SI 650 Project Report

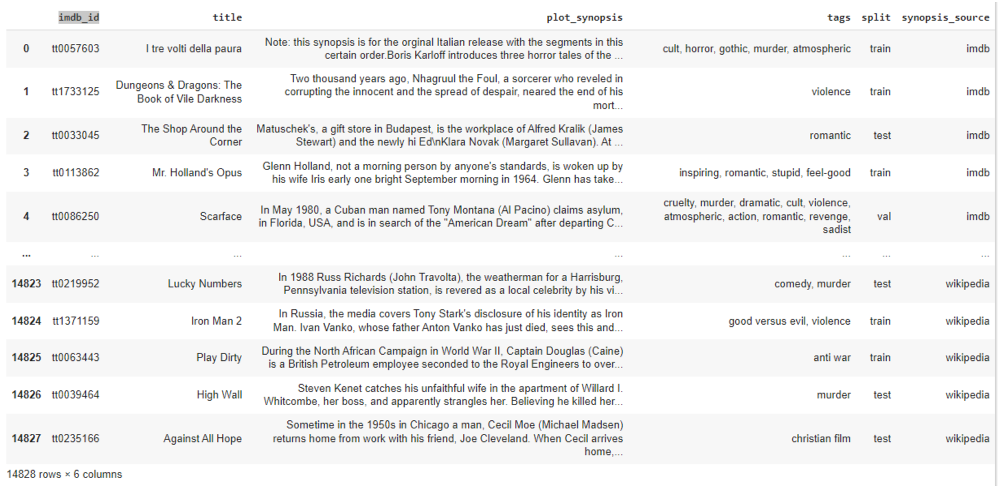
Yifei Sun, Xinqi Guo
yifeisun@umich.edu,
xinqiguo@umich.edu

Introduction

Our goal is to build a vertical search engine to search for movies. Compared to other movie websites such as IMDB, our engine aims to help audiences who just remember a piece of information. This is also known as Tip of the Tongue Known-Item Retrieval. According to Arguello et al, in long-term memory, people tend to forget specific details and retain memories that are more of high-level details or points. Therefore, in these cases, the general way of searching through movie metadata is not effective, because it is difficult for the searcher to clearly remember the director, actors, etc. of the movie. The searcher relies more on the character and image of the main character, the plot, the feeling of watching the movie, the background of the event of watching the movie, and the events that happened at the same time to identify the movie. With these blurred memories, there is a great deal of frustration for the searcher, and at the same time a great deal of satisfaction when the search is successful. In fact, there are now many communities that help with this problem, where the questioner describes a movie, music, or book and the answerer tries to give a possible answer. Our research is also very helpful for various video platforms and forums to further improve the accuracy of results when inputting fuzzy and abstract queries. Our proposed solution is to include more abstract information about movie plots, characters, etc. in the movie search database, rather than just metadata, mainly from movie reviews, taglines, plot keywords, story synopses, etc. This information comes from the distillation and generalization of the film by the viewers who have seen it, and is inherently abstract in character and can be a good response to the problems we face.

Data

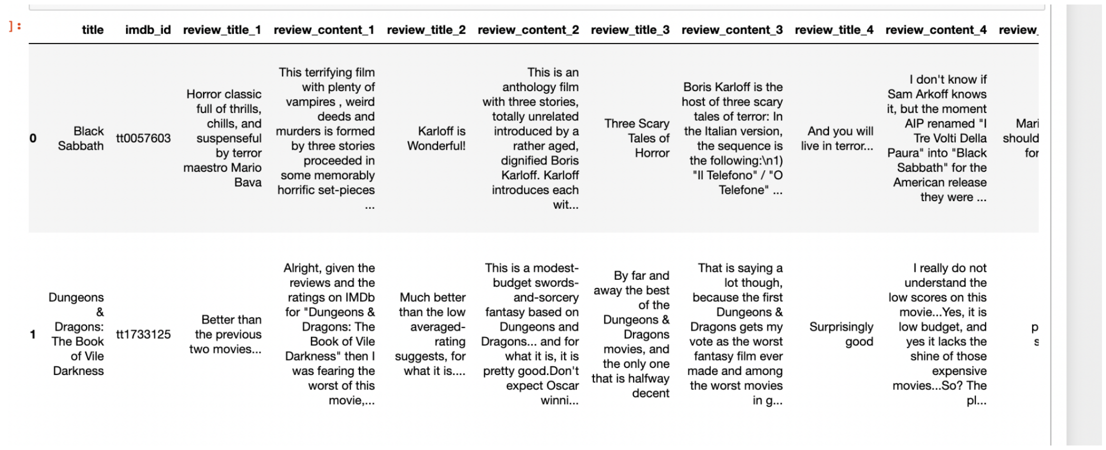
Our data consists of two datasets. The first dataset includes the title, imdb_id, plot_synopsis, and tags of 14k movies from IMDB. The link to the data can be found at https://www.aclweb.org/anthology/L18-1274.
There are around 14800 movies in this dataset. The data includes the movies' titles, the IMDB id, the plot synopsis, tags, and the sources of the plot synopsis.
Here are a few examples in our data:



	imdb_id	title	plot_synopsis	tags	split	synopsis_source
0	tt0057603	I tre volti della paura	Note: this synopsis is for the orginal Italian release with the segments in this certain order.Boris Karloff introduces three horror tales of the ...	cult, horror, gothic, murder, atmospheric	train	imdb
1	tt1733125	Dungeons & Dragons: The Book of Vile Darkness	Two thousand years ago, Nhagruul the Foul, a sorcerer who reveled in corrupting the innocent and the spread of despair, neared the end of his mort...	violence	train	imdb
2	tt0033045	The Shop Around the Corner	Matuschek's, a gift store in Budapest, is the workplace of Alfred Kralik (James Stewart) and the newly hi Ed\nKlara Novak (Margaret Sullavan). At ...	romantic	test	imdb
3	tt0113862	Mr. Holland's Opus	Glenn Holland, not a morning person by anyone's standards, is woken up by his wife Iris early one bright September morning in 1964. Glenn has take...	inspiring, romantic, stupid, feel-good	train	imdb
4	tt0086250	Scarface	In May 1980, a Cuban man named Tony Montana (Al Pacino) claims asylum, in Florida, USA, and is in search of the "American Dream" after departing C...	cruelty, murder, dramatic, cult, violence, atmospheric, action, romantic, revenge, sadist	val	imdb
...	...	...	...	...	...	...
14823	tt0219952	Lucky Numbers	In 1988 Russ Richards (John Travolta), the weatherman for a Harrisburg, Pennsylvania television station, is revered as a local celebrity by his vi...	comedy, murder	test	wikipedia
14824	tt1371159	Iron Man 2	In Russia, the media covers Tony Stark's disclosure of his identity as Iron Man. Ivan Vanko, whose father Anton Vanko has just died, sees this and...	good versus evil, violence	train	wikipedia
14825	tt0063443	Play Dirty	During the North African Campaign in World War II, Captain Douglas (Caine) is a British Petroleum employee seconded to the Royal Engineers to over...	anti war	train	wikipedia
14826	tt0039464	High Wall	Steven Kenet catches his unfaithful wife in the apartment of Willard I. Whitcombe, her boss, and apparently strangles her. Believing he killed her...	murder	test	wikipedia
14827	tt0235166	Against All Hope	Sometime in the 1950s in Chicago a man, Cecil Moe (Michael Madsen) returns home from work with his friend, Joe Cleveland. When Cecil arrives home,...	christian film	test	wikipedia

14828 rows × 6 columns

After creating the index of the current dataset, we found that the performance of base models like BM25 is poor and we were hard to label the relevance score. From the feedback from GSI, we are encouraged to find movie reviews for each movie, because there would be more information in the movie reviews. Thus, we created our second dataset by scraping the top 5 user reviews and review titles by imdb_id from IMDB. Here are the top 2 rows of the joined dataset:

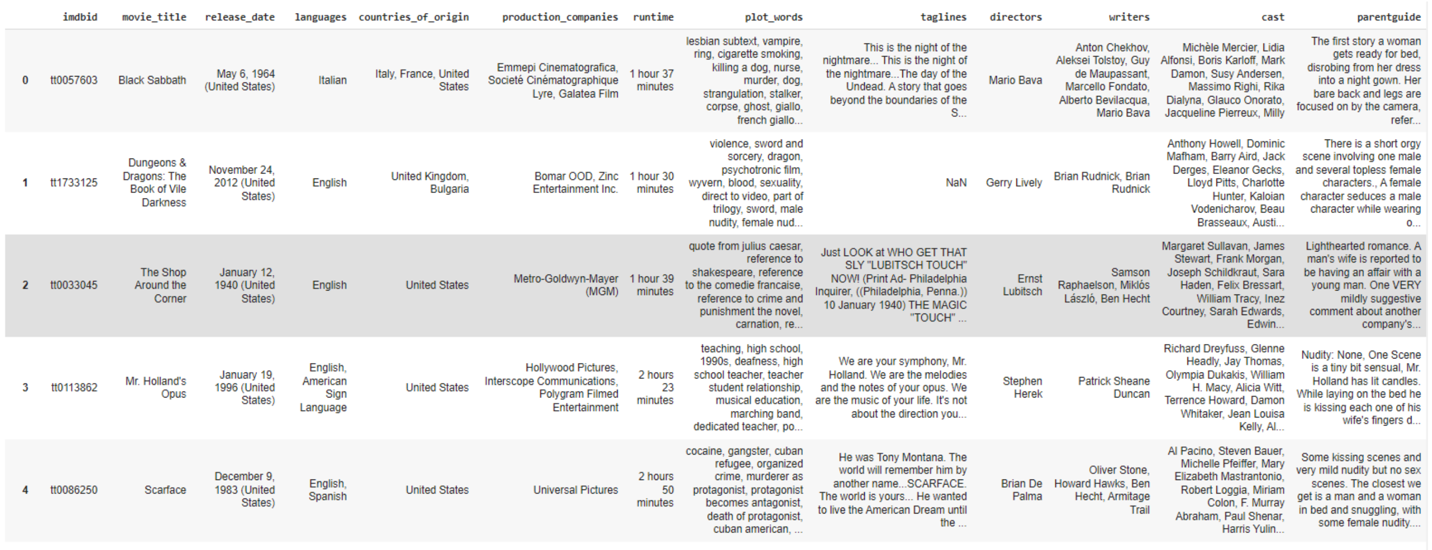


]:

	title	imdb_id	review_title_1	review_content_1	review_title_2	review_content_2	review_title_3	review_content_3	review_title_4	review_content_4	review_
0	Black Sabbath	tt0057603	Horror classic full of thrills, chills, and suspenseful by terror maestro Mario Bava	This terrifying film with plenty of vampires , weird deeds and murders is formed by three stories proceeded in some memorably horrific set-pieces ...	Karloff is Wonderful!	This is an anthology film with three stories, totally unrelated introduced by a rather aged, dignified Boris Karloff. Karloff introduces each wit...	Three Scary Tales of Horror	Boris Karloff is the host of three scary tales of terror: In the Italian version, the sequence is the following:\n1) "Il Telefono" / "O Telefone" ...	And you will live in terror...	I don't know if Sam Arkoff knows it, but the moment AIP renamed "I Tre Volti Della Paura" into "Black Sabbath" for the American release they were ...	Mari should for
1	Dungeons & Dragons: The Book of Vile Darkness	tt1733125	Better than the previous two movies...	Alright, given the reviews and the ratings on IMDb for "Dungeons & Dragons: The Book of Vile Darkness" then I was fearing the worst of this movie,...	Much better than the low averaged-rating suggests, for what it is....	This is a modest-budget swords-and-sorcery fantasy based on Dungeons and Dragons... and for what it is, it is pretty good.Don't expect Oscar winni...	By far and away the best of the Dungeons & Dragons movies, and the only one that is halfway decent	That is saying a lot though, because the first Dungeons & Dragons gets my vote as the worst fantasy film ever made and among the worst movies in g...	Surprisingly good	I really do not understand the low scores on this movie...Yes, it is low budget, and yes it lacks the shine of those expensive movies...So? The pl...	p s

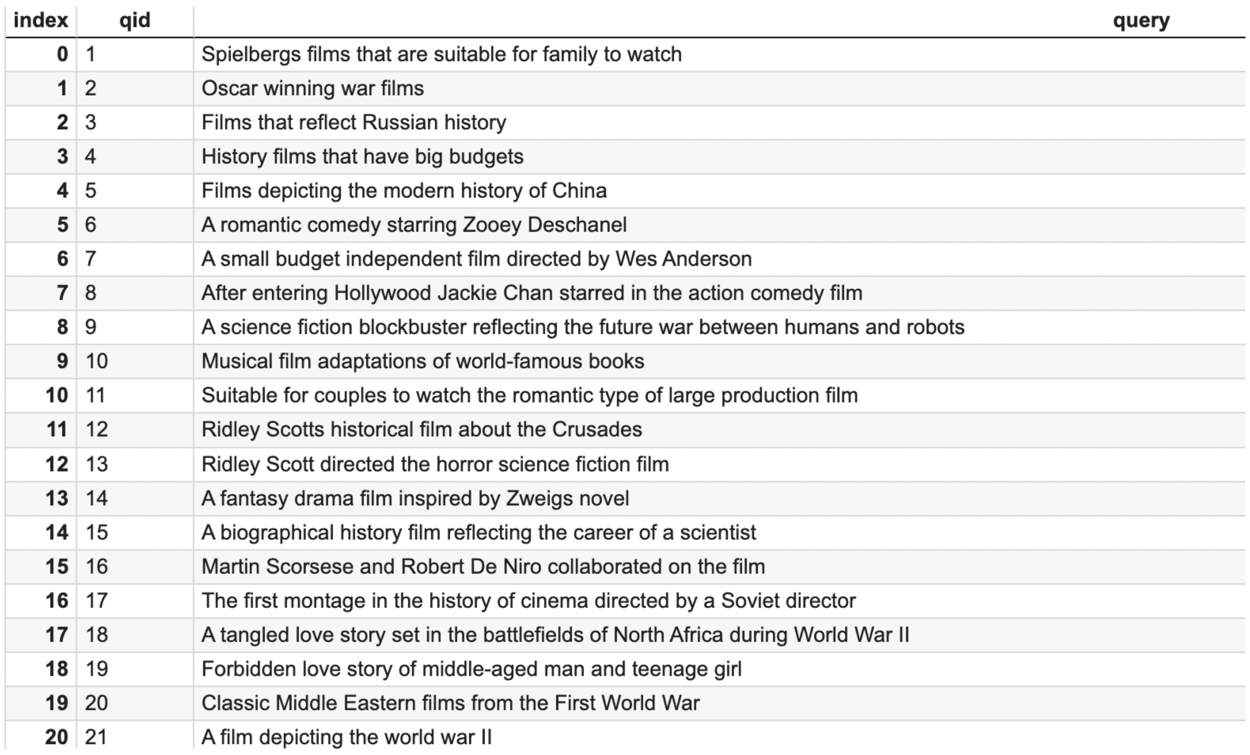
One thing worth noting is that a lot of the data is NA. For example, there are 229 movies that do not have even the first movie reviews, which means they do not have any movie reviews. And there are 931 movies that do not have the fifth review, meaning they only have four. The missing data could be a potential pain point in our future work. Also, it is clear that our dataset is not large enough, 14000 movies is still a small number compared to the full movie dataset in IMDB, but this is the farthest we can get, otherwise, we would need a lot of time scraping from the web.

To increase the number of features and the information about movies we have, we scrape more information from IMDB including the directors, writers, cast, taglines, plot keywords, etc. These features could be used in future work.
Here are some examples

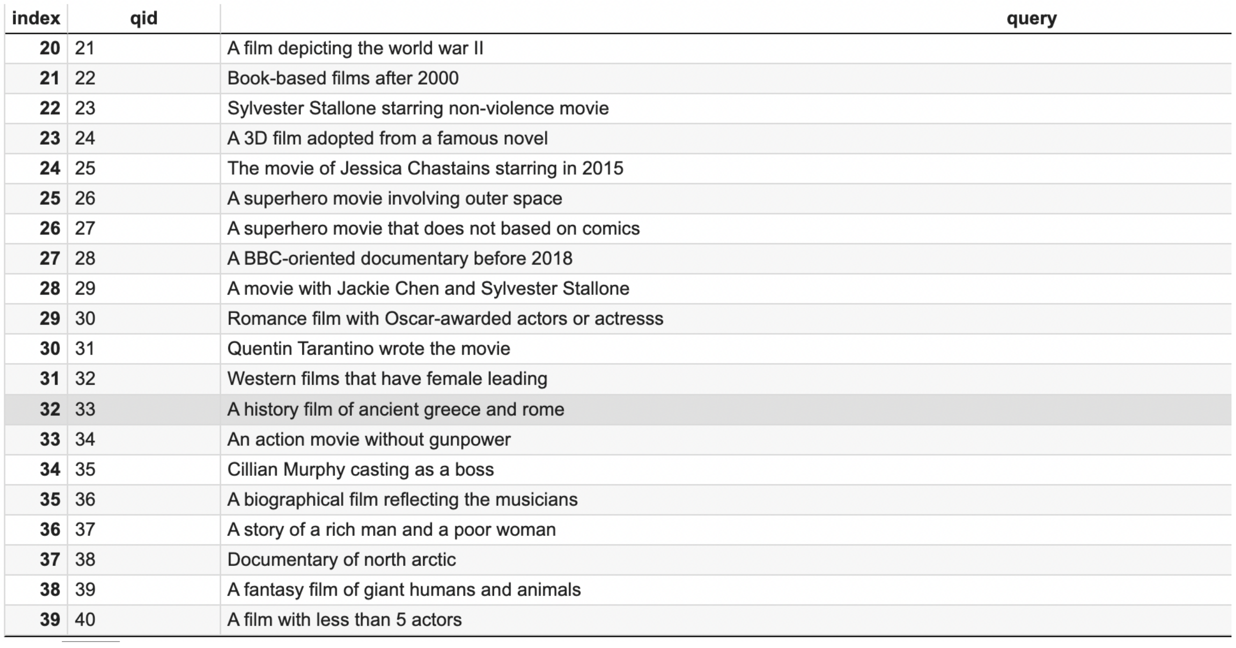


	imdbid	movie_title	release_date	languages	countries_of_origin	production_companies	runtime	plot_words	taglines	directors	writers	cast	parentguide
0	tt0057603	Black Sabbath	May 6, 1964 (United States)	Italian	Italy, France, United States	Emmepi Cinematografica, Societé Cinématographique Lyre, Galatea Film	1 hour 37 minutes	lesbian subtext, vampire, ring, cigarette smoking, killing a dog, nurse, murder, dog, strangulation, stalker, corpse, ghost, giallo, french giallo...	This is the night of the nightmare... This is the night of the nightmare...The day of the Undead. A story that goes beyond the boundaries of the S...	Mario Bava	Anton Chekhov, Aleksei Tolstoy, Guy de Maupassant, Marcello Fondato, Alberto Bevilacqua, Mario Bava	Michèle Mercier, Lidia Alfonsi, Boris Karloff, Mark Damon, Susy Andersen, Massimo Righi, Rika Dialyna, Glauco Onorato, Jacqueline Pierreux, Milly	The first story a woman gets ready for bed, disrobing from her dress into a night gown. Her bare back and legs are focused on by the camera, refer...
1	tt1733125	Dungeons & Dragons: The Book of Vile Darkness	November 24, 2012 (United States)	English	United Kingdom, Bulgaria	Bomar OOD, Zinc Entertainment Inc.	1 hour 30 minutes	violence, sword and sorcery, dragon, psychotronic film, wyvern, blood, sexuality, direct to video, part of trilogy, sword, male nudity, female nud...	NaN	Gerry Lively	Brian Rudnick, Brian Rudnick	Anthony Howell, Dominic Mafham, Barry Aird, Jack Derges, Eleanor Gecks, Lloyd Pitts, Charlotte Hunter, Kaloian Vodenicharov, Beau Brasseaux, Austi...	There is a short orgy scene involving one male and several topless female characters., A female character seduces a male character while wearing o...
2	tt0033045	The Shop Around the Corner	January 12, 1940 (United States)	English	United States	Metro-Goldwyn-Mayer (MGM)	1 hour 39 minutes	quote from julius caesar, reference to shakespeare, reference to the comedie francaise, reference to crime and punishment the novel, carnation, re...	Just LOOK at WHO GET THAT SLY "LUBITSCH TOUCH" NOW! (Print Ad- Philadelphia Inquirer, ((Philadelphia, Penna.)) 10 January 1940) THE MAGIC "TOUCH" ...	Ernst Lubitsch	Samson Raphaelson, Miklós László, Ben Hecht	Margaret Sullavan, James Stewart, Frank Morgan, Joseph Schildkraut, Sara Haden, Felix Bressart, William Tracy, Inez Courtney, Sarah Edwards, Edwin...	Lighthearted romance. A man's wife is reported to be having an affair with a young man. One VERY mildly suggestive comment about another company's...
3	tt0113862	Mr. Holland's Opus	January 19, 1996 (United States)	English, American Sign Language	United States	Hollywood Pictures, Interscope Communications, Polygram Filmed Entertainment	2 hours 23 minutes	teaching, high school, 1990s, deafness, high school teacher, teacher student relationship, musical education, marching band, dedicated teacher, po...	We are your symphony, Mr. Holland. We are the melodies and the notes of your opus. We are the music of your life. It's not about the direction you...	Stephen Herek	Patrick Sheane Duncan	Richard Dreyfuss, Glenne Headly, Jay Thomas, Olympia Dukakis, William H. Macy, Alicia Witt, Terrence Howard, Damon Whitaker, Jean Louisa Kelly, Al...	Nudity: None, One Scene is a tiny bit sensual, Mr. Holland has lit candles. While laying on the bed he is kissing each one of his wife's fingers d...
4	tt0086250	Scarface	December 9, 1983 (United States)	English, Spanish	United States	Universal Pictures	2 hours 50 minutes	cocaine, gangster, cuban refugee, organized crime, murderer as protagonist, protagonist becomes antagonist, death of protagonist, cuban american, ...	He was Tony Montana. The world will remember him by another name...SCARFACE. The world is yours... He wanted to live the American Dream until the ...	Brian De Palma	Oliver Stone, Howard Hawks, Ben Hecht, Armitage Trail	Al Pacino, Steven Bauer, Michelle Pfeiffer, Mary Elizabeth Mastrantonio, Robert Loggia, Miriam Colon, F. Murray Abraham, Paul Shenar, Harris Yulin...	Some kissing scenes and very mild nudity but no sex scenes. The closest we get is a man and a woman in bed and snuggling, with some female nudity....

The queries are below:

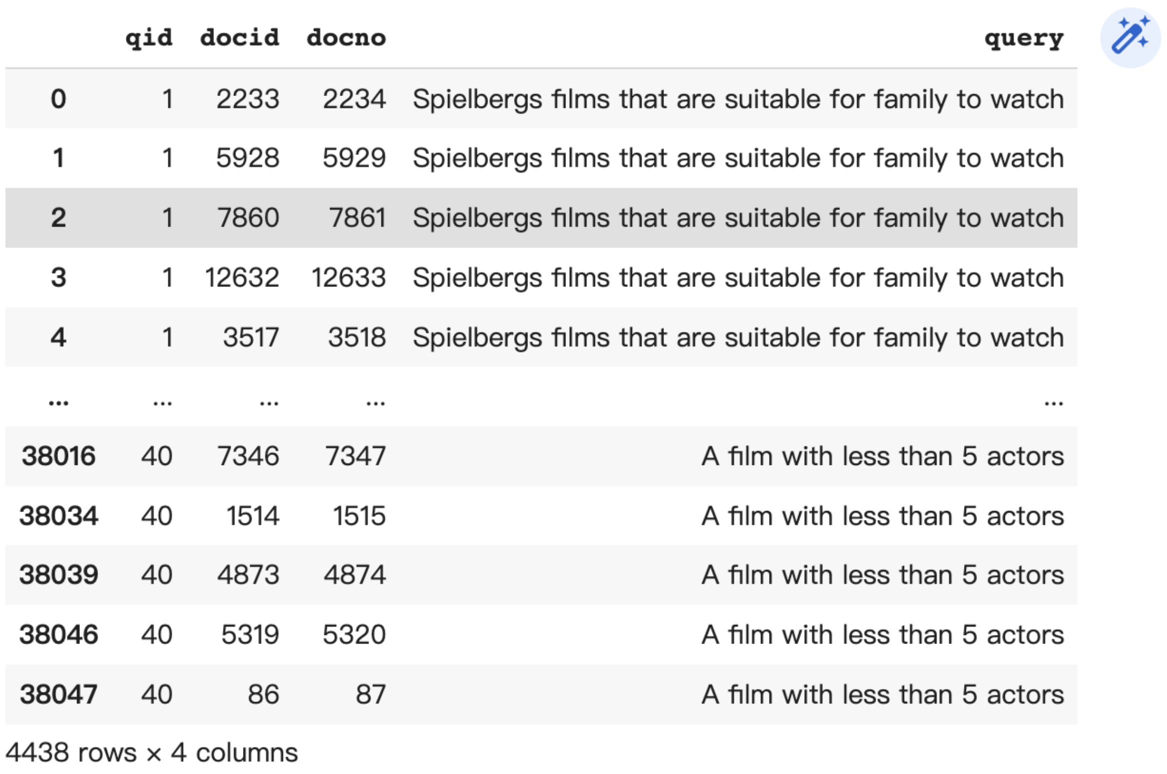


index	qid	query
0	1	Spielbergs films that are suitable for family to watch
1	2	Oscar winning war films
2	3	Films that reflect Russian history
3	4	History films that have big budgets
4	5	Films depicting the modern history of China
5	6	A romantic comedy starring Zooey Deschanel
6	7	A small budget independent film directed by Wes Anderson
7	8	After entering Hollywood Jackie Chan starred in the action comedy film
8	9	A science fiction blockbuster reflecting the future war between humans and robots
9	10	Musical film adaptations of world-famous books
10	11	Suitable for couples to watch the romantic type of large production film
11	12	Ridley Scotts historical film about the Crusades
12	13	Ridley Scott directed the horror science fiction film
13	14	A fantasy drama film inspired by Zweigs novel
14	15	A biographical history film reflecting the career of a scientist
15	16	Martin Scorsese and Robert De Niro collaborated on the film
16	17	The first montage in the history of cinema directed by a Soviet director
17	18	A tangled love story set in the battlefields of North Africa during World War II
18	19	Forbidden love story of middle-aged man and teenage girl
19	20	Classic Middle Eastern films from the First World War
20	21	A film depicting the world war II



index	qid	query
20	21	A film depicting the world war II
21	22	Book-based films after 2000
22	23	Sylvester Stallone starring non-violence movie
23	24	A 3D film adopted from a famous novel
24	25	The movie of Jessica Chastains starring in 2015
25	26	A superhero movie involving outer space
26	27	A superhero movie that does not based on comics
27	28	A BBC-oriented documentary before 2018
28	29	A movie with Jackie Chen and Sylvester Stallone
29	30	Romance film with Oscar-awarded actors or actresss
30	31	Quentin Tarantino wrote the movie
31	32	Western films that have female leading
32	33	A history film of ancient greece and rome
33	34	An action movie without gunpower
34	35	Cillian Murphy casting as a boss
35	36	A biographical film reflecting the musicians
36	37	A story of a rich man and a poor woman
37	38	Documentary of north arctic
38	39	A fantasy film of giant humans and animals
39	40	A film with less than 5 actors

We then utilized the BM25, TFIDF, DFIZ, and DirichletLM_df to search the queries in our documents and selected the top 70 documents from each model, and kept the uniqueness of them. Then, we got a 4438 rows table to annotate the relevance.

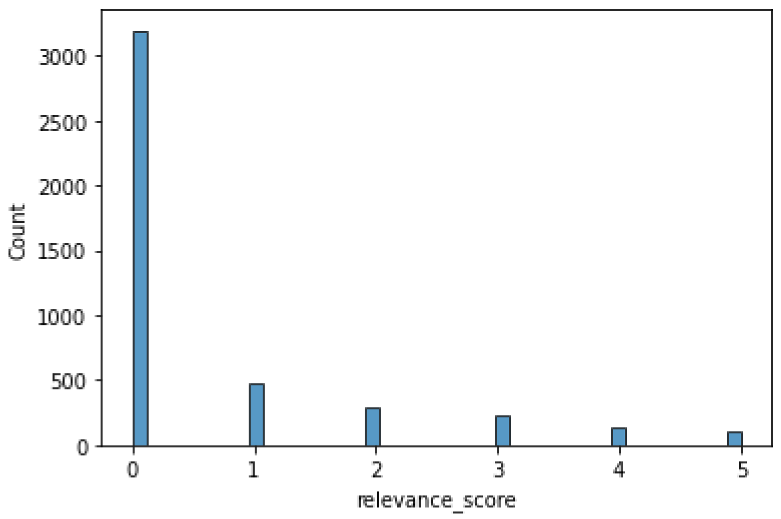


	qid	docid	docno	query
0	1	2233	2234	Spielbergs films that are suitable for family to watch
1	1	5928	5929	Spielbergs films that are suitable for family to watch
2	1	7860	7861	Spielbergs films that are suitable for family to watch
3	1	12632	12633	Spielbergs films that are suitable for family to watch
4	1	3517	3518	Spielbergs films that are suitable for family to watch
...	...	...	...	...
38016	40	7346	7347	A film with less than 5 actors
38034	40	1514	1515	A film with less than 5 actors
38039	40	4873	4874	A film with less than 5 actors
38046	40	5319	5320	A film with less than 5 actors
38047	40	86	87	A film with less than 5 actors

4438 rows × 4 columns

When labeling the documents, we added a column of the IMDB address of the film for us to refer to. Here are the labeling thresholds:

- 5: perfectly matches the query
- 4: almost matches the query, with 80% similarity and above
- 3: partially matches the query, for example, for the query “Sylvester Stallone starring non-violence movie”, the movie “Rambo: First Blood Part II” was scored 3 since it has Sylvester Stallone starring but not meets the non-violence.
- 2: meets one condition in the query, for example, for the query “Book-based films after 2000”, “Death Race 2000” was scored 2 since it is a story-based movie before 2000. The score is for its relevance between book-based and story-based.
- 1: hardly matches the query such as genre, actor or actress, or date, for example, for the query “A BBC-oriented documentary before 2018”, we scored “Titicut Follies” 1 since it only meets the condition as “documentary”.
- 0: completely irrelevant.



Here is a histogram of the relevance score in total. We can see that the majority of the relevance score is 0, and the higher the score is, the smaller the number is.

Related Work

Jingjing Liu et al. mentioned that semantic language understanding has been a difficult problem for dialogue systems (2012). Many systems use context-free grammar for sentence parsing and semantic understanding. Because CFG-based language understanding is controlled by experts, it maintains high accuracy on a closed corpus. However, the coverage of the grammar is limited, and parsing is poor for extra-lexical words and spelling errors, as well as unseen language patterns. And developing a CFG requires a lot of expert knowledge and manpower; thus it is not easily scalable to larger datasets or generalized to other domains. Therefore they have adopted a data-driven approach. There are many researchers who use statistical methods such as conditional random fields and semi-conditional random fields for language understanding. For a query, this method can first perform CRF tagging to group the corresponding phrases in the query into corresponding metadata, such as whether the information belongs to genre, plot, or actor. They then generate topic clusters using topic modeling such as Latent Dirichlet Allocation. These elements can be learned from movie reviews and plots. The topic clusters include synonyms, near-synonyms, and related words that can be used for query expansion. For words that are easily misspelled, such as people's names, the researchers used the Lucene search engine to convert query words into standard phonemes, which are queried by pronunciation rather than spelling. (Most of the time consonant pronunciations and corresponding letters are more stable, while vowels tend to vary more widely.) The researchers used the crowdsourcing platform Amazon Mechanical Turk to automatically collect large-scale annotated data for CRF training and evaluation. This study does something very similar to our project in that it is a method of querying movies based on a sentence or a few phrases. The difference is that their study uses a CRF parser, which would segment the query constituents and assign a semantic class to each segment. Therefore their query may be more concrete and structured and can correspond well with the existing metadata of the movie, with less abstract information about the movie plot, characterization, and viewing experience. In addition, they used topic clustering in the query expansion section, where the vocabulary was learned from texts such as movie reviews and episodes.

Hyung W. Kim et al. investigated how to use user reviews for personalized movie searches (2012). The background of the study is also based on the fact that most movie searches on the market today are based on movie metadata, and more complex information, such as the plot, is not supported by the search system. The second context is that most search results are not personalized, yet different people have different biases in understanding the words in the same query and therefore may also expect not exactly the same returned results. The traditional solution is either manual sorting with a lot of extra manpower or constructing a user profile by filling in the user's personal preferences at the time of registration. The final solution proposed in this study consists of the following four main elements: a user analysis module, a query extension module, a movie indexer, and a movie ranking score calculation module. and a movie ranking score calculation module. The User Analytics module extracts a single user's favorite user comments from their past terms. The module analyzes these comments in order to find the most relevant keywords related to the given query. The query expansion module expands the given query with the keywords obtained from the user analysis module. query with the keywords obtained from the user analysis module. The module selects one or more terms for query expansion, starting with the most common term. This study is similar to ours in that both utilize movie reviews. The difference is that the problems addressed are much different from ours. Their problem is similar to that of a recommender system, focusing on personalized search with movie reviews. Our project, on the other hand, focuses on providing more general and abstract information about movies through movie reviews.

Priya et al. proposed a semantic query proposal for ontology-based movie search (2013). An ontology is a "formal, explicit shared conceptualization of a specification that is useful for improving the accuracy of web search. Ontologies can bridge the gap between query terms and documents through semantic mechanisms. In the framework they designed, if a user enters a query in the search engine's user interface enters a query, then that query will be mapped and projected onto the ontology, and concepts related to the user's query are generated. The network constructed by the ontology contains various concepts, such as movie, genre, director, actor, song, etc. There are various relationships between them, such as a movie is directed by a certain director, starred by a certain actor, and contains a certain song. This information can be used as material for query expansion. This study also gives us some insight into a query expansion approach. However, this study does not indicate how to collect data and how to train and build this ontology network.

Xin Wang et al. used knowledge graphs for movie searches and introduced the MSeeker system (2017). Knowledge graphs use structured data to organize rich information and therefore can provide efficient answers to user queries.MSeeker identifies named entities in queries as the natural language through conditional random fields and determines the relationships between entities. This process is done by Query Interpreter. Query Engine performs query evaluation with an early termination algorithm to save resources and reduce unnecessary computations. This method is very different from the one we used. We use a query based on indices such as BM2.5, where most of the data are unstructured data. Their knowledge graph approach, on the other hand, is highly structured and requires a lot of organization of the raw data to form an entire data network. Also, the knowledge graph approach is difficult to allow for more abstract information, such as plot, etc.

Dongling Chen et al. present iMovie, a prototype sentiment search engine for movie reviews (2015). iMovie utilizes a semi-supervised ontology tree model to represent the complex relationships between features and opinion words in movie reviews. The proposed ontology tree model can guarantee the accuracy of mining sentiment in reviews. The ontology tree model for movie reviews has three layers, namely the movie root, aspects, and feature opinion layers, which are learned in a semi-supervised manner. In order to learn this model, first several specific aspects are defined, such as music, story, performance, etc. Secondly, features, opinion words and their pairings in the review training set need to be manually labeled. This method is quite different from our project and has a very different purpose, but the approach to sentiment analysis can be learned.

Methods

Packages we used: Pandas, NumPy, PyTerrier, train_test_split, FastRank, LightGBM, RandomForestRegressor

We began by preprocessing the original dataset. First, we merged the original dataset with the review data scrapping from imdb. Then, we converted all used variables into string, created doc number and dropped null values.

We started out with features that contained only a synopsis of each movie. After a prompt from GSI, we added the first five reviews of each movie and the corresponding titles. These written movie reviews are obtained by web scraping. After we identified the queries, we realized that we needed more features, because our queries contain a lot of information about actors, directors, and other similar metadata, which are rarely represented in movie reviews. We used web scraping again to obtain these features, including language, country, release date, director, actor, screenwriter, taglines, production company, and parent guides. A number of films were missing a few of these features. In the next step, we will compare the models to determine which features to use.

In these two web scraping, we used the same movies and the same IMDbids. We merged the two data by their IMDbids, and use this data to do information retrieval. Then, we begin to create pipelines.

First, we tried to add all 5 reviews' BM25 scores together, that is to combine them into one feature. This model also contains other features such as the sequential dependence model, scored by BM25, and the coordinate match score for the query--i.e. how many query terms appear.

Second, we tried to add 5 new features, which is whether the word count for every one of the 5 reviews exceeds 300. The reason why to choose 300 as the threshold is that all five reviews have about 300 words on average. If the review contains more than 300 words, add more score; if the review contains less than 300 then no added score.

We also tried many different features, which are elaborated on in the Other Things Tried section, but one of these methods worked. We think this is because there is something wrong with our data.

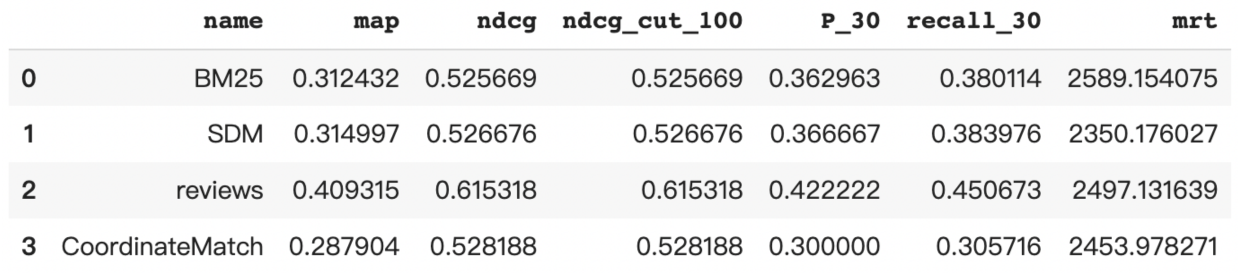
After selecting features, we built our pipelines. We implemented a Random Forest classifier using 400 trees and requiring a minimum of 2 items per leaf. We also build other pipelines by fast rank and LGBM ranker. We may build 7 or 8 feature unions by adding scores to certain columns like review contents or tags and adding scores to some vocabulary(we have not detected the pattern yet). We think that a major problem would be to add appropriate features because we have many queries to search for, so it is inappropriate to add scores for certain vocabulary. We tried coordinate ascent form fast rank, LightGBM, and random forest.

Evaluation and Results

After evaluating the performance of these models, we found that the first pipeline works well, but the other three pipelines have incredibly low performance.

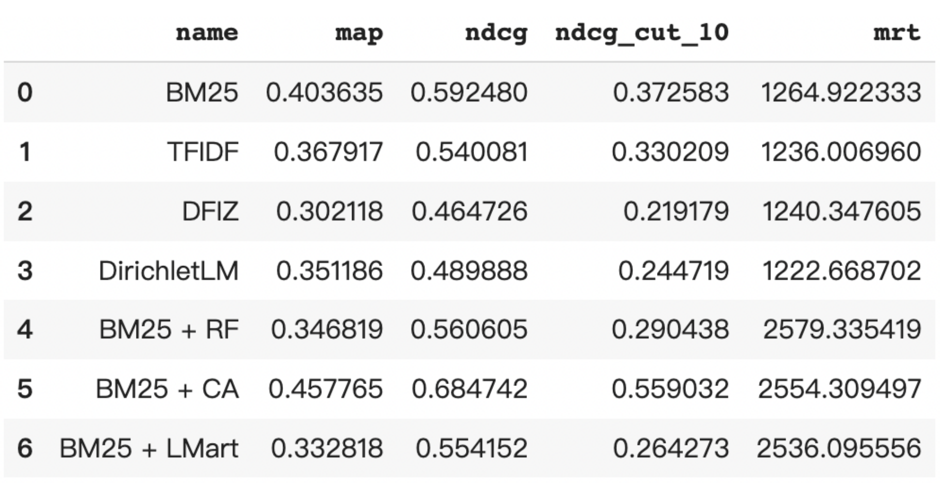
The first pipeline:
The first pipeline includes features such as the sequential dependence model, scored by BM25, BM25 score of reviews, and the coordinate match score for the query--i.e. how many query terms appear

The performance of every single feature:



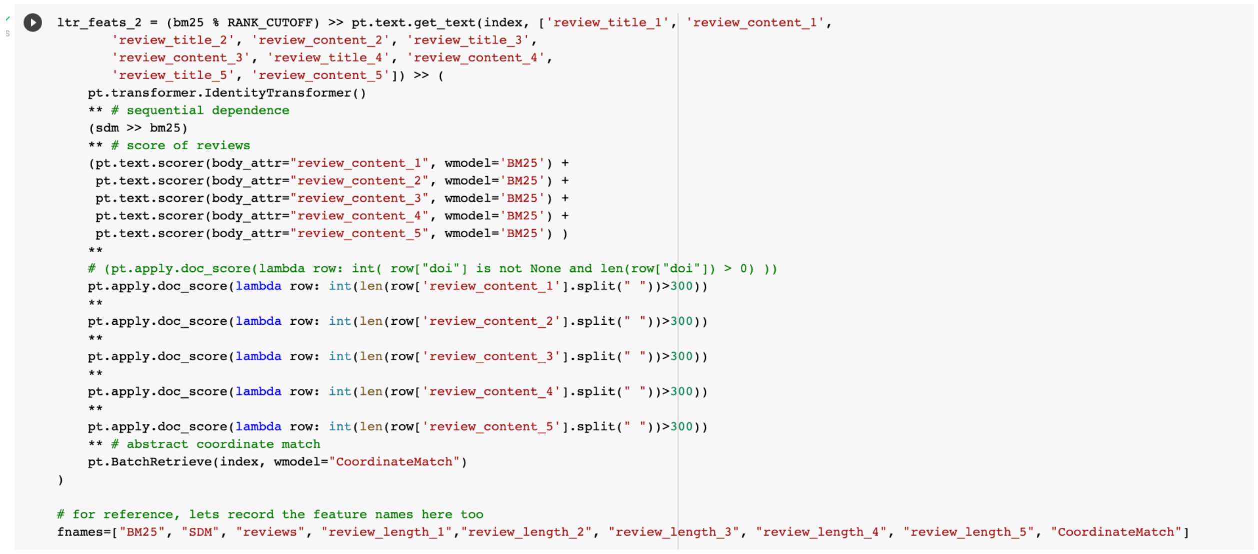
	name	map	ndcg	ndcg_cut_100	P_30	recall_30	mrt
0	BM25	0.312432	0.525669	0.525669	0.362963	0.380114	2589.154075
1	SDM	0.314997	0.526676	0.526676	0.366667	0.383976	2350.176027
2	reviews	0.409315	0.615318	0.615318	0.422222	0.450673	2497.131639
3	CoordinateMatch	0.287904	0.528188	0.528188	0.300000	0.305716	2453.978271

The performance of these features using different machine learning models:



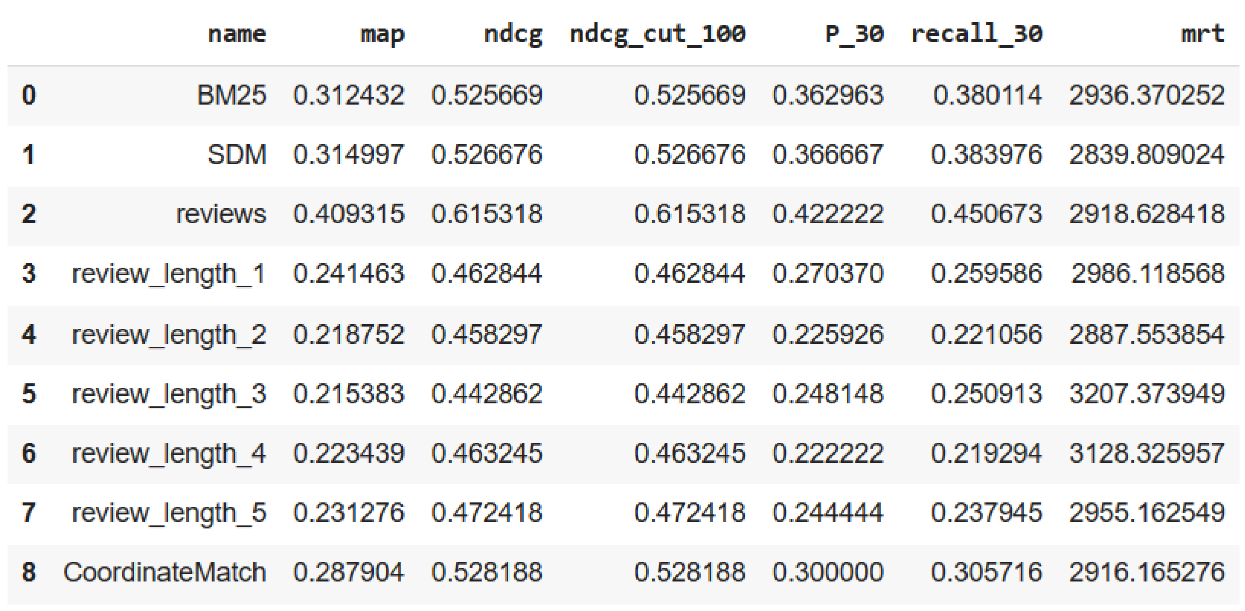
	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.403635	0.592480	0.372583	1264.922333
1	TFIDF	0.367917	0.540081	0.330209	1236.006960
2	DFIZ	0.302118	0.464726	0.219179	1240.347605
3	DirichletLM	0.351186	0.489888	0.244719	1222.668702
4	BM25 + RF	0.346819	0.560605	0.290438	2579.335419
5	BM25 + CA	0.457765	0.684742	0.559032	2554.309497
6	BM25 + LMart	0.332818	0.554152	0.264273	2536.095556

The second pipeline:



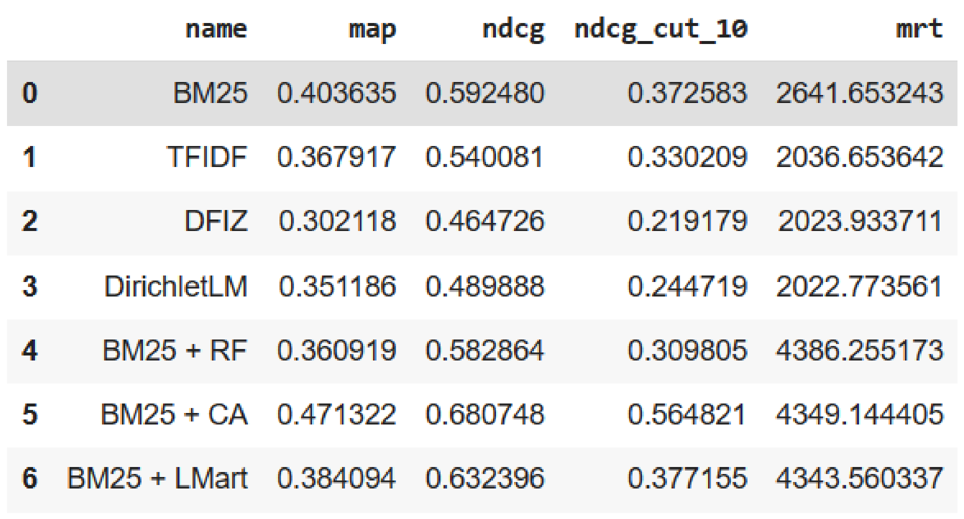
```
ltr_feats_2 = (bm25 % RANK_CUTOFF) >> pt.text.get_text(index, ['review_title_1', 'review_content_1',
        'review_title_2', 'review_content_2', 'review_title_3',
        'review_content_3', 'review_title_4', 'review_content_4',
        'review_title_5', 'review_content_5']) >> (
    pt.transformer.IdentityTransformer()
    ** # sequential dependence
    (sdm >> bm25)
    ** # score of reviews
    (pt.text.scorer(body_attr="review_content_1", wmodel='BM25') +
     pt.text.scorer(body_attr="review_content_2", wmodel='BM25') +
     pt.text.scorer(body_attr="review_content_3", wmodel='BM25') +
     pt.text.scorer(body_attr="review_content_4", wmodel='BM25') +
     pt.text.scorer(body_attr="review_content_5", wmodel='BM25') )
    **
    # (pt.apply.doc_score(lambda row: int( row["doi"] is not None and len(row["doi"]) > 0) ))
    pt.apply.doc_score(lambda row: int(len(row['review_content_1'].split(" "))>300))
    **
    pt.apply.doc_score(lambda row: int(len(row['review_content_2'].split(" "))>300))
    **
    pt.apply.doc_score(lambda row: int(len(row['review_content_3'].split(" "))>300))
    **
    pt.apply.doc_score(lambda row: int(len(row['review_content_4'].split(" "))>300))
    **
    pt.apply.doc_score(lambda row: int(len(row['review_content_5'].split(" "))>300))
    ** # abstract coordinate match
    pt.BatchRetrieve(index, wmodel="CoordinateMatch")
)

# for reference, lets record the feature names here too
fnames=["BM25", "SDM", "reviews", "review_length_1","review_length_2", "review_length_3", "review_length_4", "review_length_5", "CoordinateMatch"]
```

The second pipeline includes features such as the sequential dependence model, scored by BM25, BM25 score of reviews, and the coordinate match score for the query--i.e. how many query terms appear, and 5 new features: whether the every one of the 5 reviews has more than 300 words. The performance of every single feature:



	name	map	ndcg	ndcg_cut_100	P_30	recall_30	mrt
0	BM25	0.312432	0.525669	0.525669	0.362963	0.380114	2936.370252
1	SDM	0.314997	0.526676	0.526676	0.366667	0.383976	2839.809024
2	reviews	0.409315	0.615318	0.615318	0.422222	0.450673	2918.628418
3	review_length_1	0.241463	0.462844	0.462844	0.270370	0.259586	2986.118568
4	review_length_2	0.218752	0.458297	0.458297	0.225926	0.221056	2887.553854
5	review_length_3	0.215383	0.442862	0.442862	0.248148	0.250913	3207.373949
6	review_length_4	0.223439	0.463245	0.463245	0.222222	0.219294	3128.325957
7	review_length_5	0.231276	0.472418	0.472418	0.244444	0.237945	2955.162549
8	CoordinateMatch	0.287904	0.528188	0.528188	0.300000	0.305716	2916.165276

The performance of these features using different machine learning models:

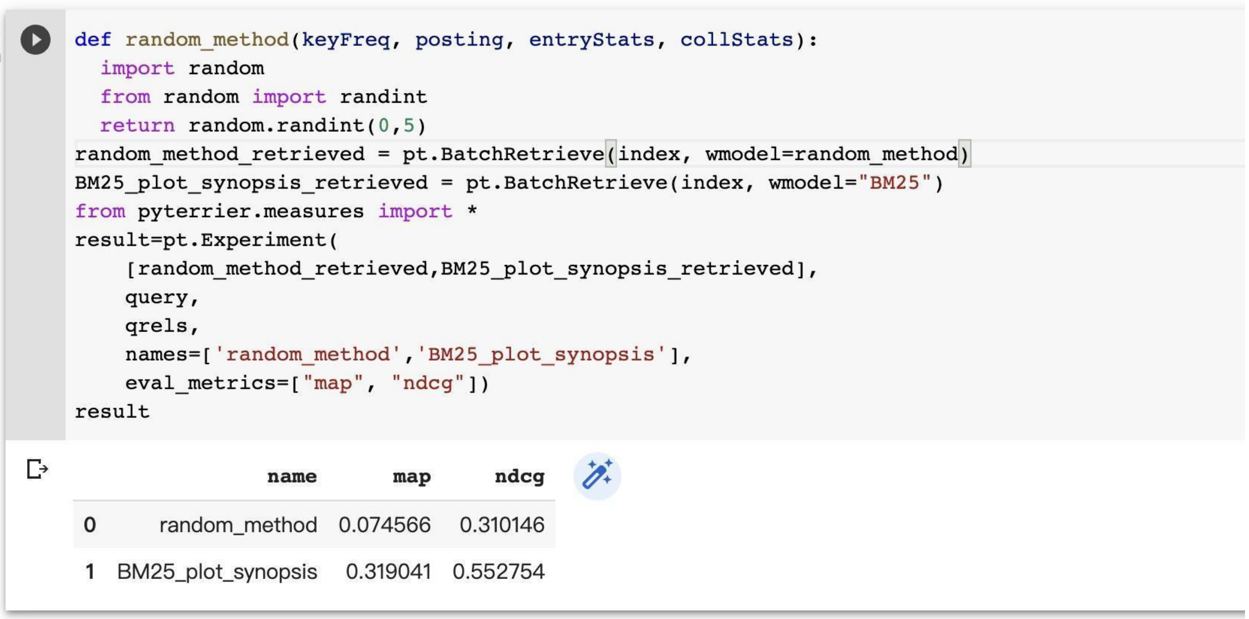


	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.403635	0.592480	0.372583	2641.653243
1	TFIDF	0.367917	0.540081	0.330209	2036.653642
2	DFIZ	0.302118	0.464726	0.219179	2023.933711
3	DirichletLM	0.351186	0.489888	0.244719	2022.773561
4	BM25 + RF	0.360919	0.582864	0.309805	4386.255173
5	BM25 + CA	0.471322	0.680748	0.564821	4349.144405
6	BM25 + LMart	0.384094	0.632396	0.377155	4343.560337

We can see that the combined BM25 score of reviews has very high ndcg. The coordinate ascent pipeline works exceptionally well.

We used two baselines here, the first one is random performance, which ranks the documents from the most relevant to the least relevant using the random method, without looking at any score. The second baseline is the BM2.5 score based only on plot_synopsis. We can see that the random method has a very low ndcg, which indicates it is not a good method. And BM2.5 based on plot_synopsis has a relatively high ndcg.

Here are the codes and results of the performance of these two baseline methods:



```
def random_method(keyFreq, posting, entryStats, collStats):
  import random
  from random import randint
  return random.randint(0,5)
random_method_retrieved = pt.BatchRetrieve(index, wmodel=random_method)
BM25_plot_synopsis_retrieved = pt.BatchRetrieve(index, wmodel="BM25")
from pyterrier.measures import *
result=pt.Experiment(
    [random_method_retrieved,BM25_plot_synopsis_retrieved],
    query,
    qrels,
    names=['random_method','BM25_plot_synopsis'],
    eval_metrics=["map", "ndcg"])
result
```

	name	map	ndcg
0	random_method	0.074566	0.310146
1	BM25_plot_synopsis	0.319041	0.552754

We can see that even the random method performs with a relatively high ndcg. Our model performance far exceeds the two baselines, especially the CA pipeline. I think this is because we add reviews as the feature.

Discussion

Based on the performance of our "short" data, the performance improved a little after we added the lengths of reviews as a feature. However, the small improvement results in more running time. We think that it might not be worth spending more time on such improvement. Therefore, we decided that it is important to have enough features to try for creating pipelines. Then, we decided to add more data to bring us more possible features.

For the two models trained using the first data only containing reviews, the performance is good. Its performance is higher than the two baselines. We tried to search some queries using this model, and the returned movies are satisfactory, though not perfect because our dataset is very limited. The coordinate ascent pipeline from fast rank has significantly better performance than the baselines. I think this is because we added reviews as a feature, which tells a lot of the movie's information.

According to our pipelines and performance, the pipelines built by new data have an extremely low-performance score despite which features we added. We speculated that the reason is that the new data has more empty values. The new data has approximately 14000 records, but after dropping null values, only about 8500 records remained. We attempted to fit the new data with null values and replace them with empty strings, either method led to poor performance.
We also created different pipelines with different features, but the performance still turned out to be poor. We created the same pipeline as we did with the original data, but the same pipeline with new data is very poor. Therefore, we concluded that the problem is within our new data.

Our goal is to make more features into pipelines and then generate different models. However, we should have some understanding of our data and reasonably create indexes and pipelines. The “bigger” data may lead to unexpected consequences.

Conclusion

In this project, we built a vertical search engine to search for movies. We finally used features containing the BM25 index of the first five movie reviews and whether the movie reviews were longer than 300 words. After experiments, we found that the coordinate ascent pipeline in fast rank performed the best. The new feature obtained by adding up the BM25 index of the five movie reviews has the best ndcg performance and is arguably the most important feature. I think in the future we can add more features such as movie plots, keywords, taglines, directors, and so on. We have tried to add these features, but it seems that we can't do it successfully due to the problem of crawling the data ourselves.In the future, a dataset with a larger number of points welcome can be found and other learning-to-rank methods, as well as deep learning models, can be used.

Other Things We Tried

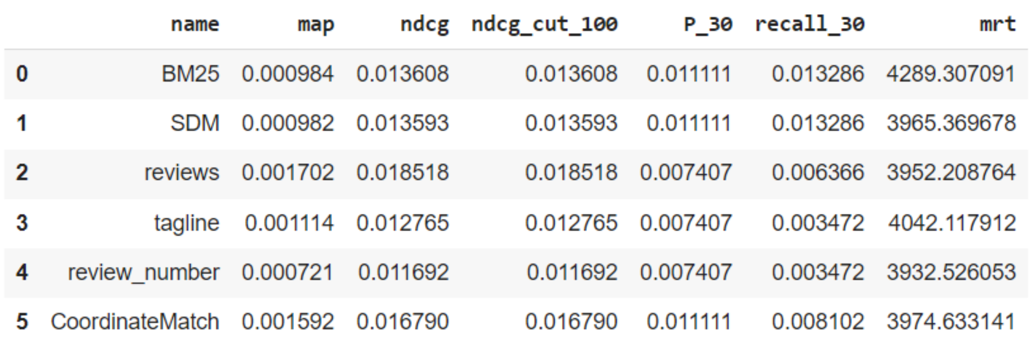
Based on the first pipeline we created, in which we add all 5 reviews’ BM25 scores together, that is to combine them into one feature, we tried the following method which turned out to be not successful.

Then we tried adding two more features: 1. Whether the feature “taglines” is not none, if not none, then add more score to the movie (The score would be 1 if “taglines” is not empty, 0 if “taglines” is empty). 2. Whether the feature “review_content_1” (the content of the first review) is not none, if not none, then add more score to the movie (The score would be 1 if “review_content_1” is not empty, 0 if “review_content_1” is empty).
And then we tried the third model, which is to replace the two features before by the BM25 score of “taglines” and “plot_words”. When doing so, we filled all the none cells with an empty string.

We also tried the fourth model, which is to preprocess the document data frame by concatenating the 5 review string into one string. And then use the BM25 score of the whole string as a feature to replace the feature which is the sum of all 5 BM25 scores of reviews.
All the features performed extremely poorly. We surmise it may be caused by the data we used because one thing we noticed is that when we used the data we scraped first which contained only 5 reviews, the performance is reasonable, but if we changed to the later scraped data, which contains more metadata such as languages, directors, cast, etc., even if we only use the review columns, the performance is incredibly low. We can not fully explain why this is

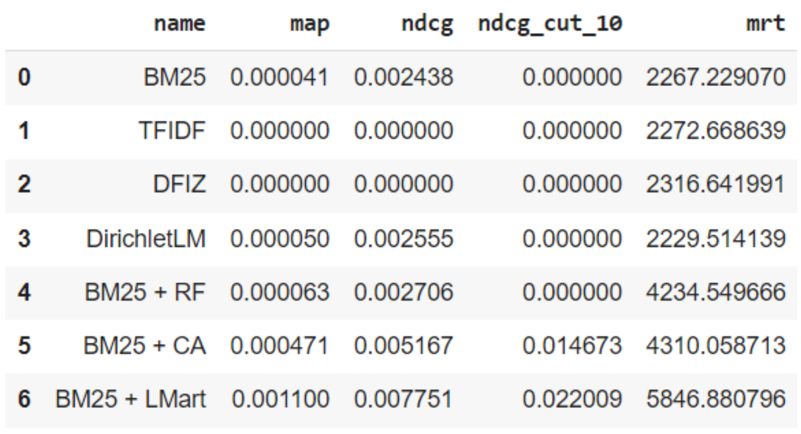
happening, but after many tries, this only happened if we used the second data. We wasted a lot of time because of this error. This may be due to there being many empty data in the later scraped data.

The second pipeline includes features such as the sequential dependence model, scored by BM25, BM25 score of reviews, and the coordinate match score for the query--i.e. how many query terms appear. And there are two more features: 1. Whether the feature “taglines” is not none, if not none, then add more score to the movie (The score would be 1 if “taglines” is not empty, 0 if “taglines” is empty). 2. Whether the feature “review_content_1” (the content of the first review) is not none, if not none, then add more score to the movie (The score would be 1 if “review_content_1” is not empty, 0 if “review_content_1” is empty).
The performance of every single feature:



	name	map	ndcg	ndcg_cut_100	P_30	recall_30	mrt
0	BM25	0.000984	0.013608	0.013608	0.011111	0.013286	4289.307091
1	SDM	0.000982	0.013593	0.013593	0.011111	0.013286	3965.369678
2	reviews	0.001702	0.018518	0.018518	0.007407	0.006366	3952.208764
3	tagline	0.001114	0.012765	0.012765	0.007407	0.003472	4042.117912
4	review_number	0.000721	0.011692	0.011692	0.007407	0.003472	3932.526053
5	CoordinateMatch	0.001592	0.016790	0.016790	0.011111	0.008102	3974.633141

The performance of these features using different machine learning models:



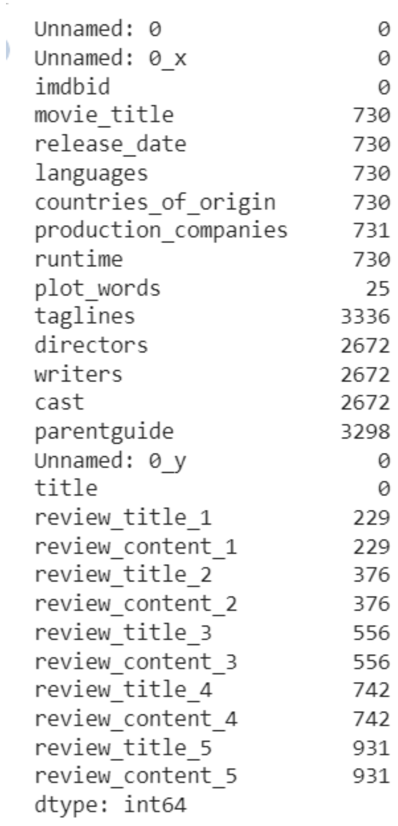
	name	map	ndcg	ndcg_cut_10	mrt
0	BM25	0.000041	0.002438	0.000000	2267.229070
1	TFIDF	0.000000	0.000000	0.000000	2272.668639
2	DFIZ	0.000000	0.000000	0.000000	2316.641991
3	DirichletLM	0.000050	0.002555	0.000000	2229.514139
4	BM25 + RF	0.000063	0.002706	0.000000	4234.549666
5	BM25 + CA	0.000471	0.005167	0.014673	4310.058713
6	BM25 + LMart	0.001100	0.007751	0.022009	5846.880796

Other pipelines have similar performance as this one.

What You Would Have Done Differently or Next

Through this study, we realized that data are important for research. We speculate that the poor performance of the experiment may be due to a large number of missing data in the data we found. At the beginning of the idea, we wanted to make a vertical search engine in the movie search category. So we found the starting dataset, which is the movie data crawled from IMDb, each movie includes the corresponding IMDb ID and the corresponding plot synopsis. This data is complete and does not include movies with missing plot synopses. We would have liked to find a dataset with more features, such as the language, country, and review of the movie, but could not find one. At GSI's suggestion, we crawled through the first five reviews of these movies on IMDb, where the number of missing movies increased from the first to the fifth. Then due to the characteristics of the query we identified, we felt the need to crawl more data and therefore crawled more metadata of the movie. There are more movies missing from these crawls, with 3,000 missing data in items such as taglines.

Pipeline performed better at the beginning when only 5 movie reviews were used. However, when using data with new fields such as taglines, plot_words, etc., the performance of the pipeline becomes poor.



```
Unnamed: 0               0
Unnamed: 0_x             0
imdbid                   0
movie_title            730
release_date           730
languages              730
countries_of_origin    730
production_companies   731
runtime                730
plot_words              25
taglines              3336
directors             2672
writers               2672
cast                  2672
parentguide           3298
Unnamed: 0_y             0
title                    0
review_title_1         229
review_content_1       229
review_title_2         376
review_content_2       376
review_title_3         556
review_content_3       556
review_title_4         742
review_content_4       742
review_title_5         931
review_content_5       931
dtype: int64
```

Here you can see that there is a large amount of empty data for most of the metadata. If dropping all rows that contain empty data, we get only 8000 data. We believe it is important to take longer at the beginning to define the content and data of the project to ensure a higher-quality dataset.

We believe there may be something wrong with this dataset. We have scraped two pieces of data from IMDb. This first one only contains reviews, and the second one contains reviews and other metadata. When using the first data, the performance is reasonable, but when using the second data and adding features such as taglines, and plot_words, the performance became very low.

Also, in the labeling phase, we found that our dataset may be too small. There are about 14000 movies in the dataset. When we labeled the documents for our queries, we found that many queries have no matching in the dataset. For example, one query is “independent films directed by Wes Anderson”, but there was not any film directed by Wes Anderson. Next time we might try to find a larger dataset.
From the related works we have read, we realize that query expansion is necessary. We use query expansion tools or other query rewriting methods to rewrite queries in the future. The sequential dependence proximity model could be used just by inputting queries and getting new queries, and Bo1QueryExpansion could be used by inputting some returned documents to get new queries.

Reference

Chen, Dongling, and Feng Yang. "iMovie: A Prototype Sentiment Search Engine for Movie Reviews." *2015 International Conference on Mechatronics, Electronic, Industrial and Control Engineering (MEIC-15)*. Atlantis Press, 2015.
Kim, Hyung W., et al. "Moviemine: personalized movie content search by utilizing user comments." *IEEE Transactions on Consumer Electronics* 58.4 (2012): 1416-1424.
Liu, Jingjing, et al. "A conversational movie search system based on conditional random fields." *Thirteenth Annual Conference of the International Speech Communication Association*. 2012.
Priya, P., and R. Rajalaxmi. "Ontology based semantic query suggestion for movie search." *2013 International Conference on Information Communication and Embedded Systems (ICICES)*. IEEE, 2013.
Wang, Xin, et al. "MovieFinder: A Movie Search System via Graph Pattern Matching." *EDBT*. 2017.