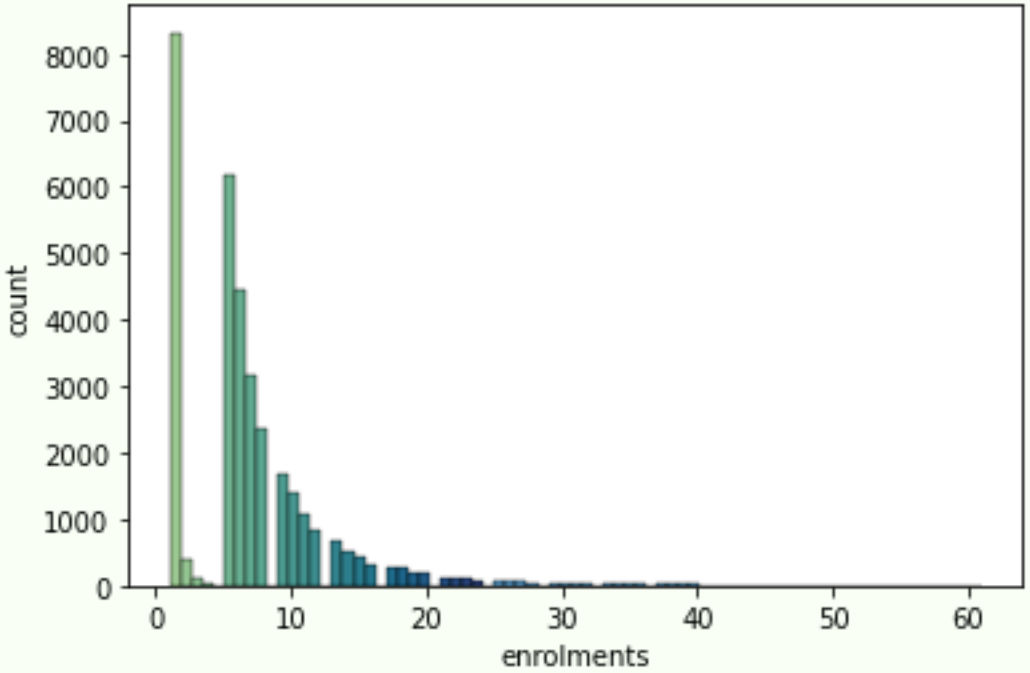
Course **Genre** Distribution

- **BackendDev** most common genre with **78** relevant courses
- **Chatbot** and **Blockchain** least common genres with only **4** relevant courses



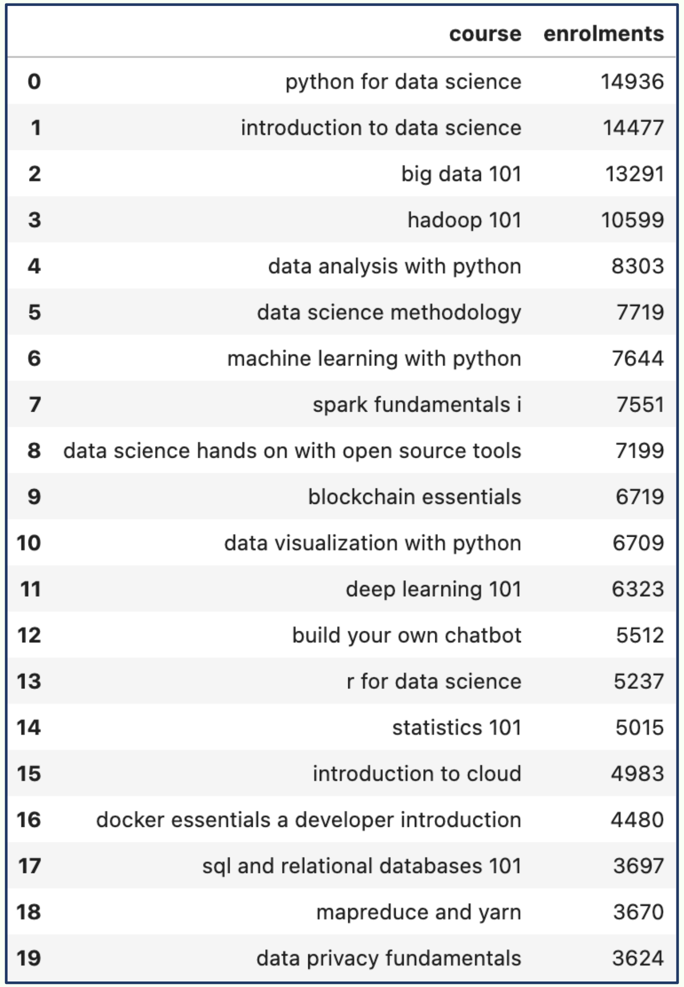
Course Enrolments Distribution

- Largest group of users (8,320 / 33,901: 24.5%) enrolled in only **1 course**
- Next-largest group of users (6,179 / 33,901: 18.2%) enrolled in **5 courses**
- Data observes **exponential decay**
- Some enrolment numbers held by **~0** users
    - These numbers appear at **regular intervals** of 4 bars – *why?*



Top 20 Courses

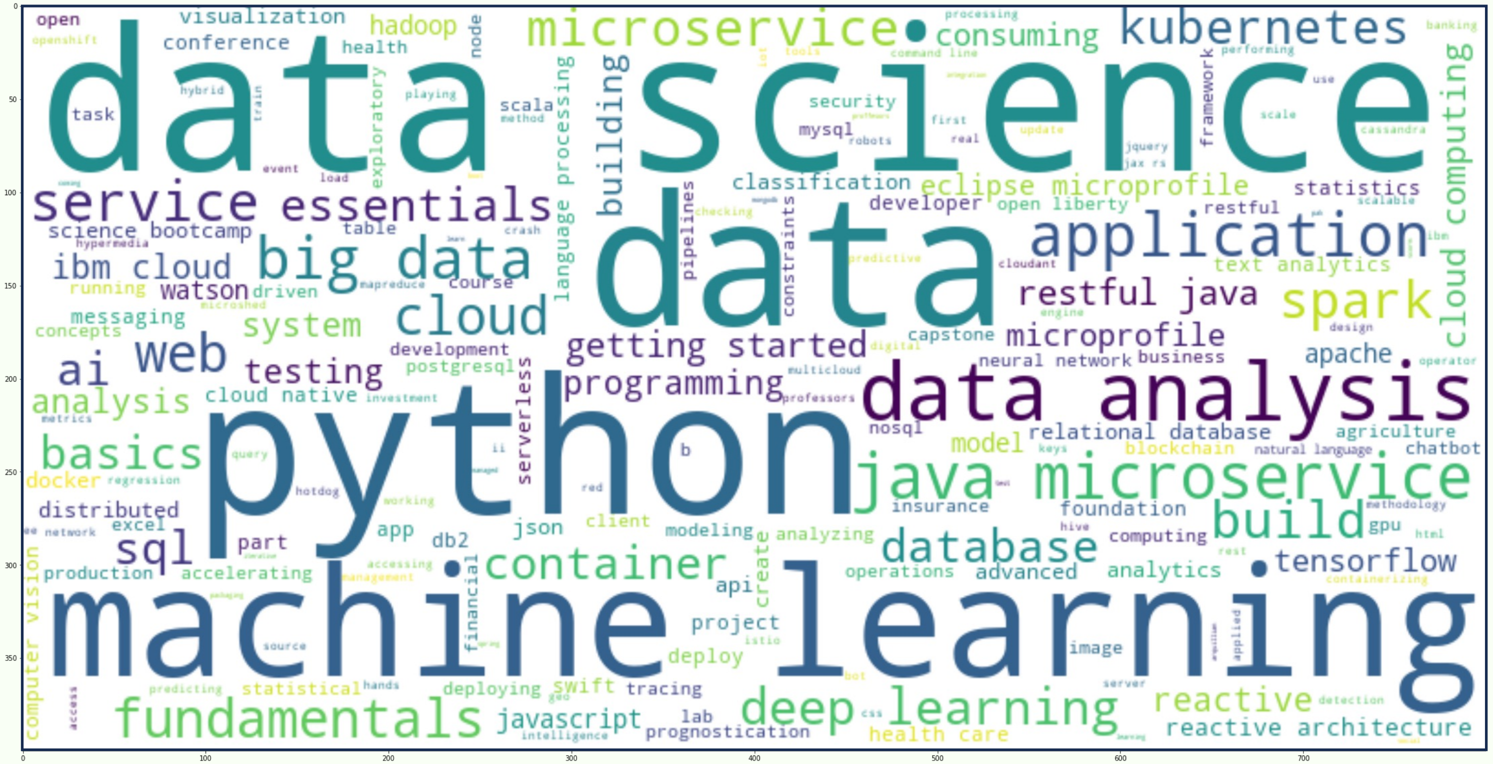
- **Python for Data Science #1** with **14,936** enrolments
- **4 of top 10** courses **Python-related**
- **10 of top 20** courses marketed to **beginner learners** (course title containing terms *introduction*, *101*, *fundamentals* or *essentials*)



	course	enrolments
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

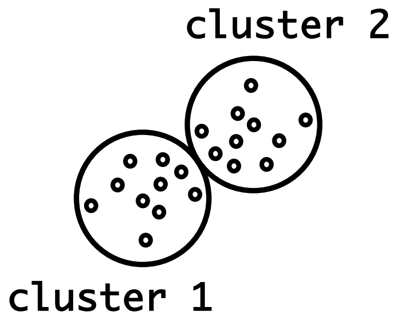
- Terms appearing most frequently in online course descriptions dataset
  - Top terms include *data (science/analysis)*, *python* and *machine learning*



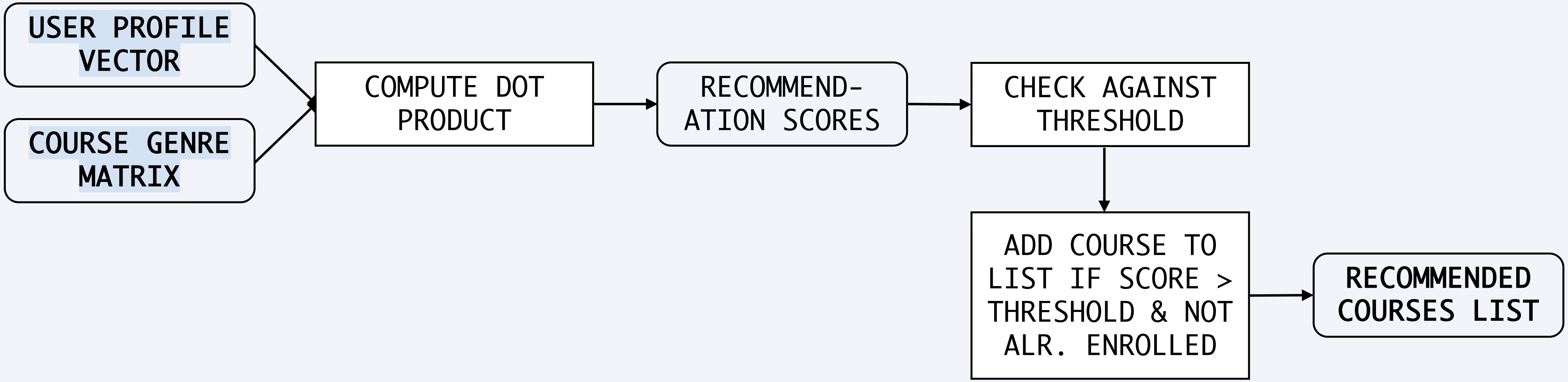
Content-Based

Recommender System

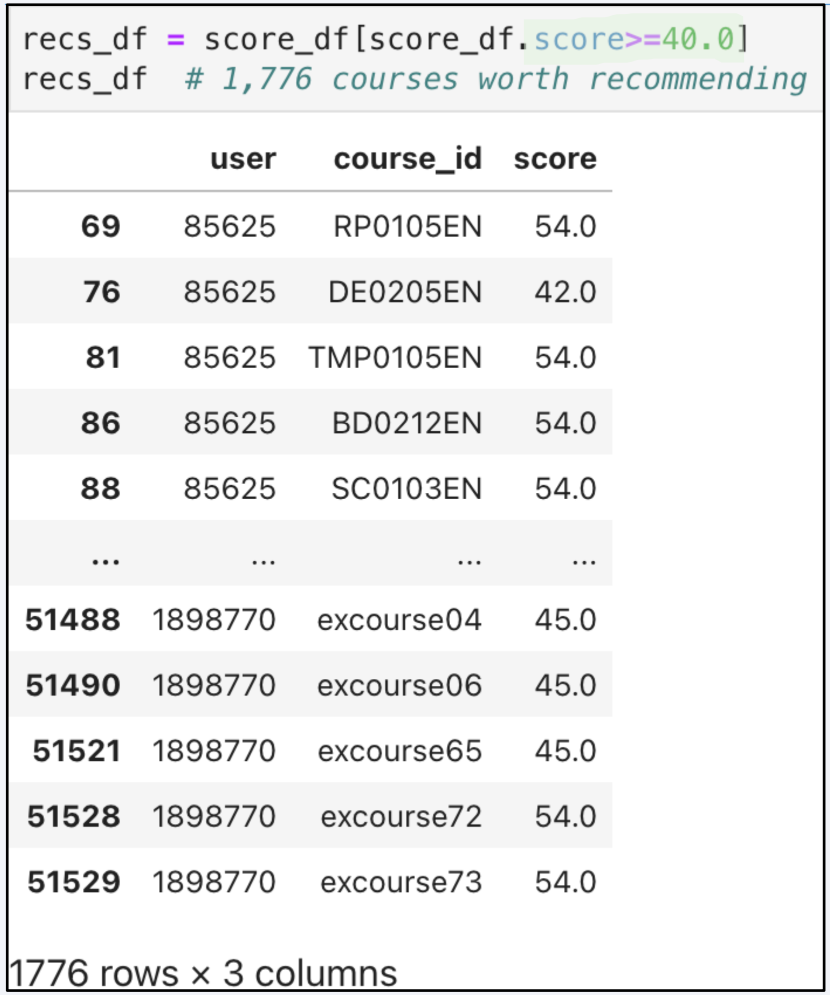
> *Unsupervised Learning*



Content-Based Recommender System Flowchart (user profile, course genres)



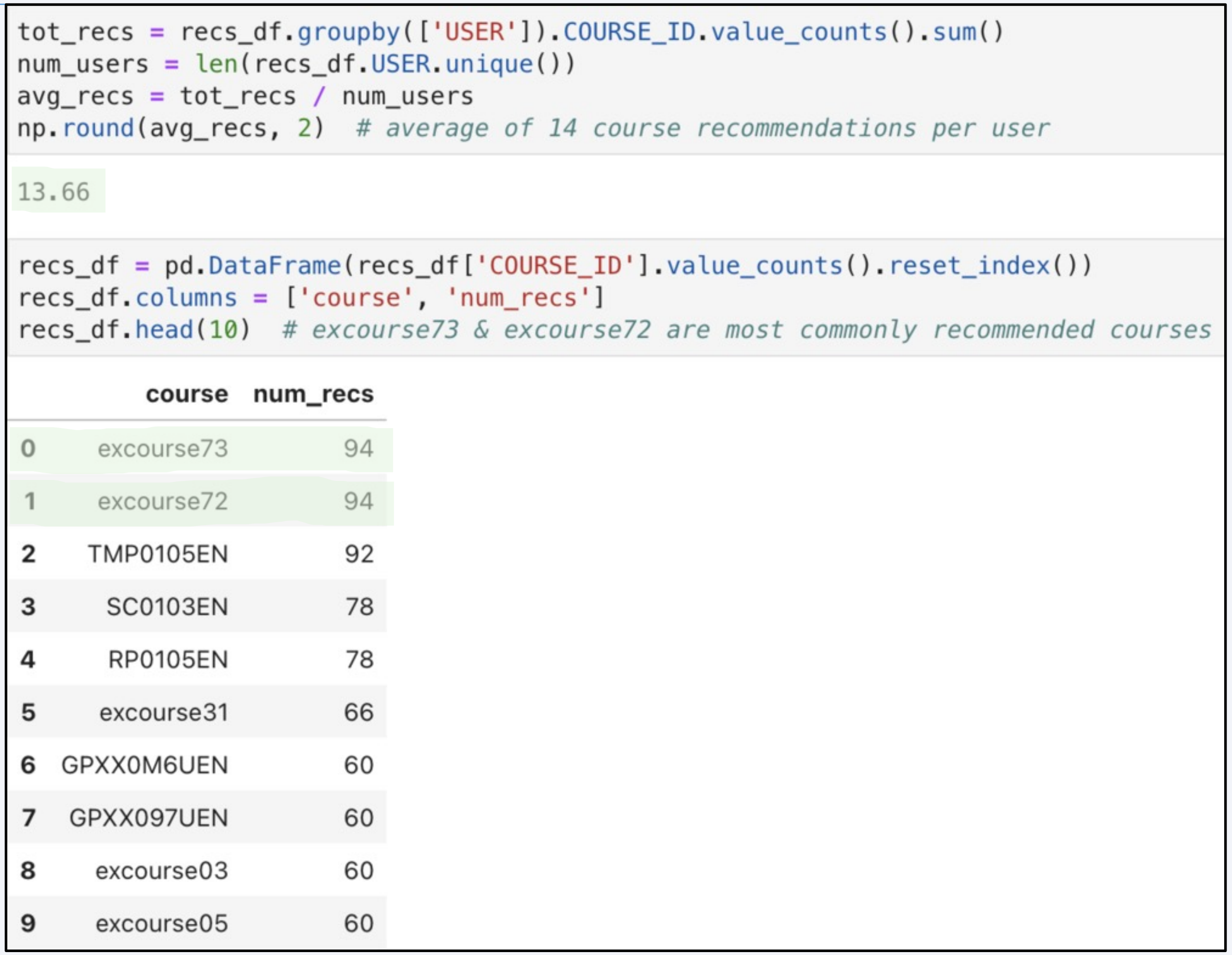
User Profile-Based Recommender System Evaluation Results



```
recs_df = score_df[score_df.score>=40.0]
recs_df  # 1,776 courses worth recommending
```

	user	course_id	score
69	85625	RP0105EN	54.0
76	85625	DE0205EN	42.0
81	85625	TMP0105EN	54.0
86	85625	BD0212EN	54.0
88	85625	SC0103EN	54.0
...	...	...	...
51488	1898770	excourse04	45.0
51490	1898770	excourse06	45.0
51521	1898770	excourse65	45.0
51528	1898770	excourse72	54.0
51529	1898770	excourse73	54.0

1776 rows × 3 columns



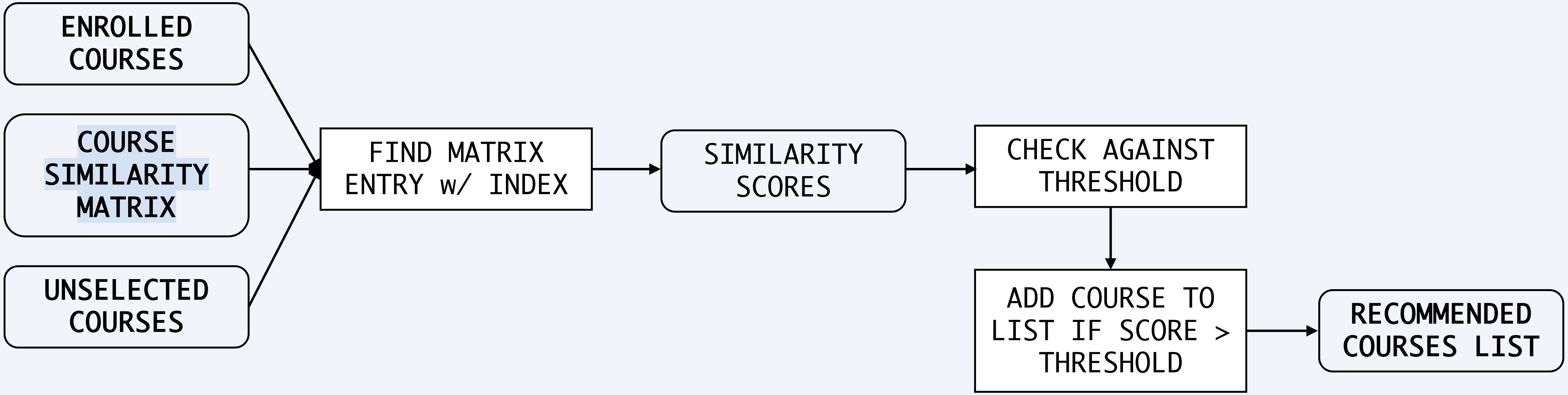
```
tot_recs = recs_df.groupby(['USER']).COURSE_ID.value_counts().sum()
num_users = len(recs_df.USER.unique())
avg_recs = tot_recs / num_users
np.round(avg_recs, 2)  # average of 14 course recommendations per user
```

```
13.66
```

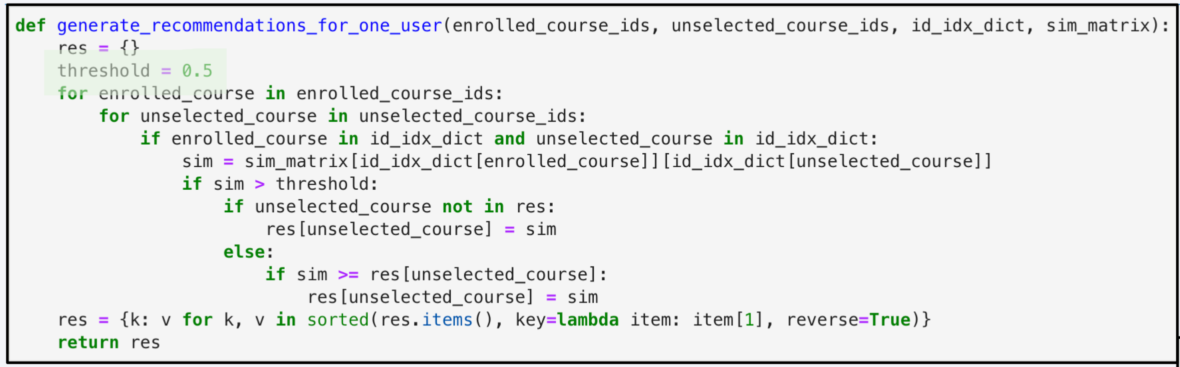
```
recs_df = pd.DataFrame(recs_df['COURSE_ID'].value_counts().reset_index())
recs_df.columns = ['course', 'num_recs']
recs_df.head(10)  # excourse73 & excourse72 are most commonly recommended courses
```

	course	num_recs
0	excourse73	94
1	excourse72	94
2	TMP0105EN	92
3	SC0103EN	78
4	RP0105EN	78
5	excourse31	66
6	GPXX0M6UEN	60
7	GPXX097UEN	60
8	excourse03	60
9	excourse05	60

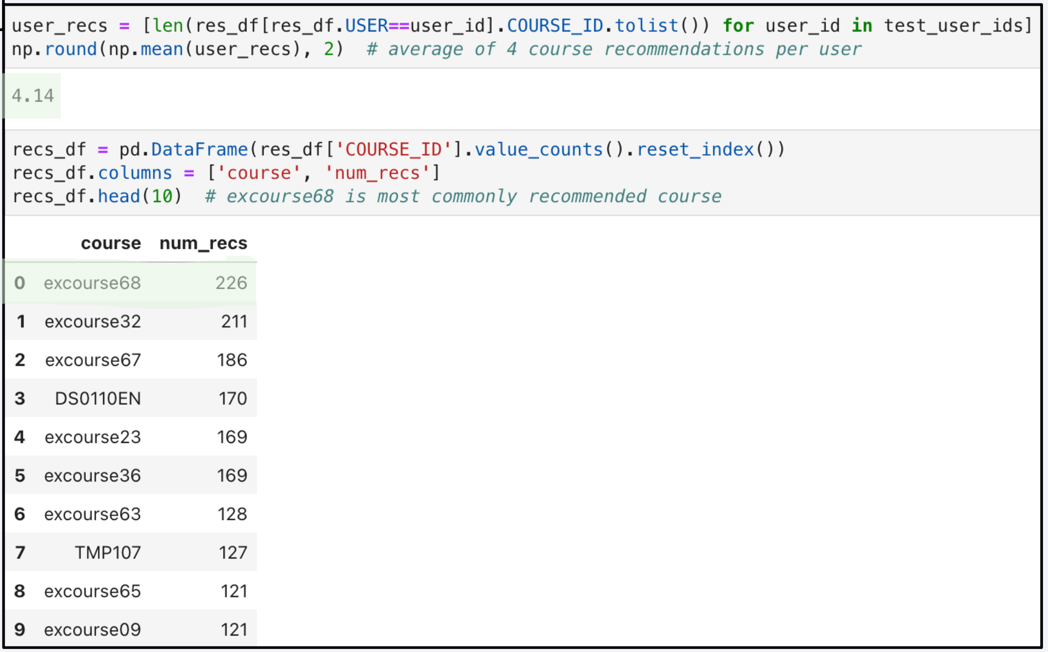
Content-Based Recommender System Flowchart (course similarity)



Course Similarity-Based Recommender System Evaluation Results



```
def generate_recommendations_for_one_user(enrolled_course_ids, unselected_course_ids, id_idx_dict, sim_matrix):
    res = {}
    threshold = 0.5
    for enrolled_course in enrolled_course_ids:
        for unselected_course in unselected_course_ids:
            if enrolled_course in id_idx_dict and unselected_course in id_idx_dict:
                sim = sim_matrix[id_idx_dict[enrolled_course]][id_idx_dict[unselected_course]]
                if sim > threshold:
                    if unselected_course not in res:
                        res[unselected_course] = sim
                    else:
                        if sim >= res[unselected_course]:
                            res[unselected_course] = sim
    res = {k: v for k, v in sorted(res.items(), key=lambda item: item[1], reverse=True)}
    return res
```



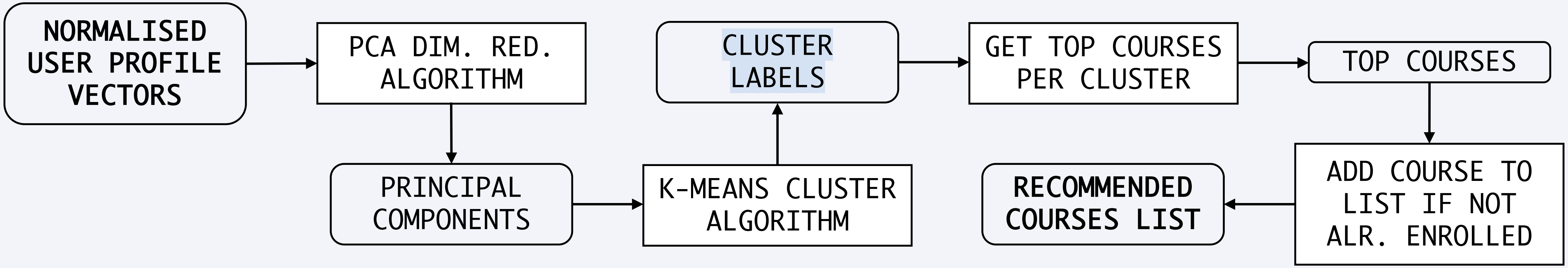
```
user_recs = [len(res_df[res_df.USER==user_id].COURSE_ID.tolist()) for user_id in test_user_ids]
np.round(np.mean(user_recs), 2)  # average of 4 course recommendations per user
```

```
4.14
```

```
recs_df = pd.DataFrame(res_df['COURSE_ID'].value_counts().reset_index())
recs_df.columns = ['course', 'num_recs']
recs_df.head(10)  # excourse68 is most commonly recommended course
```

	course	num_recs
0	excourse68	226
1	excourse32	211
2	excourse67	186
3	DS0110EN	170
4	excourse23	169
5	excourse36	169
6	excourse63	128
7	TMP107	127
8	excourse65	121
9	excourse09	121

Clustering-Based Recommender System Flowchart



NORMALISED USER PROFILE VECTORS

PCA DIM. RED. ALGORITHM

PRINCIPAL COMPONENTS

K-MEANS CLUSTER ALGORITHM

CLUSTER LABELS

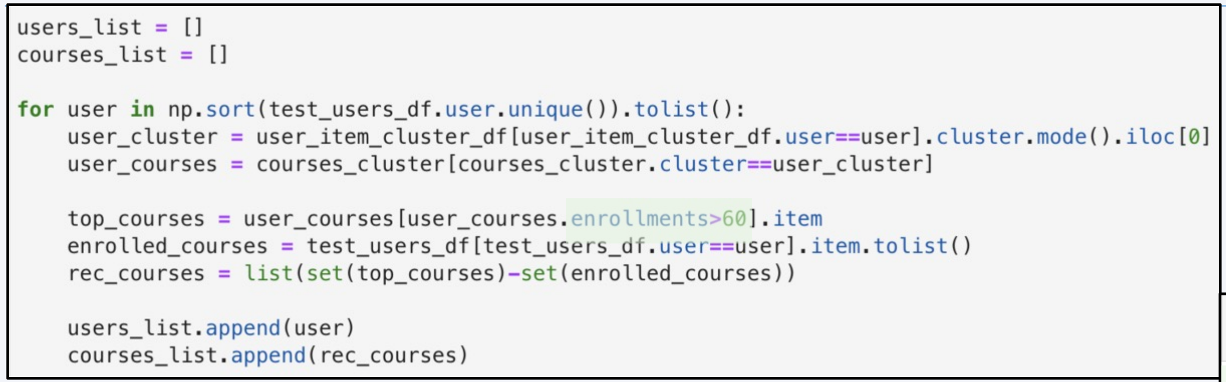
GET TOP COURSES PER CLUSTER

TOP COURSES

ADD COURSE TO LIST IF NOT ALR. ENROLLED

RECOMMENDED COURSES LIST

Clustering-Based Recommender System Evaluation Results

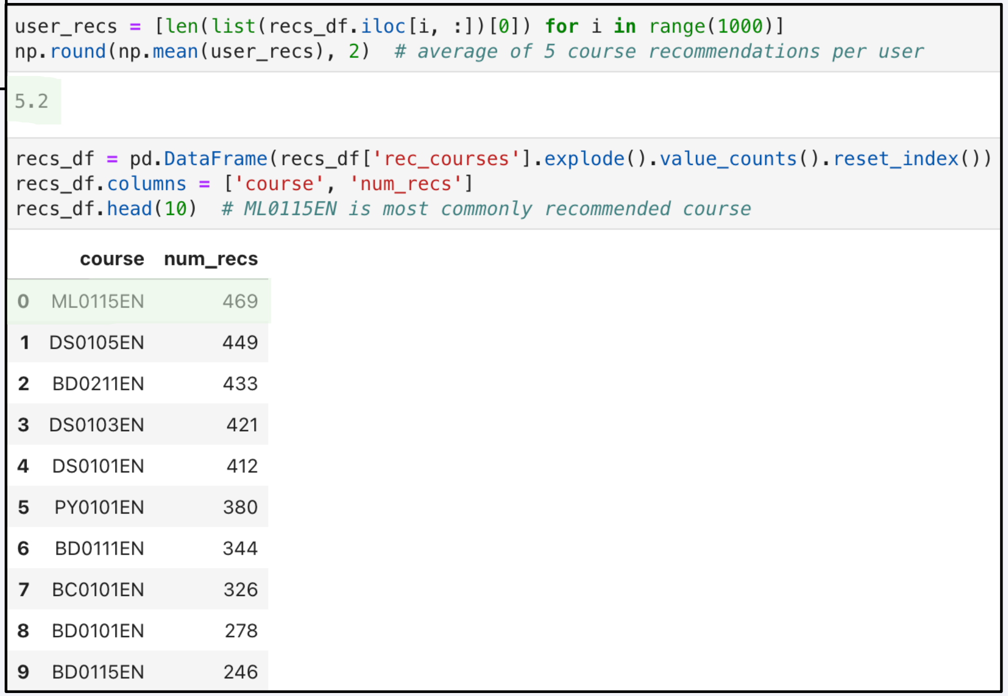


```python
users_list = []
courses_list = []

for user in np.sort(test_users_df.user.unique()).tolist():
    user_cluster = user_item_cluster_df[user_item_cluster_df.user==user].cluster.mode().iloc[0]
    user_courses = courses_cluster[courses_cluster.cluster==user_cluster]

    top_courses = user_courses[user_courses.enrollments>60].item
    enrolled_courses = test_users_df[test_users_df.user==user].item.tolist()
    rec_courses = list(set(top_courses)-set(enrolled_courses))

    users_list.append(user)
    courses_list.append(rec_courses)
```



```python
user_recs = [len(list(recs_df.iloc[i, :])[0]) for i in range(1000)]
np.round(np.mean(user_recs), 2)  # average of 5 course recommendations per user
```

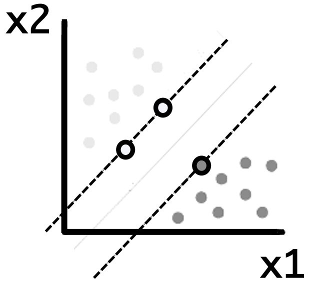
```
5.2
```

```python
recs_df = pd.DataFrame(recs_df['rec_courses'].explode().value_counts().reset_index())
recs_df.columns = ['course', 'num_recs']
recs_df.head(10)  # ML0115EN is most commonly recommended course
```

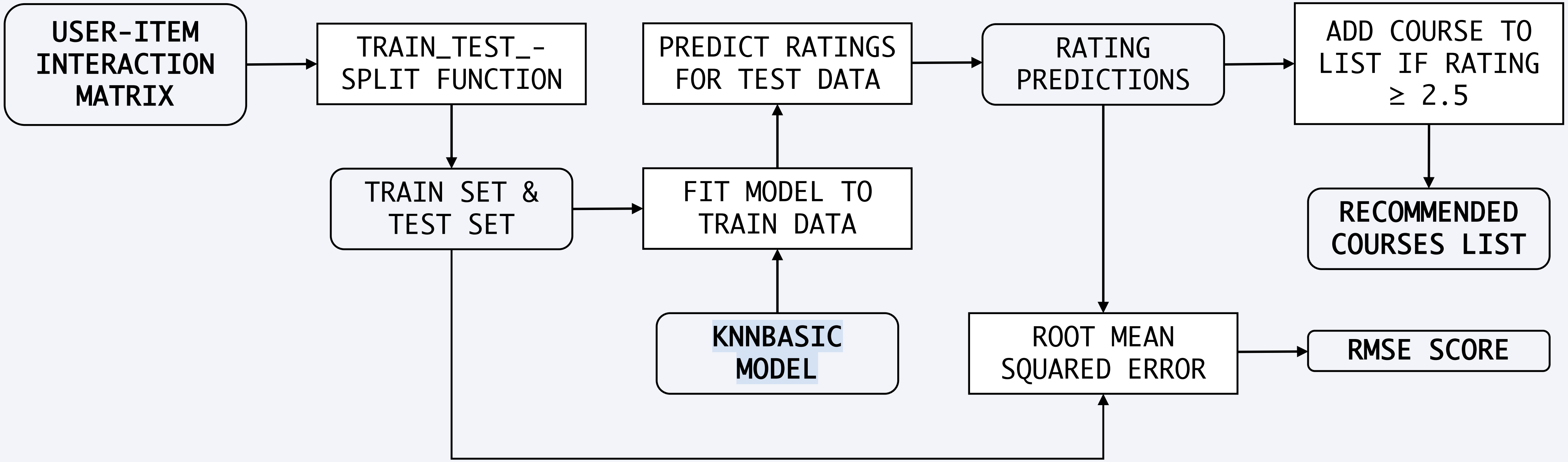
	course	num_recs
0	ML0115EN	469
1	DS0105EN	449
2	BD0211EN	433
3	DS0103EN	421
4	DS0101EN	412
5	PY0101EN	380
6	BD0111EN	344
7	BC0101EN	326
8	BD0101EN	278
9	BD0115EN	246

Collaborative Filtering-Based Recommender System

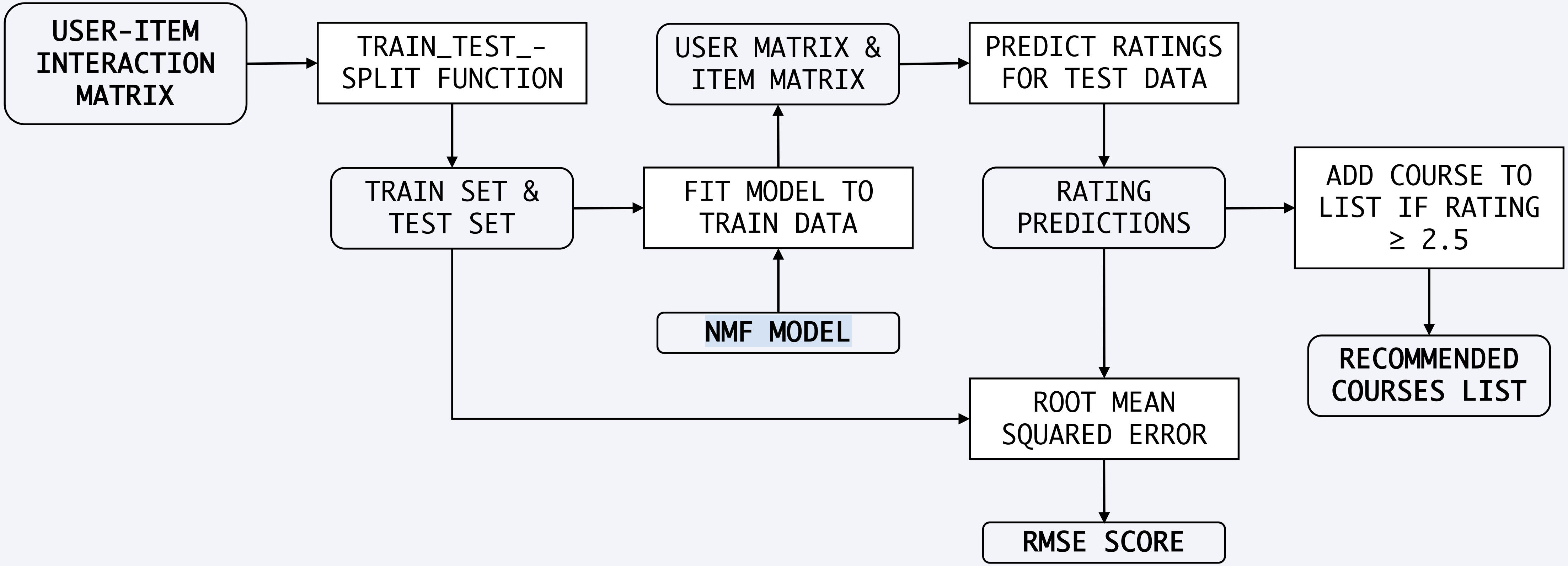
> *Supervised Learning*



KNN-Based Recommender System Flowchart

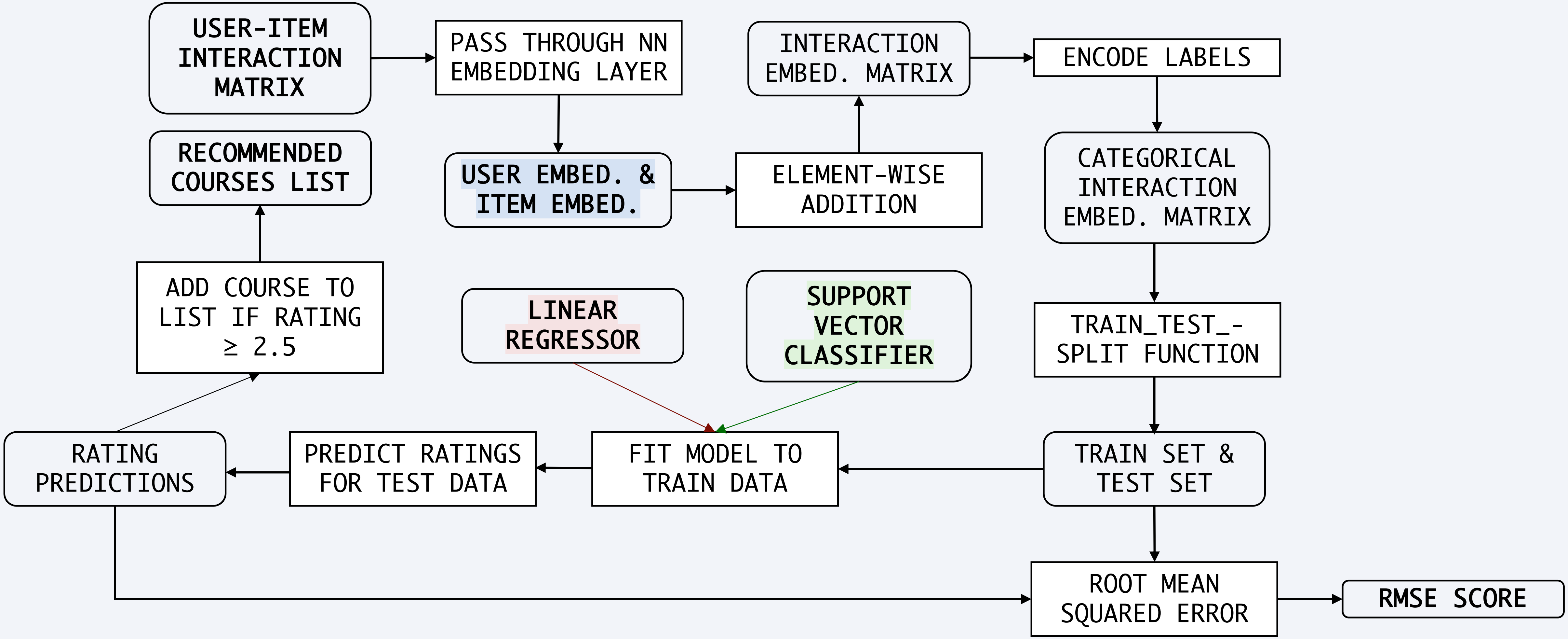


NMF-Based Recommender System Flowchart


USER-ITEM INTERACTION MATRIX
TRAIN_TEST_-SPLIT FUNCTION
USER MATRIX & ITEM MATRIX
PREDICT RATINGS FOR TEST DATA
TRAIN SET & TEST SET
FIT MODEL TO TRAIN DATA
RATING PREDICTIONS
ADD COURSE TO LIST IF RATING ≥ 2.5
NMF MODEL
RECOMMENDED COURSES LIST
ROOT MEAN SQUARED ERROR
RMSE SCORE

Neural Network Embedding-Based Recommender System Flowchart

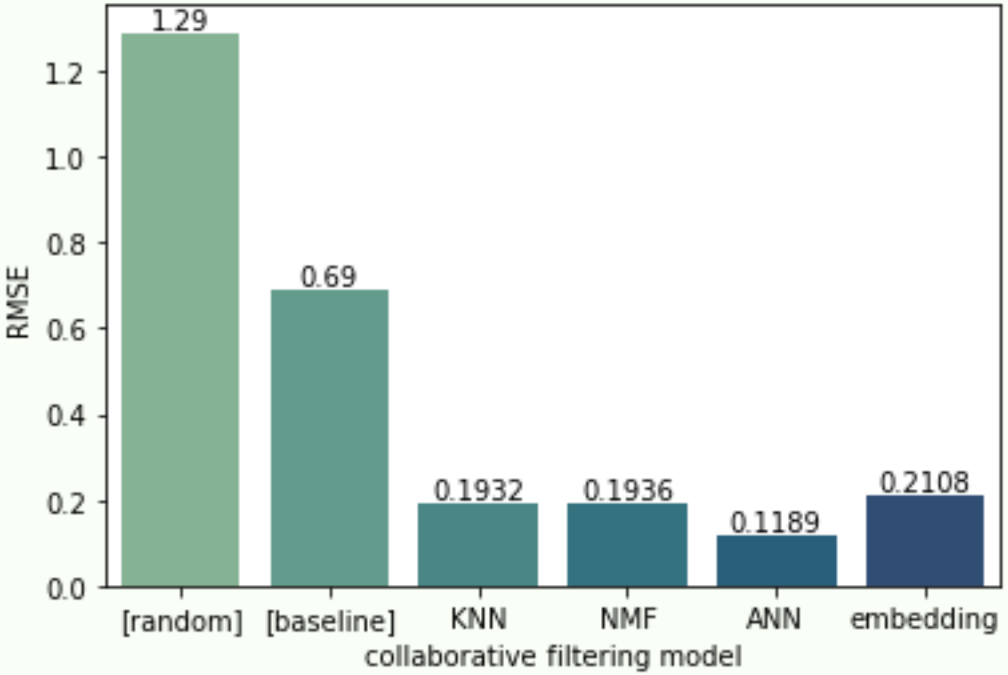
REGRESSION
CLASSIFICATION



USER-ITEM INTERACTION MATRIX
PASS THROUGH NN EMBEDDING LAYER
USER EMBED. & ITEM EMBED.
ELEMENT-WISE ADDITION
INTERACTION EMBED. MATRIX
ENCODE LABELS
CATEGORICAL INTERACTION EMBED. MATRIX
TRAIN_TEST_-SPLIT FUNCTION
TRAIN SET & TEST SET
ROOT MEAN SQUARED ERROR
RMSE SCORE
LINEAR REGRESSOR
SUPPORT VECTOR CLASSIFIER
FIT MODEL TO TRAIN DATA
PREDICT RATINGS FOR TEST DATA
RATING PREDICTIONS
ADD COURSE TO LIST IF RATING ≥ 2.5
RECOMMENDED COURSES LIST

Performance of Collaborative Filtering Models Compared

- **Artificial neural network (ANN)** model performed best with RMSE score of **0.1189**
- Scores of models tested all lie within **0.1 range**
  - Overall performance more evenly distributed than expected
- **Linear regression** model fit to neural network embedding
  - **Support vector machine classification** model also performed well with **98.4%** accuracy, **99.7%** recall, **98.7%** precision and F1 score of **0.9926**

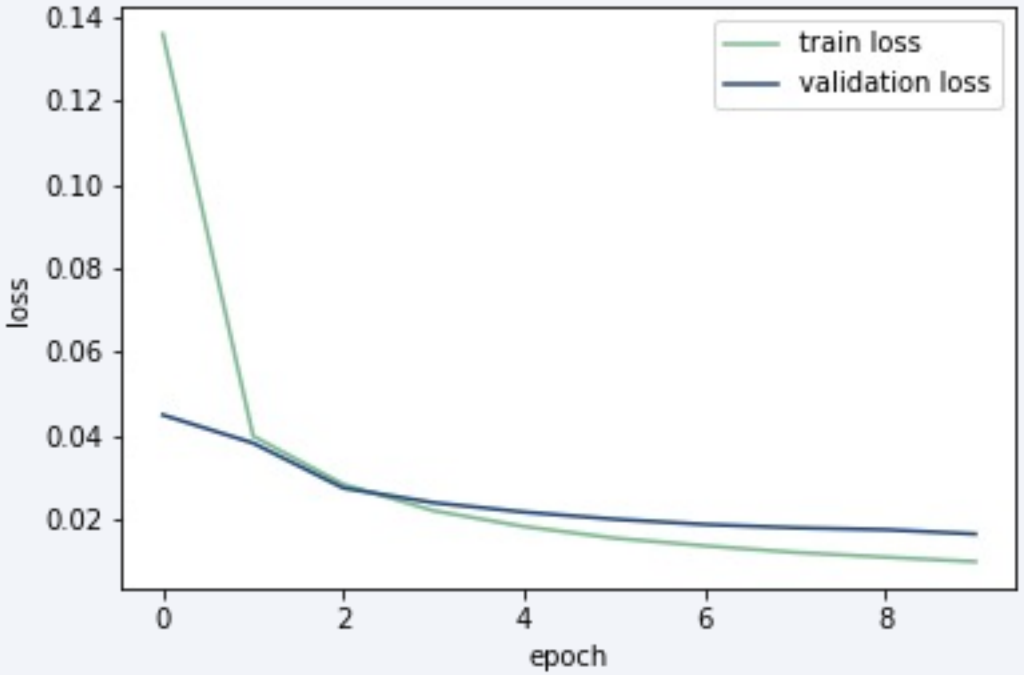


Observations

- **User profile-based** recommender system most liberal with **~14** course recommendations per user
  - **Course similarity-** and **clustering-based** systems discriminate relevant courses from rest more strictly, making **~4** and **~5** recommendations respectively
- Recommended courses **vary greatly** from **model to model**
  - No course recommended by all 3 content-based systems
  - User profile- and course similarity-based systems generate **similar** recommendations, clustering-based system noticeably **different**
    - Former systems push course category *excourse* (5 and 8 recommendations in top 10 respectively), latter does not (0 in top 10)
    - Checks out – former are similar methodologically (supervised learning; rows of database are users, columns are courses), latter is different (unsupervised learning)
- Diminishing returns on *RecommenderNet* ANN model train and validation loss after **2nd epoch** (see FIGURE →)
  - **~600,000-parameter** model potentially overfit; can be **regularised** and optimised

Appendix

- FIGURE



- All *Machine Learning Capstone* labs accessible **here**

github.com/danswk
/ibm/tree/main/ml-capstone