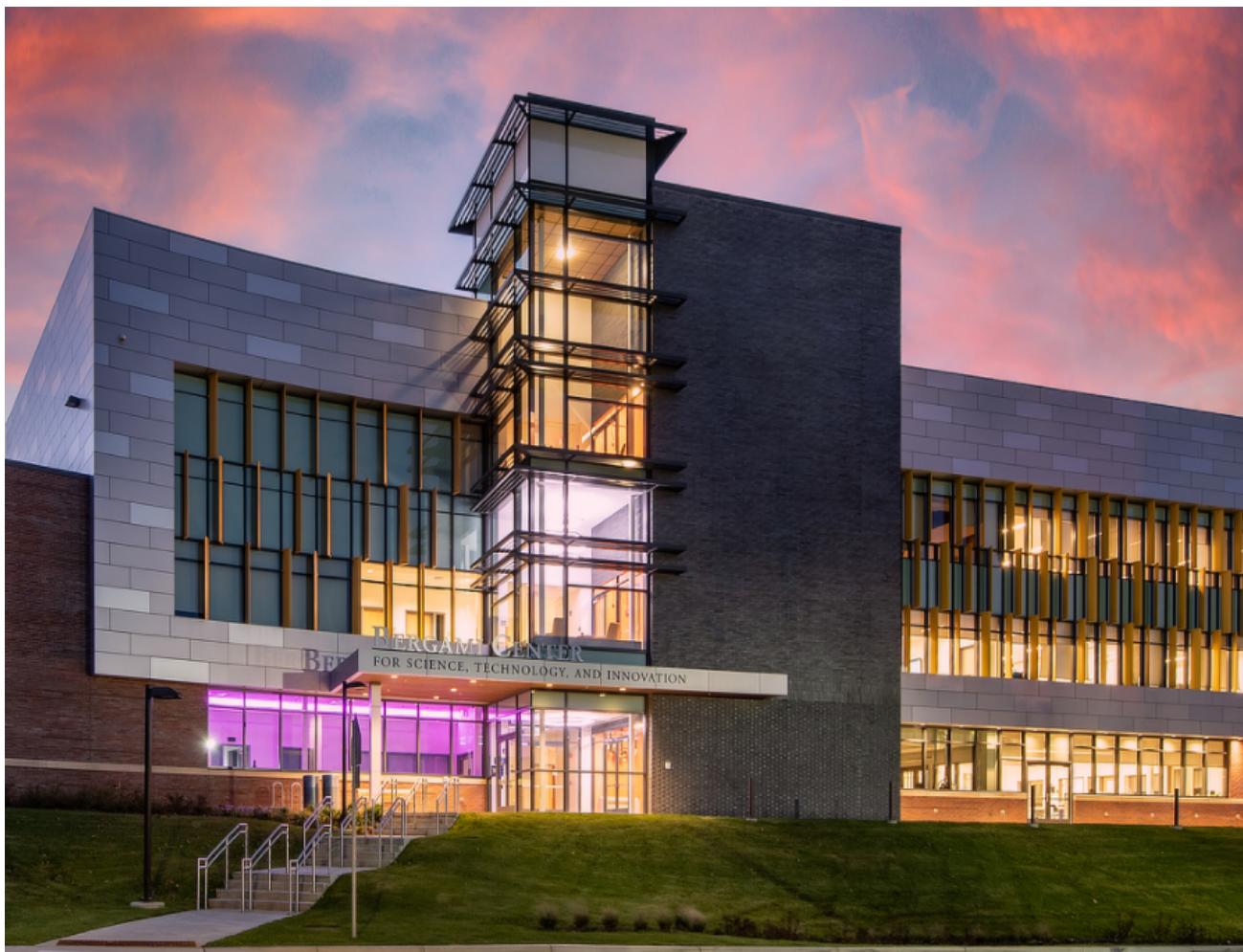


SPRING 22

Electrical & Computer Engineering & Computer Science (ECECS)

TECHNICAL REPORT



CONTENTS

Technical Report	3
Project Name	3
Submitted on:	3
Executive Summary	4
Data Understanding	5
Data Preparation	6
Modeling	7-8
Evaluation	9
Deployment	10
Conclusion	11

Technical Report

Movie Recommendation

Submitted on: 04/24/2022

Team Members:

Name 1 Guanyu Zhou

Name 2 Chunwang Yuan

Questions?

Contact :

cyuan3@unh.newhaven.edu

gzhou2@unh.newhaven.edu



Movie Recommendation

Executive Summary

Movie Recommendation analysis would provide valuable business insights that drive decisions, as it is able to recommend real-time streamed movies and classify the movie based on user preference. A movie watcher would want to know what movies they like are similar, and if he/she will watch before making the next decision.

We were implementing machine learning models in order to compute similar movies based on user input. The outcome recommendation is very important and valuable in business decisions such as keeping daily active users or advertising on the webpage. Companies can estimate how well their movie recommendation performs and decide which business move they are taking next, cutting off a negative movie or promoting a popular movie.

We examined sentiment analysis on TMDB 5000 Movie Dataset from Kaggle.



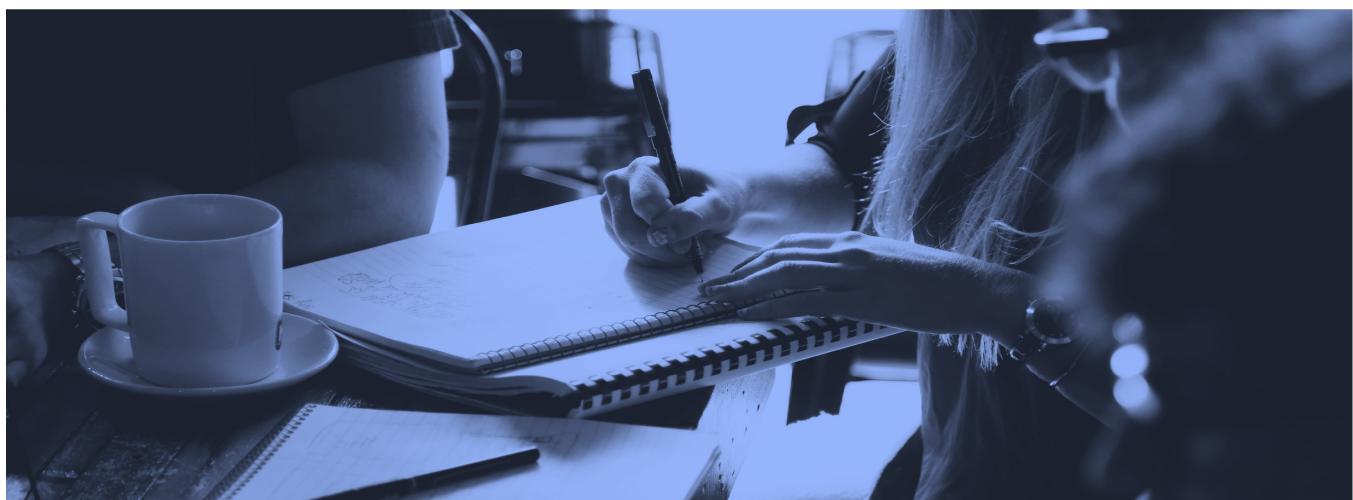
Data Understanding

we are using 2 data files: tmdb_5000_movies.csv and tmdb_5000_credits.csv.
The tmdb_5000_movies.csv has four columns:movie_id, title, cast, crew

df_movies.head(2)				
	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "order": 1, "person_id": 431}, {"cast_id": 243, "character": "Moat", "order": 2, "person_id": 11871}, {"cast_id": 244, "character": "Raya", "order": 3, "person_id": 464}, {"cast_id": 245, "character": "Holland", "order": 4, "person_id": 830}, {"cast_id": 246, "character": "Tugua", "order": 5, "person_id": 432}, {"cast_id": 247, "character": "Murua", "order": 6, "person_id": 11872}, {"cast_id": 248, "character": "Cora", "order": 7, "person_id": 2469}, {"cast_id": 249, "character": "Wol", "order": 8, "person_id": 348}, {"cast_id": 250, "character": "Mirella", "order": 9, "person_id": 2329}, {"cast_id": 251, "character": "Navi", "order": 10, "person_id": 11873}, {"cast_id": 252, "character": "Oce", "order": 11, "person_id": 11874}, {"cast_id": 253, "character": "Killa", "order": 12, "person_id": 2328}, {"cast_id": 254, "character": "Viper", "order": 13, "person_id": 11875}, {"cast_id": 255, "character": "Pezz", "order": 14, "person_id": 11876}, {"cast_id": 256, "character": "Tunk", "order": 15, "person_id": 11877}, {"cast_id": 257, "character": "Leyla", "order": 16, "person_id": 11878}, {"cast_id": 258, "character": "Zu", "order": 17, "person_id": 11879}, {"cast_id": 259, "character": "Dukka", "order": 18, "person_id": 11880}, {"cast_id": 260, "character": "Mam", "order": 19, "person_id": 11881}, {"cast_id": 261, "character": "Gilliam", "order": 20, "person_id": 11882}, {"cast_id": 262, "character": "Bentley", "order": 21, "person_id": 11883}, {"cast_id": 263, "character": "Bakari", "order": 22, "person_id": 11884}, {"cast_id": 264, "character": "Bastian", "order": 23, "person_id": 11885}, {"cast_id": 265, "character": "Bentley", "order": 24, "person_id": 11886}, {"cast_id": 266, "character": "Bentley", "order": 25, "person_id": 11887}, {"cast_id": 267, "character": "Bentley", "order": 26, "person_id": 11888}, {"cast_id": 268, "character": "Bentley", "order": 27, "person_id": 11889}, {"cast_id": 269, "character": "Bentley", "order": 28, "person_id": 11890}, {"cast_id": 270, "character": "Bentley", "order": 29, "person_id": 11891}, {"cast_id": 271, "character": "Bentley", "order": 30, "person_id": 11892}, {"cast_id": 272, "character": "Bentley", "order": 31, "person_id": 11893}, {"cast_id": 273, "character": "Bentley", "order": 32, "person_id": 11894}, {"cast_id": 274, "character": "Bentley", "order": 33, "person_id": 11895}, {"cast_id": 275, "character": "Bentley", "order": 34, "person_id": 11896}, {"cast_id": 276, "character": "Bentley", "order": 35, "person_id": 11897}, {"cast_id": 277, "character": "Bentley", "order": 36, "person_id": 11898}, {"cast_id": 278, "character": "Bentley", "order": 37, "person_id": 11899}, {"cast_id": 279, "character": "Bentley", "order": 38, "person_id": 11900}, {"cast_id": 280, "character": "Bentley", "order": 39, "person_id": 11901}, {"cast_id": 281, "character": "Bentley", "order": 40, "person_id": 11902}, {"cast_id": 282, "character": "Bentley", "order": 41, "person_id": 11903}, {"cast_id": 283, "character": "Bentley", "order": 42, "person_id": 11904}, {"cast_id": 284, "character": "Bentley", "order": 43, "person_id": 11905}, {"cast_id": 285, "character": "Bentley", "order": 44, "person_id": 11906}, {"cast_id": 286, "character": "Bentley", "order": 45, "person_id": 11907}, {"cast_id": 287, "character": "Bentley", "order": 46, "person_id": 11908}, {"cast_id": 288, "character": "Bentley", "order": 47, "person_id": 11909}, {"cast_id": 289, "character": "Bentley", "order": 48, "person_id": 11910}, {"cast_id": 290, "character": "Bentley", "order": 49, "person_id": 11911}, {"cast_id": 291, "character": "Bentley", "order": 50, "person_id": 11912}, {"cast_id": 292, "character": "Bentley", "order": 51, "person_id": 11913}, {"cast_id": 293, "character": "Bentley", "order": 52, "person_id": 11914}, {"cast_id": 294, "character": "Bentley", "order": 53, "person_id": 11915}, {"cast_id": 295, "character": "Bentley", "order": 54, "person_id": 11916}, {"cast_id": 296, "character": "Bentley", "order": 55, "person_id": 11917}, {"cast_id": 297, "character": "Bentley", "order": 56, "person_id": 11918}, {"cast_id": 298, "character": "Bentley", "order": 57, "person_id": 11919}, {"cast_id": 299, "character": "Bentley", "order": 58, "person_id": 11920}, {"cast_id": 300, "character": "Bentley", "order": 59, "person_id": 11921}, {"cast_id": 301, "character": "Bentley", "order": 60, "person_id": 11922}, {"cast_id": 302, "character": "Bentley", "order": 61, "person_id": 11923}, {"cast_id": 303, "character": "Bentley", "order": 62, "person_id": 11924}, {"cast_id": 304, "character": "Bentley", "order": 63, "person_id": 11925}, {"cast_id": 305, "character": "Bentley", "order": 64, "person_id": 11926}, {"cast_id": 306, "character": "Bentley", "order": 65, "person_id": 11927}, {"cast_id": 307, "character": "Bentley", "order": 66, "person_id": 11928}, {"cast_id": 308, "character": "Bentley", "order": 67, "person_id": 11929}, {"cast_id": 309, "character": "Bentley", "order": 68, "person_id": 11930}, {"cast_id": 310, "character": "Bentley", "order": 69, "person_id": 11931}, {"cast_id": 311, "character": "Bentley", "order": 70, "person_id": 11932}, {"cast_id": 312, "character": "Bentley", "order": 71, "person_id": 11933}, {"cast_id": 313, "character": "Bentley", "order": 72, "person_id": 11934}, {"cast_id": 314, "character": "Bentley", "order": 73, "person_id": 11935}, {"cast_id": 315, "character": "Bentley", "order": 74, "person_id": 11936}, {"cast_id": 316, "character": "Bentley", "order": 75, "person_id": 11937}, {"cast_id": 317, "character": "Bentley", "order": 76, "person_id": 11938}, {"cast_id": 318, "character": "Bentley", "order": 77, "person_id": 11939}, {"cast_id": 319, "character": "Bentley", "order": 78, "person_id": 11940}, {"cast_id": 320, "character": "Bentley", "order": 79, "person_id": 11941}, {"cast_id": 321, "character": "Bentley", "order": 80, "person_id": 11942}, {"cast_id": 322, "character": "Bentley", "order": 81, "person_id": 11943}, {"cast_id": 323, "character": "Bentley", "order": 82, "person_id": 11944}, {"cast_id": 324, "character": "Bentley", "order": 83, "person_id": 11945}, {"cast_id": 325, "character": "Bentley", "order": 84, "person_id": 11946}, {"cast_id": 326, "character": "Bentley", "order": 85, "person_id": 11947}, {"cast_id": 327, "character": "Bentley", "order": 86, "person_id": 11948}, {"cast_id": 328, "character": "Bentley", "order": 87, "person_id": 11949}, {"cast_id": 329, "character": "Bentley", "order": 88, "person_id": 11950}, {"cast_id": 330, "character": "Bentley", "order": 89, "person_id": 11951}, {"cast_id": 331, "character": "Bentley", "order": 90, "person_id": 11952}, {"cast_id": 332, "character": "Bentley", "order": 91, "person_id": 11953}, {"cast_id": 333, "character": "Bentley", "order": 92, "person_id": 11954}, {"cast_id": 334, "character": "Bentley", "order": 93, "person_id": 11955}, {"cast_id": 335, "character": "Bentley", "order": 94, "person_id": 11956}, {"cast_id": 336, "character": "Bentley", "order": 95, "person_id": 11957}, {"cast_id": 337, "character": "Bentley", "order": 96, "person_id": 11958}, {"cast_id": 338, "character": "Bentley", "order": 97, "person_id": 11959}, {"cast_id": 339, "character": "Bentley", "order": 98, "person_id": 11960}, {"cast_id": 340, "character": "Bentley", "order": 99, "person_id": 11961}, {"cast_id": 341, "character": "Bentley", "order": 100, "person_id": 11962}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族和充满引力的自然环境中找到引线，才能帮助地球人类重新建立家园。", "popularity": 150.437577, "vote_average": 7.8, "vote_count": 11800}], [{"id": 19995, "original_title": "Avatar", "overview": "In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族 and..."}]	

The tmdb_5000_credits has 21 columns: title, genres, poster, status, runtime, release date, IMDB Rating, etc

df_credits.head(1)																			
budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_countries	release_date	revenue	runtime	spoken_languages	status	tagline	title	vote_average	vote_count
237000000	[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Science Fiction"}]	http://www.avatarmovie.com/	1995	["id": 1463, "name": "culture clash"], ["id": 12, "name": "nam..."]	en	Avatar	In the 22nd century, a paraplegic Marine is assigned to work with an alien race on the planet of潘多拉。他必须在这些原始的、神秘的民族 and...	150.437577	[{"name": "Ingenious Film Partners", "id": 289}, {"name": "United States o..."}]	[{"iso_3166_1": "US", "name": "United States of America"}]	2009-12-10	2787965087	162.0	[{"iso_639_1": "en", "name": "English"}, {"iso_639_1": "...", "name": "..."}]	Released	Enter the World of Avatar Pandora.	Avatar	7.2	11800



Data Preparation

Raw look:

As our data columns contain json objects, we need to convert them back to a list of literals. such as column "genres" etc.



```
df_movies.iloc[0].genres
```



```
[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}, {"id": 878, "name": "Science Fiction"}]
```

Issues:

During cleaning, similar issues such as spaces between words and uppercase/lowercase letters have been solved by us using the NLTK library, which is a natural language machine library. And eventually, acquired the final data frame that we need.



```
new_ml.ipynb()
```



	movie_id	title	tags
0	19995	Avatar	in the 22nd century, a parapleg marin is disp...
1	285	Pirates of the Caribbean: At World's End	captain barbossa, long believ to be dead, ha c...
2	206647	Spectre	a cryptic messag from bond' past send him on a...
3	49026	The Dark Knight Rises	follow the death of district attorney harvey d...
4	49529	John Carter	john carter is a war-weary, former militari ca...



Modeling

Cosine Similarity

Cosine similarity measures the similarity between two vectors of their inner product space. It simply means to measure if two vectors are in the same direction using the cosine of the angle. The score is between 0 and 1. Two movies can be similarly based on their tags, name, or genres. We are using “tags” as our vectors and determine the similarity between movies.

A vector of the “Tag” look can be found in the above picture. and as the result we enumerate it with movie ID.

```
array([ 19995,     285, 206647, ..., 231617, 126186,  25975])  
[60] sorted(list(enumerate(similarity[0])), reverse=True, key=lambda x:x[1])[1:6]  
[(1216, 0.28676966733820225),  
(2409, 0.26901379342448517),  
(3730, 0.2605130246476754),  
(507, 0.255608593705383),  
(539, 0.25038669783359574)]
```

As we have ready-to-go data frames and we can now query our data frames to recommend movies.
“input movie”: A list of movies of sorted cosine similarity between it and other movies “Tags”.

```

▶ recommend('Batman Begins')

[ (65, 0.40218090755486674), (1364, 0.35434169344615046), (1362, 0.3340765523905305), (3, 0.3177444546511212),
  The Dark Knight
  Batman
  Batman
  The Dark Knight Rises
  10th & Wolf
  Rockaway
  Batman v Superman: Dawn of Justice
  Synecdoche, New York
  Defendor
  Sexy Beast
  City By The Sea
  American Psycho
  Harsh Times
  Batman & Robin

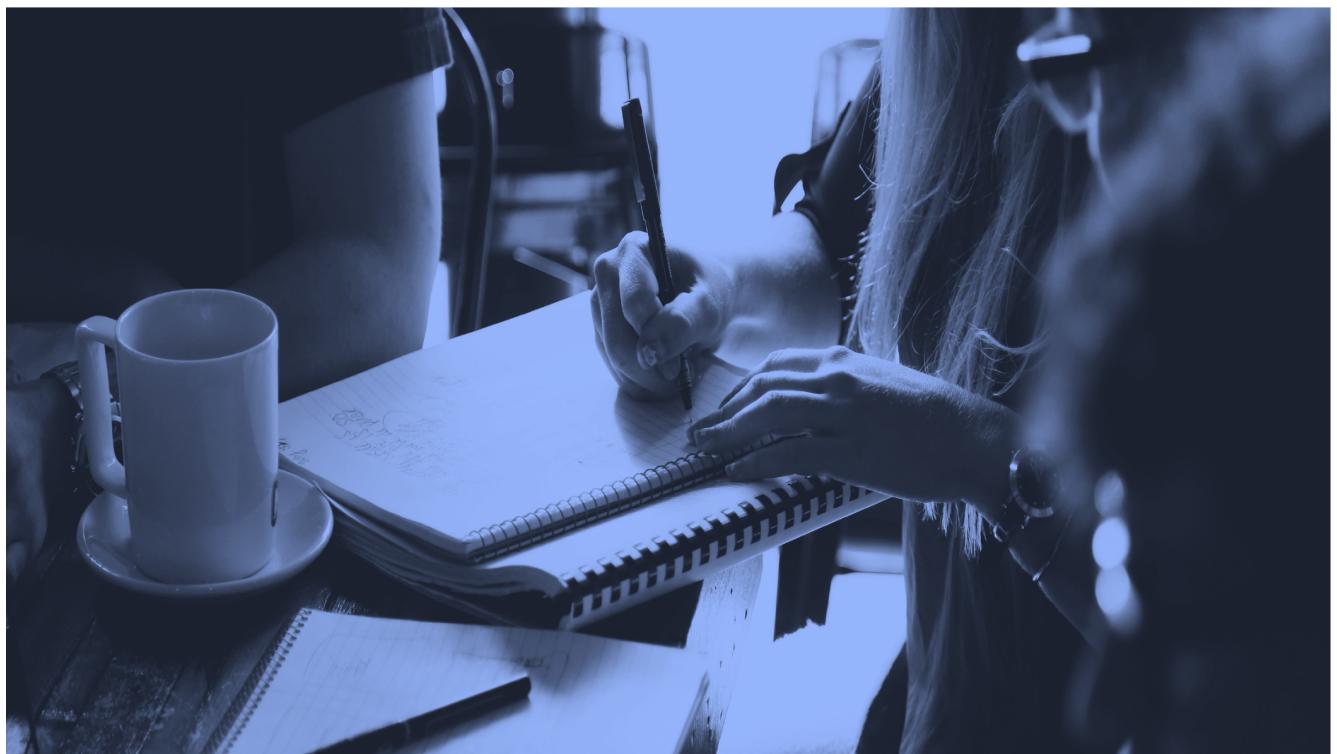
```

Libraries	Processing
NumPy	Cleaning and rearranging arrays
nltk	Cleaning and extracting characters
	Cleaning and removing spaces
	Cleaning and changing lower/upper letters
Pandas	Cleaning and processing data frame
Stream it	Similar to the flask, making the model to the web for later evaluation



Evaluation

We are pickling our final model data for web building later use. As we are using content-based recommendations, the similarity of tags is a very simple recommendation model. The function makes it easy to show other recommendation movies and to improve this function, we can add a lot more features such as “actors”, “status”, “runtime” etc columns. But to prevent it from overfitting with too many features and simply test our cosine similarity method, we are using the simple recommendation model here.



Deployment

The Streamlight is an open-source library for machine learning and data science to turn data scripts into sharable web apps in minutes with its powerful and easy-to-use APIs compared to the flask.

We are making a webpage with a container that can take “user input”, and since it connects to the data frame files that we pickled and provided. It runs through our data frame and gets all of the closed “cosine similarity” movies back. Also since all of the movie's ID is listed in the IMDB database, we are using the API to get images based on the “moive_id”. The website is as shown below:

Movie Recommendations

Which movie do you want to recommend?

Titanic

Recommend

Aliens vs Predator: Requiem



ALIENS VS. PREDATOR
REQUIEM

Link Movie

Aliens



ALIENS

Link Movie

Falcon Rising



FALCON
RISING

Link Movie

The iconic creatures from two of the scariest film franchises in movie history wage their most brutal battle ever—in our When Ripley's lifepod is found by a salvage crew over 50 years later, she finds that terra-formers are on the very planet Chapman is an ex-marine in Brazil's slums, battling the yakuzas who attacked his sister and left her for dead.

Conclusion

In our task of movie recommendation, our cosine similarity method finding out the similarity between vectors, in the words, the relationship between movies, can help movie companies to decide the next move in the business. For example, if a movie is poorly searched or watched, they can delete it from the database to save resources. Similarly, if a movie is very popular, they can redirect a lot of customers to this movie and insert the advertisements in between the watching for profit.

The difficulty that we faced was a feature that we used in cosine similarity is hard to decide which feature should be put in. A simple recommendation model means a simple calculation between two vectors, it could also be underfitting. However, content-based recommendations could also be easily overfitting by adding extra features, since most of them can be completely irrelevant to the input movie.

Our approach is still in its developing stage and our models will need further improvement by adding more information. In the future, we will be exploring more linguistic analysis and using other methods of machine learning to predict and improve our model.

