THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

COMP4434 Big Data Analytics

Lab 9
Content-based Recommendation

HUANG Xiao
xiaohuang@comp.polyu.edu.hk

A Recommendation Dataset: MovieLens

The ml-latest.zip Dataset describes 5-star rating and tagging activity from MovieLens, a movie recommender service. It contains approximately 33,000,000 ratings and 2,000,000 tag applications applied to 86,000 movies by 330,975 users. Includes tag genome data with 14 million relevance scores across 1,100 tags. Last updated 9/2018. The data are contained in the files *genome-scores.csv*, *genome-tags.csv*, *links.csv*, *movies.csv*, *ratings.csv* and *tags.csv*.

ratings.csv

User Id	movie Id	Rating	Time stamp
1	307	3.5	1256677221
…	…	…	…

tags.csv

User Id	movie Id	tag	Time stamp
14	110	philosophy	1442615158
…	…	…	…

movies.csv

movieId title genres

links.csv

movieId imdbId tmdbId

Genome-scores.csv

movieId tagId relevance

Genome-tags.csv

tagId tag

Load the MovieLens Data

Download the file ml_latest.zip here and then unzip into the data/ directory.

```
In [7]: !ls data/

README.txt          genome-tags.csv    ml-latest.zip      ratings.csv
genome-scores.csv   links.csv          movies.csv         tags.csv
```

```
In [8]: # Read dataframes
df_movies = pd.read_csv('data/movies.csv')
df_links = pd.read_csv('data/links.csv')
df_ratings = pd.read_csv('data/ratings.csv')
df_genome_tags = pd.read_csv('data/genome-tags.csv')
df_genome_scores = pd.read_csv('data/genome-scores.csv')

# Merge scores and tags
df_movie_tags_in_text = pd.merge(df_genome_scores, df_genome_tags, on='tagId')[['movieId', 'tag', 'relevance']]

# Only keep tags with relevance higher than 0.3
df_movie_tags = df_genome_scores[df_genome_scores.relevance > 0.3][['movieId', 'tagId']]
```

Merge score and tags

Show the movie with Id 1:

```
In [9]: df_movies[df_movies.movieId == 1]
```

Out[9]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy

Content-Based Recommender Systems

Convert the above tags into a vector representation:

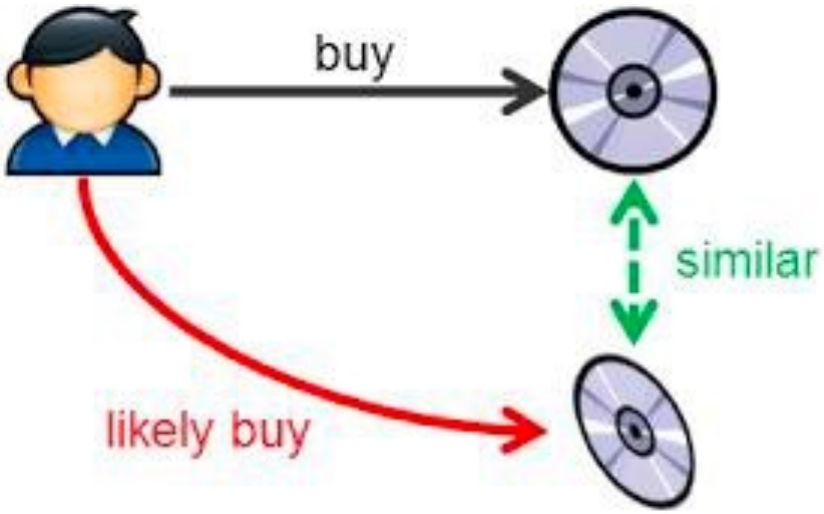
```
In [53]: tf_idf = TfidfVectorizer()

In [54]: df_movies_tf_idf_described = tf_idf.fit_transform(df_data_with_tags.movie_tags)
```

The TF*IDF algorithm is used to weight a keyword in any document and assign the importance to that keyword based on the number of times it appears in the document.

	movieId	title	genres	rating_mean	rating_median	num_ratingsdf_tags_per_movie	movie_tags
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.246583	3.0	27143.0	193 345 1076 1074 389 1090 61 439 454 29 717 4...
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.173981	3.0	15585.0	302 387 417 742 1057 810 299 445 465 1102 264 ...
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.874540	3.0	2989.0	302 545 387 613 497 179 396 742 299 445 412 80...
4	5	Father of the Bride Part II (1995)	Comedy	3.077291	3.0	15474.0	376 387 1004 497 417 348 742 299 445 1102 264 ...
5	6	Heat (1995)	Action|Crime|Thriller	3.844211	4.0	28683.0	297 423 622 467 303 465 162 758 300 1051 269 3...

df_data_with_tags (13176 rows x 7 columns)

How can we find the similarity between items?



- In model-building stage, the system first find the similarity between all pairs of items;
- Then, it uses the most similar items to a user's already-rated items to generate a list of recommendations in recommendation stage.

Cosine Similarity $sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$

Calculating Cosine Similarity

$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Calculate cosine similarity

```
In [*]: m2m = cosine_similarity(df_movies_tf_idf_described)

In [*]: df_tfidf_m2m = pd.DataFrame(cosine_similarity(df_movies_tf_idf_described))

In [*]: index_to_movie_id = df_data_with_tags['movieId']

In [*]: df_tfidf_m2m.columns = [str(index_to_movie_id[int(col)]) for col in df_tfidf_m2m.columns]

In [*]: df_tfidf_m2m.index = [index_to_movie_id[idx] for idx in df_tfidf_m2m.index]
```

```
In [104]: df_tfidf_m2m.head()
```

Out[104]:

	1	2	3	4	5	6	7	8	9	10	...	184987	184997	185029	185135	185425
1	1.000000	0.359995	0.140584	0.163904	0.197146	0.267026	0.240104	0.233925	0.075557	0.223134	...	0.231415	0.323718	0.449159	0.415062	0.115754
2	0.359995	1.000000	0.116658	0.123059	0.119013	0.090835	0.215883	0.221415	0.167558	0.221940	...	0.309822	0.231912	0.207119	0.253158	0.151519
3	0.140584	0.116658	1.000000	0.192486	0.407801	0.090215	0.246536	0.151995	0.077091	0.142224	...	0.118169	0.198064	0.173156	0.146563	0.090056
4	0.163904	0.123059	0.192486	1.000000	0.278716	0.075740	0.334642	0.200485	0.049504	0.079378	...	0.151011	0.195374	0.211978	0.181477	0.214305
5	0.197146	0.119013	0.407801	0.278716	1.000000	0.085531	0.309019	0.151632	0.067623	0.109039	...	0.147010	0.264331	0.182410	0.163857	0.117392

5 rows × 13176 columns

Most similar movies to “Toy Story”

```
In [23]: df_tfidf_m2m.iloc[0].sort_values(ascending=False)[:10]

Out[23]: 1        1.000000
         3114     0.737982
         4886     0.736047
         2355     0.721830
         78499    0.708378
         76093    0.685637
         5218     0.653424
         4306     0.642794
         6377     0.639971
         68954    0.635059
         Name: 1, dtype: float64
```

Sort the result in descending order

movieId

cosine similarity

```
In [132]: df_data_with_tags[df_data_with_tags.movieId == 3114]
```

Out[132]:

	movieId	title	genres	rating_mean	rating_median	num_ratingsdf_tags_per_movie	movie_tags
2809	3114	Toy Story 2 (1999)	Adventure|Animation|Children|Comedy|Fantasy	3.809977	4.0	29820.0	193 30 1051 215 452 840 61 897 1035 1090 454 2...

```
In [25]: df_data_with_tags[df_data_with_tags.movieId == 4886]
```

Out[25]:

	movieId	title	genres	rating_mean	rating_median	num_ratingsdf_tags_per_movie	movie_tags
4331	4886	Monsters, Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy	3.861679	4.0	8708.0	216 215 669 664 663 765 136 497 490 493 690 10...

Further Practice

Further tasks:

- Implement Content based recommender system using project dataset
- Implement Collaborative Filtering based recommended system using project dataset

Further readings:

- https://realpython.com/build-recommendation-engine-collaborative-filtering/
- https://en.wikipedia.org/wiki/Tf%E2%80%93idf#:~:text=which%20it%20occurs.-,Definition,document%20or%20a%20web%20page.
- https://www.kdnuggets.com/2019/09/machine-learning-recommender-systems.html
- https://github.com/grahamjenson/list_of_recommender_systems
- An academic Survey