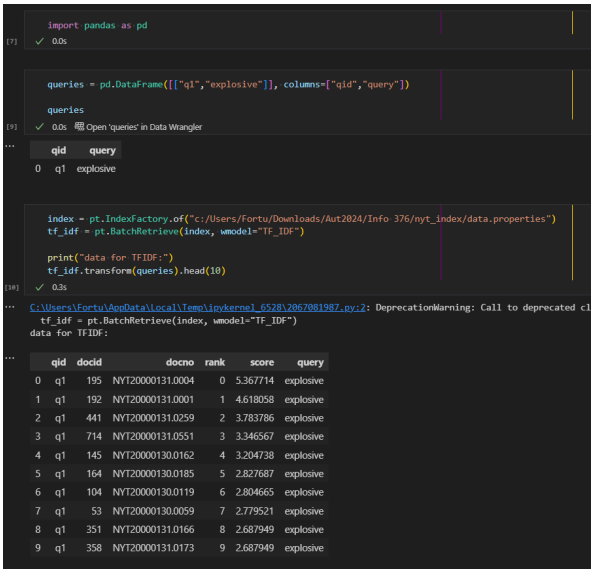
1. Explain how language models work for IR in your own words. Why are they called 'language models'? Respond in at most half a page. [2 points]
    a. Language models in IR are essentially algorithms that use probability to predict the likelyhood of a document to answer the user's query. These algorithms are called language models because of how they were designed to represent or *model* patterns, structures, and relationships between words in a given language. They learn from large amounts of textual data, hence the language part of their name, to predict how language behaves.
2. Continue working with the NYT index from before. This time running the same query with language models. Report the top 10 results. Do you see any differences between these and the previous two models used? If yes, comment on which one you think (qualitatively) is better. [2 points]
    a. I don't see any differences between Heimstra's language model, the BM25, or the TF_IDF in the top 10 results for the query 'explosive'.



```
import pandas as pd
```
0.0s

```
queries = pd.DataFrame([["q1","explosive"]], columns=["qid","query"])

queries
```
0.0s Open 'queries' in Data Wrangler

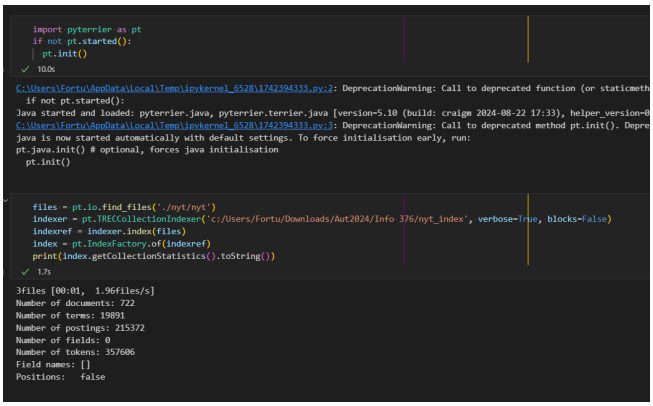
	qid	query
0	q1	explosive

```
index = pt.IndexFactory.of("c:/Users/Fortu/Downloads/Aut2024/Info 376/nyt_index/data.properties")
tf_idf = pt.BatchRetrieve(index, wmodel="TF_IDF")

print("data for TFIDF:")
tf_idf.transform(queries).head(10)
```
0.3s

```
C:\Users\Fortu\AppData\Local\Temp\ipykernel_6528\2067081987.py:2: DeprecationWarning: Call to deprecated cl
  tf_idf = pt.BatchRetrieve(index, wmodel="TF_IDF")
data for TFIDF:
```

	qid	docid	docno	rank	score	query
0	q1	195	NYT20000131.0004	0	5.367714	explosive
1	q1	192	NYT20000131.0001	1	4.618058	explosive
2	q1	441	NYT20000131.0259	2	3.783786	explosive
3	q1	714	NYT20000131.0551	3	3.346567	explosive
4	q1	145	NYT20000130.0162	4	3.204738	explosive
5	q1	164	NYT20000130.0185	5	2.827687	explosive
6	q1	104	NYT20000130.0119	6	2.804665	explosive
7	q1	53	NYT20000130.0059	7	2.779521	explosive
8	q1	351	NYT20000131.0166	8	2.687949	explosive
9	q1	358	NYT20000131.0173	9	2.687949	explosive

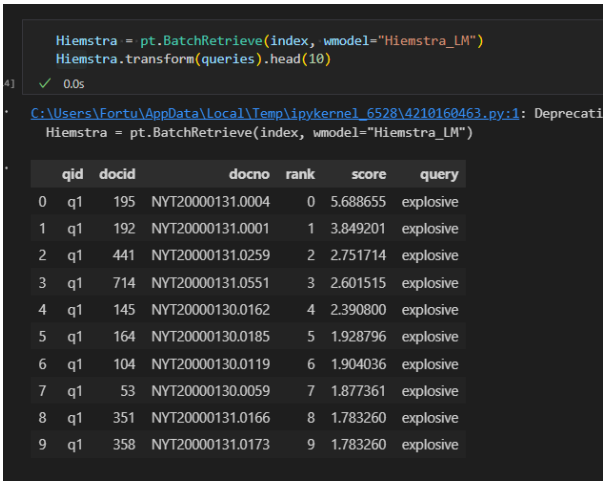


```
import pyterrier as pt
if not pt.started():
  pt.init()
```
10.0s

```
C:\Users\Fortu\AppData\Local\Temp\ipykernel_6528\1742394333.py:2: DeprecationWarning: Call to deprecated function (or staticmeth
  if not pt.started():
Java started and loaded: pyterrier.java, pyterrier.terrier.java [version=5.10 (build: craigm 2024-08-22 17:33), helper_version=0
C:\Users\Fortu\AppData\Local\Temp\ipykernel_6528\1742394333.py:3: DeprecationWarning: Call to deprecated method pt.init(). Depre
java is now started automatically with default settings. To force initialisation early, run:
pt.java.init() # optional, forces java initialisation
  pt.init()
```

```
files = pt.io.find_files('./nyt/nyt')
indexer = pt.TRECCollectionIndexer('c:/Users/Fortu/Downloads/Aut2024/Info 376/nyt_index', verbose=True, blocks=False)
indexref = indexer.index(files)
index = pt.IndexFactory.of(indexref)
print(index.getCollectionStatistics().toString())
```
1.7s

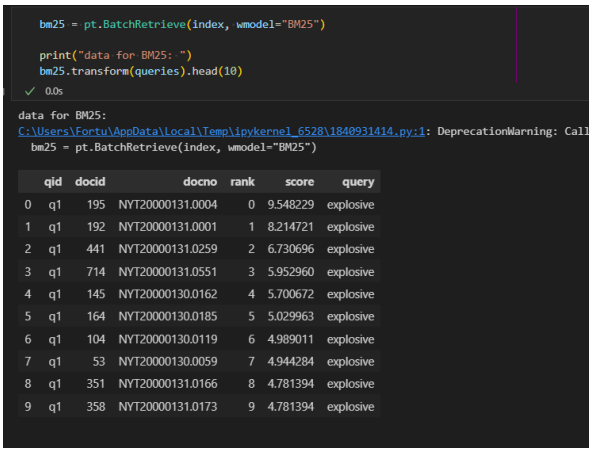
```
3files [00:01,  1.96files/s]
Number of documents: 722
Number of terms: 19891
Number of postings: 215372
Number of fields: 0
Number of tokens: 357606
Field names: []
Positions:   false
```



```
Hiemstra = pt.BatchRetrieve(index, wmodel="Hiemstra_LM")
Hiemstra.transform(queries).head(10)
```
0.0s

```
C:\Users\Fortu\AppData\Local\Temp\ipykernel_6528\4210160463.py:1: Deprecati
  Hiemstra = pt.BatchRetrieve(index, wmodel="Hiemstra_LM")
```

	qid	docid	docno	rank	score	query
0	q1	195	NYT20000131.0004	0	5.688655	explosive
1	q1	192	NYT20000131.0001	1	3.849201	explosive
2	q1	441	NYT20000131.0259	2	2.751714	explosive
3	q1	714	NYT20000131.0551	3	2.601515	explosive
4	q1	145	NYT20000130.0162	4	2.390800	explosive
5	q1	164	NYT20000130.0185	5	1.928796	explosive
6	q1	104	NYT20000130.0119	6	1.904036	explosive
7	q1	53	NYT20000130.0059	7	1.877361	explosive
8	q1	351	NYT20000131.0166	8	1.783260	explosive
9	q1	358	NYT20000131.0173	9	1.783260	explosive



```
bm25 = pt.BatchRetrieve(index, wmodel="BM25")

print("data for BM25: ")
bm25.transform(queries).head(10)
```
0.0s

```
data for BM25:
C:\Users\Fortu\AppData\Local\Temp\ipykernel_6528\1840931414.py:1: DeprecationWarning: Call
  bm25 = pt.BatchRetrieve(index, wmodel="BM25")
```

	qid	docid	docno	rank	score	query
0	q1	195	NYT20000131.0004	0	9.548229	explosive
1	q1	192	NYT20000131.0001	1	8.214721	explosive
2	q1	441	NYT20000131.0259	2	6.730696	explosive
3	q1	714	NYT20000131.0551	3	5.952960	explosive
4	q1	145	NYT20000130.0162	4	5.700672	explosive
5	q1	164	NYT20000130.0185	5	5.029963	explosive
6	q1	104	NYT20000130.0119	6	4.989011	explosive
7	q1	53	NYT20000130.0059	7	4.944284	explosive
8	q1	351	NYT20000131.0166	8	4.781394	explosive
9	q1	358	NYT20000131.0173	9	4.781394	explosive

3. Crawl (run wget on) a website of your choice and collect 50-100 indexable documents (e.g., html, txt, pdf). [2 points]

	qid	docid	docno	rank	score	query
0	q1	57	d58	0	3.126462	phones
1	q1	9	d10	1	1.817748	phones
2	q1	3	d4	2	1.181763	phones
3	q1	62	d63	3	1.069977	phones
4	q1	67	d68	4	0.435833	phones
5	q2	22	d23	0	3.195160	equity
6	q2	43	d44	1	1.841865	equity
7	q2	7	d8	2	0.963619	equity
8	q2	56	d57	3	0.587240	equity
9	q3	46	d47	0	3.070312	reddit
10	q3	37	d38	1	1.672388	reddit
11	q3	28	d29	2	1.667580	reddit
12	q3	12	d13	3	0.795276	reddit
13	q3	67	d68	4	0.068320	reddit
14	q	11	d12	0	0.739097	start-ups
15	q	27	d28	1	0.615168	start-ups
16	q	51	d52	2	0.589969	start-ups
17	q	69	d70	3	0.465846	start-ups
18	q	30	d31	4	0.456731	start-ups
19	q	46	d47	5	0.415946	start-ups
20	q	12	d13	6	0.361403	start-ups
21	q	67	d68	7	0.343771	start-ups
22	q	9	d10	8	0.343491	start-ups
23	q	42	d43	9	0.339634	start-ups
24	q	44	d45	10	0.339374	start-ups
25	q	0	d1	11	0.326315	start-ups
26	q	6	d7	12	0.322113	start-ups

a. I ran wget crawl on a multitude of websites, like Google and Yellow Pages. I did not get the results I desired, so I returned to the website we covered in class, called paulgraham.com.

b. I ran:

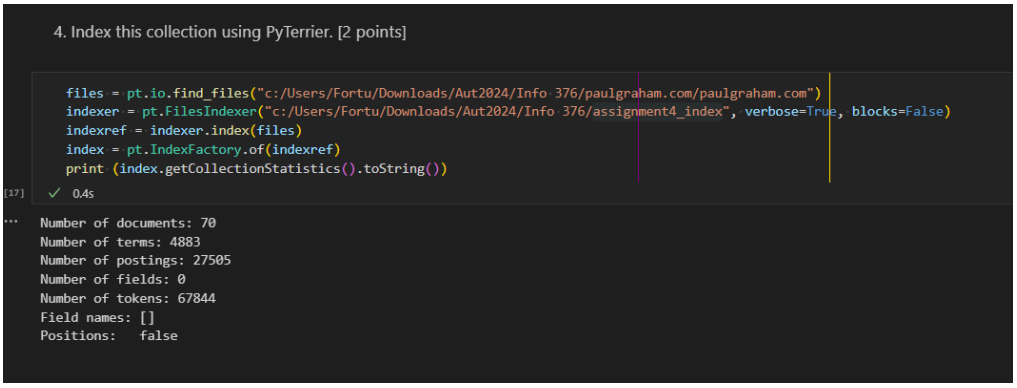
i. wget -r paulgraham.com

c. for my data, but I also tested:

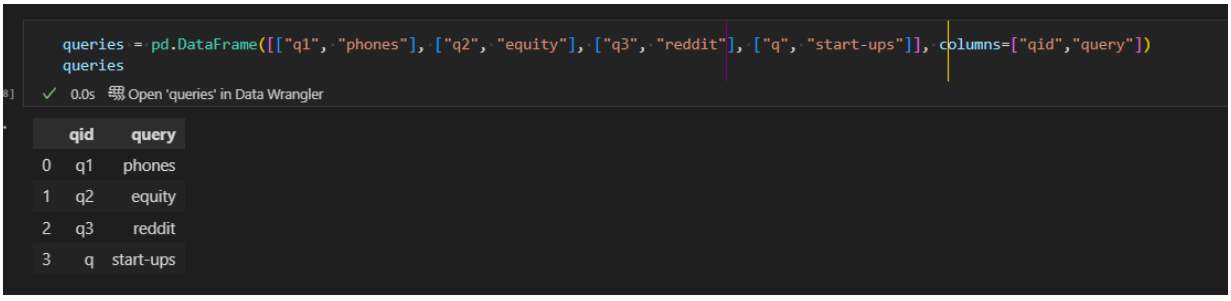
i. wget -r google.com (Got an error saying it was moved permanently, but I got the index.html of it.)

ii. wget -r yellowpages.com (Got the same message for this as I got for Google.com.)

4. Index this collection using PyTerrier. [2 points]



5. Run at least 4 different queries using your choice of retrieval model and show your retrieval results. [2 points]



6. Report your wget command, index statistics, and retrieval results.