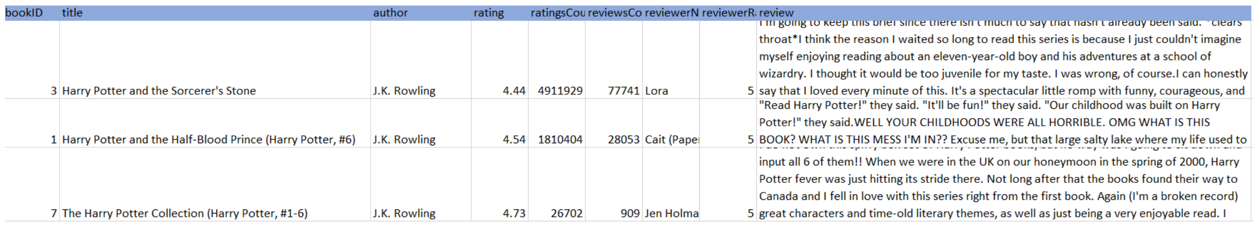
Read and Review Search Engine:

Introduction

Read and Review is a search engine which takes input from the user and provides the top 5 books or novels related to the input.

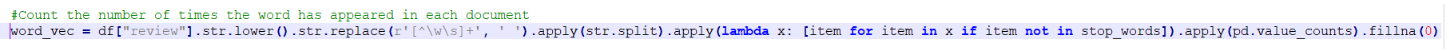
I have used Goodreads dataset for this search engine from Kaggle.com. The Search engine searches for reviews that relate to the keywords or the query the user inputs and then puts the results found in the order of relevance to the topic that was searched for.



bookID	title	author	rating	ratingsCou	reviewsCo	reviewerN	reviewerR	review
3	Harry Potter and the Sorcerer's Stone	J.K. Rowling	4.44	4911929	77741	Lora	5	I'm going to keep this brief since there isn't much to say that hasn't already been said. *clears throat*I think the reason I waited so long to read this series is because I just couldn't imagine myself enjoying reading about an eleven-year-old boy and his adventures at a school of wizardry. I thought it would be too juvenile for my taste. I was wrong, of course.I can honestly say that I loved every minute of this. It's a spectacular little romp with funny, courageous, and
1	Harry Potter and the Half-Blood Prince (Harry Potter, #6)	J.K. Rowling	4.54	1810404	28053	Cait (Pape	5	"Read Harry Potter!" they said. "It'll be fun!" they said. "Our childhood was built on Harry Potter!" they said.WELL YOUR CHILDHOODS WERE ALL HORRIBLE. OMG WHAT IS THIS BOOK? WHAT IS THIS MESS I'M IN?? Excuse me, but that large salty lake where my life used to
7	The Harry Potter Collection (Harry Potter, #1-6)	J.K. Rowling	4.73	26702	909	Jen Holma	5	input all 6 of them!! When we were in the UK on our honeymoon in the spring of 2000, Harry Potter fever was just hitting its stride there. Not long after that the books found their way to Canada and I fell in love with this series right from the first book. Again (I'm a broken record) great characters and time-old literary themes, as well as just being a very enjoyable read. I

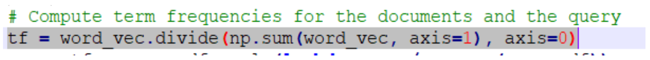
Implementation

Processing the data by removing punctuation and symbols, stop words and tokenizing them and adding them in an index.



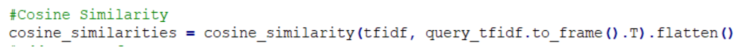
```
#Count the number of times the word has appeared in each document
word_vec = df["review"].str.lower().str.replace(r'[^\w\s]+', ' ').apply(str.split).apply(lambda x: [item for item in x if item not in stop_words]).apply(pd.value_counts).fillna(0)
```

Calculate the tf and idf and tfidf.



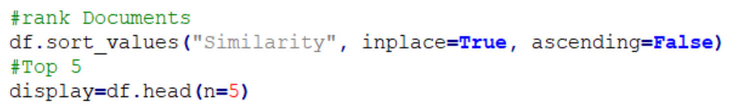
```
# Compute term frequencies for the documents and the query
tf = word_vec.divide(np.sum(word_vec, axis=1), axis=0)
```

Calculate the Cosine similarity between the query and the tfidf.



```
#Cosine Similarity
cosine_similarities = cosine_similarity(tfidf, query_tfidf.to_frame().T).flatten()
```

Rank the documents



```
#rank Documents
df.sort_values("Similarity", inplace=True, ascending=False)
#Top 5
display=df.head(n=5)
```

Wrapped it around with Django and hosted it on PythonAnywhere.

The link: http://sushmitaravip.pythonanywhere.com/Search

References:

https://www.geeksforgeeks.org/counting-the-frequencies-in-a-list-using-dictionary-in-python/

https://www.sharpsightlabs.com/blog/numpy-sum/

https://github.com/williamscott701/Information-Retrieval/blob/master/2.%20TF-IDF%20Ranking%20-%20Cosine%20Similarity%2C%20Matching%20Score/TF-IDF.ipynb

https://stackoverflow.com/questions/18424228/cosine-similarity-between-2-number-lists

http://www.sfs.uni-tuebingen.de/~ddekok/ir/lectures/tf-idf-dump.html

https://stackoverflow.com/questions/12118720/python-tf-idf-cosine-to-find-document-similarity

https://github.com/williamscott701/Information-Retrieval/blob/master/2.%20TF-IDF%20Ranking%20-%20Cosine%20Similarity%2C%20Matching%20Score/TF-IDF.ipynb

https://www.saltycrane.com/blog/2007/10/using-pythons-finditer-to-highlight/

https://stackoverflow.com/questions/50573053/similarity-between-2-columns-of-a-dataframe

https://docs.djangoproject.com/en/2.1/ref/templates/builtins/