Modern Language Tool Kit (MLTK)

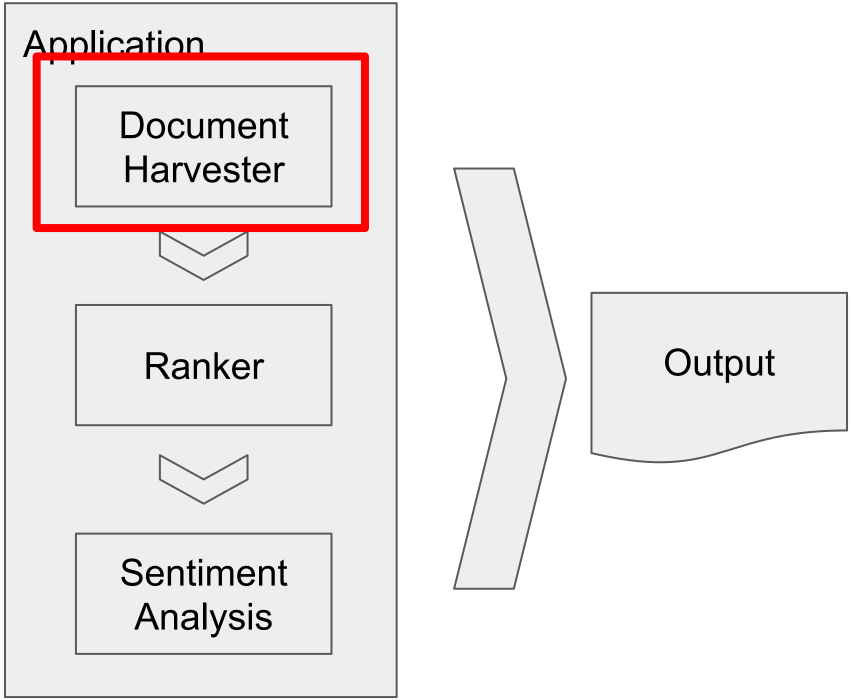
https://github.com/luiswally/MLTK

Document Harvester

TwitterDocumentHarvester.py

Provides a simplified interface to harvest opinionated text documents from Twitter.

- **Input**: User defined query string (e.g., “Hawaii volcano”, “God of War”)
- **Output**: Collection of tweets found on Twitter based on the provided query string.
    - Represented as a text document with each tweet separated by a newline.
- **Additional Functionality**:
    - Tweet filter criteria (e.g., *number of retweets, number of likes*)
- **Dependencies**:
    - Snscrape version 0.4.3.20220106
    - Pandas version 1.5.1

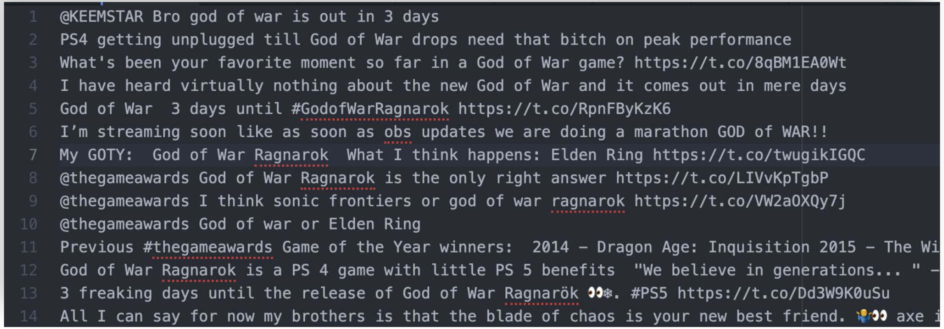


Document Harvester

Available Metadata *as dataframe self.tweets*

- *Id* - Unique identifier of tweet.
- *Date* - Date on which the tweet was created.
- *Username* - Username of the account that created the tweet.
- *Hashtags* - Hashtags included with the tweet.
- *Tweet* - The text associated with the tweet.
- *Likes* - The number of likes associated with the tweet.
- *Retweets* - The number of retweets the tweet has received from other users.

Sample Output *as a text file*

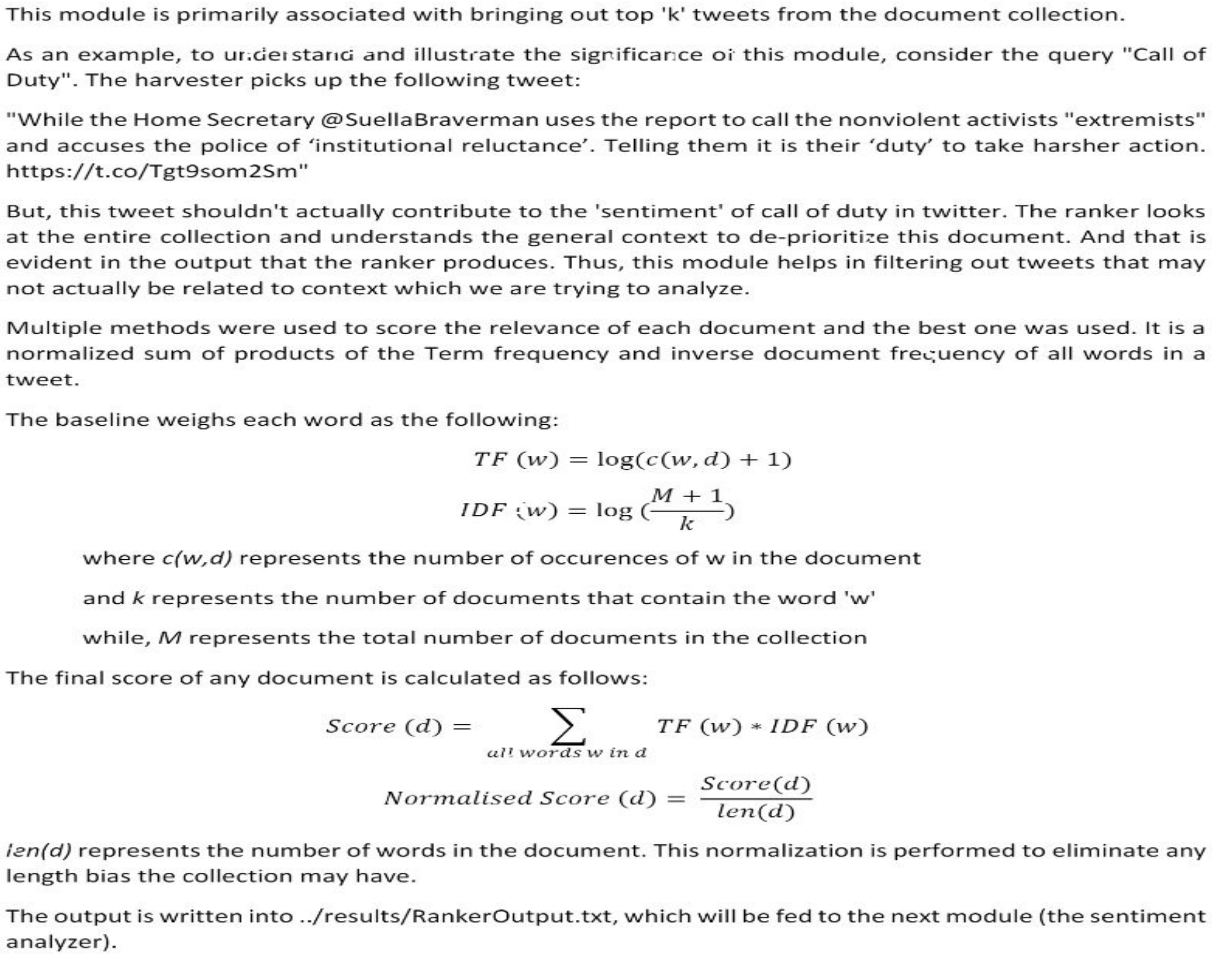


```
@KEEMSTAR Bro god of war is out in 3 days
PS4 getting unplugged till God of War drops need that bitch on peak performance
What's been your favorite moment so far in a God of War game? https://t.co/8qBM1EA0Wt
I have heard virtually nothing about the new God of War and it comes out in mere days
God of War  3 days until #GodofWarRagnarok https://t.co/RpnFByKzK6
I’m streaming soon like as soon as obs updates we are doing a marathon GOD of WAR!!
My GOTY:  God of War Ragnarok  What I think happens: Elden Ring https://t.co/twugikIGQC
@thegameawards God of War Ragnarok is the only right answer https://t.co/LIVvKpTgbP
@thegameawards I think sonic frontiers or god of war ragnarok https://t.co/VW2aOXQy7j
@thegameawards God of war or Elden Ring
Previous #thegameawards Game of the Year winners:  2014 – Dragon Age: Inquisition 2015 – The Wi
God of War Ragnarok is a PS 4 game with little PS 5 benefits  "We believe in generations... " –
3 freaking days until the release of God of War Ragnarök 👀❄️. #PS5 https://t.co/Dd3W9K0uSu
All I can say for now my brothers is that the blade of chaos is your new best friend. 🤷👀 axe i
```

Relevant Links:

- **TwitterDocumentHarvester:** https://github.com/luiswally/MLTK/tree/main/twitterDocumentHarvester
- **SNScrape:** https://github.com/JustAnotherArchivist/snscrape
- **Pandas:** https://pandas.pydata.org/

How does the ranker work



This module is primarily associated with bringing out top 'k' tweets from the document collection.

As an example, to understand and illustrate the significance of this module, consider the query "Call of Duty". The harvester picks up the following tweet:

"While the Home Secretary @SuellaBraverman uses the report to call the nonviolent activists "extremists" and accuses the police of ‘institutional reluctance’. Telling them it is their ‘duty’ to take harsher action. https://t.co/Tgt9som2Sm"

But, this tweet shouldn't actually contribute to the 'sentiment' of call of duty in twitter. The ranker looks at the entire collection and understands the general context to de-prioritize this document. And that is evident in the output that the ranker produces. Thus, this module helps in filtering out tweets that may not actually be related to context which we are trying to analyze.

Multiple methods were used to score the relevance of each document and the best one was used. It is a normalized sum of products of the Term frequency and inverse document frequency of all words in a tweet.

The baseline weighs each word as the following:

$$TF\ (w) = \log(c(w,d) + 1)$$

$$IDF\ (w) = \log\ (\frac{M+1}{k})$$

where *c(w,d)* represents the number of occurences of w in the document

and *k* represents the number of documents that contain the word 'w'

while, *M* represents the total number of documents in the collection

The final score of any document is calculated as follows:

$$Score\ (d) = \sum_{all\ words\ w\ in\ d} TF\ (w) * IDF\ (w)$$

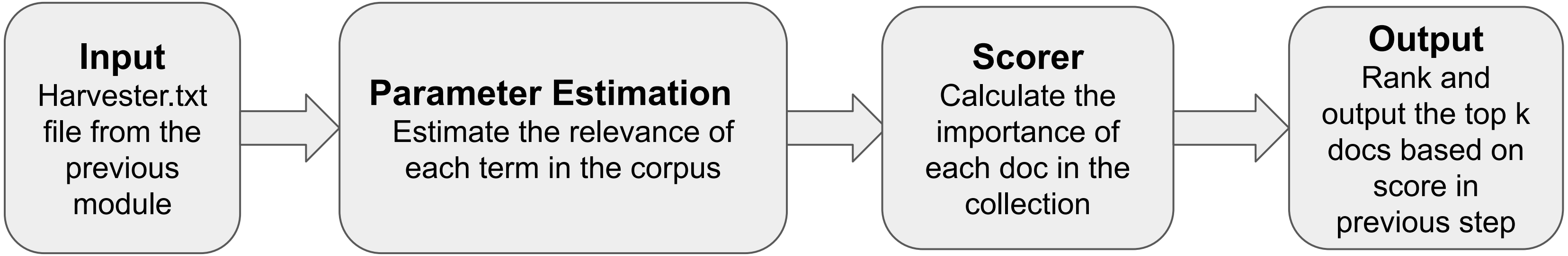
$$Normalised\ Score\ (d) = \frac{Score(d)}{len(d)}$$

len(d) represents the number of words in the document. This normalization is performed to eliminate any length bias the collection may have.

The output is written into ../results/RankerOutput.txt, which will be fed to the next module (the sentiment analyzer).

Document Ranker

- Input: Output txt file from the Harvester
- Output: Highly ranked documents in the input
- Parameters: Term Frequency Calculation Method
- Format: Score and Document (Refer same output file)



Input
Harvester.txt file from the previous module

Parameter Estimation
Estimate the relevance of each term in the corpus

Scorer
Calculate the importance of each doc in the collection

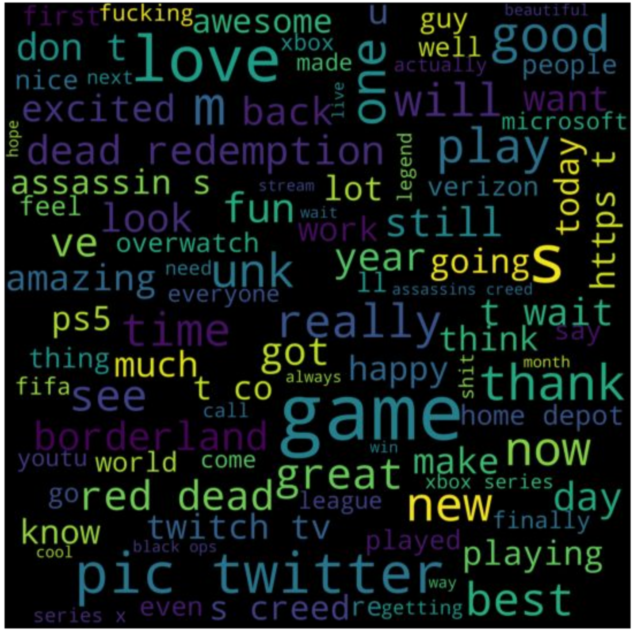
Output
Rank and output the top k docs based on score in previous step

NLTK Implemented Text Analysis

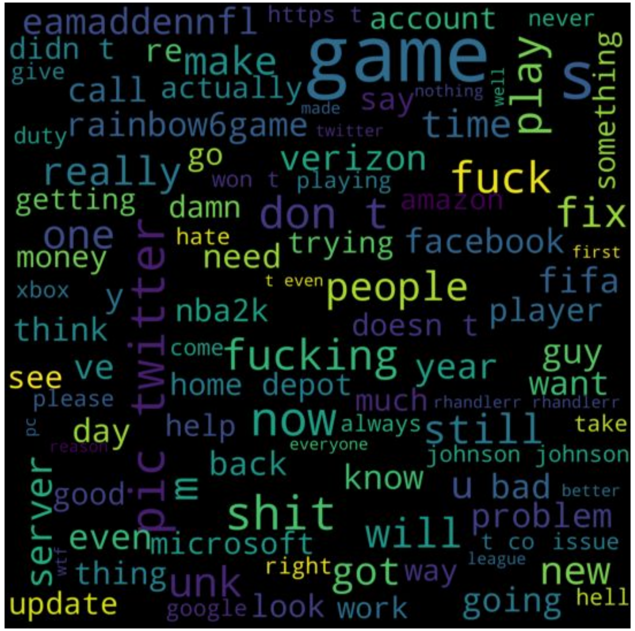
1. Initial data transformation
2. Plotting features
3. Text analysis
4. Logistic Regression model

Plotting features

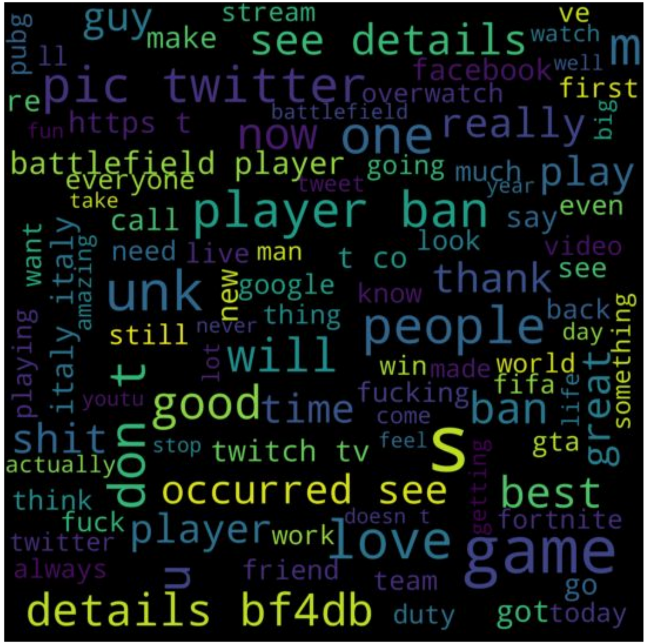
Positive:



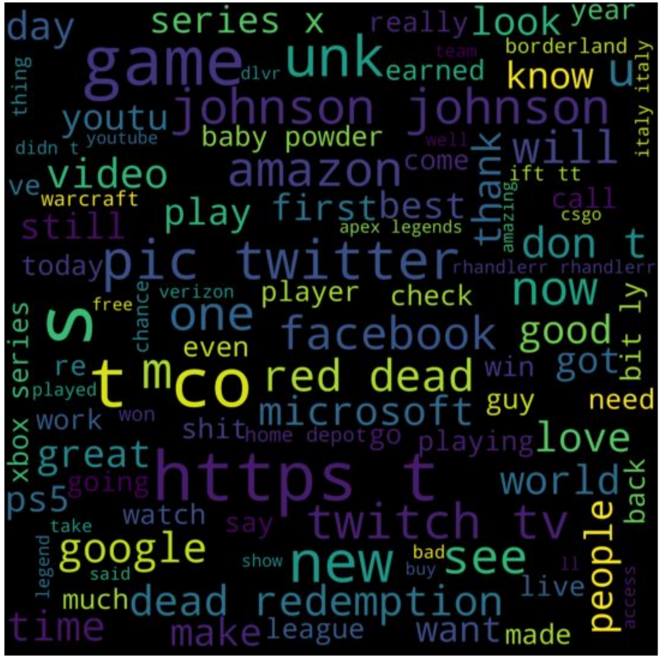
Negative:



Irrelevant

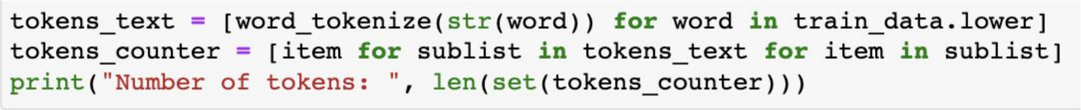


Neutral



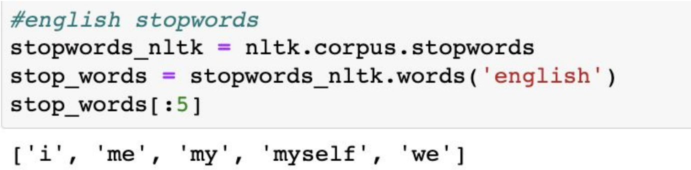
Text analysis

1. Calculate the initial number of unique tokens to determine the complexity of the model. The tokens_text variable groups all the texts by the different words stored on a List.



```
tokens_text = [word_tokenize(str(word)) for word in train_data.lower]
tokens_counter = [item for sublist in tokens_text for item in sublist]
print("Number of tokens: ", len(set(tokens_counter)))
```

2. The main English stopwords were saved on an additional variable, to be used in the following modeling.



```
#english stopwords
stopwords_nltk = nltk.corpus.stopwords
stop_words = stopwords_nltk.words('english')
stop_words[:5]
```

```
['i', 'me', 'my', 'myself', 'we']
```

Model

1. Initial Bag of Words.
2. The main data was split on train and test datasets alongside the encoding of the words by using the training dataset as a reference.

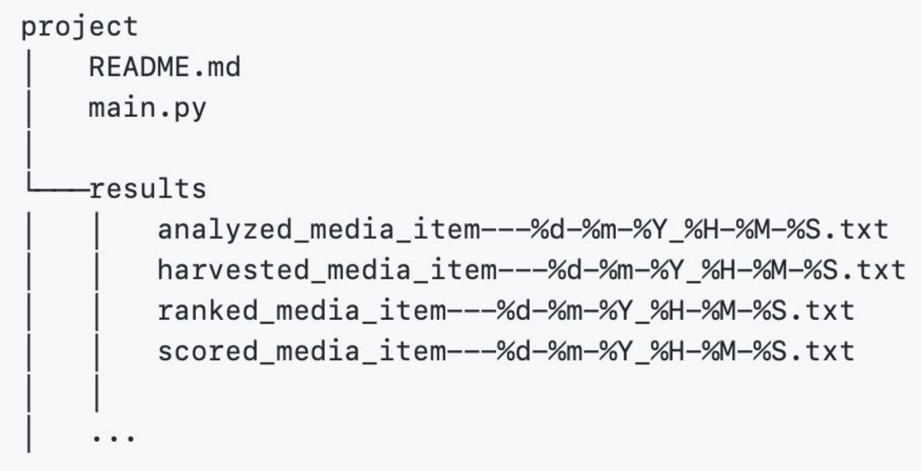
MLTK Command-Line Interface

User Prompts:

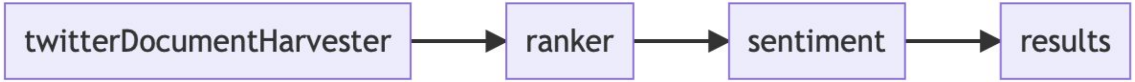
- Social media platform: [twitter]*
- Media item: [video game, e.g. ‘Genshin Impact’]

Output:

- harvested_.txt
- ranked_.txt
- analyzed_.txt
- scored_.txt

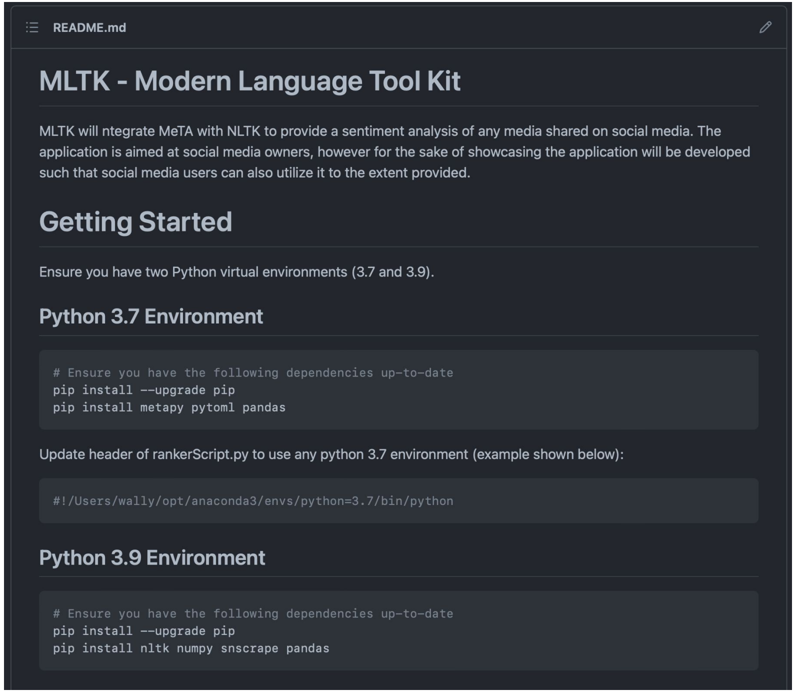


```
project
│   README.md
│   main.py
│
└───results
│   │   analyzed_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   harvested_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   ranked_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   scored_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │
│   ...
```



twitterDocumentHarvester → ranker → sentiment → results

MLTK Preview



MLTK - Modern Language Tool Kit

MLTK will ntegrate MeTA with NLTK to provide a sentiment analysis of any media shared on social media. The application is aimed at social media owners, however for the sake of showcasing the application will be developed such that social media users can also utilize it to the extent provided.

Getting Started

Ensure you have two Python virtual environments (3.7 and 3.9).

Python 3.7 Environment

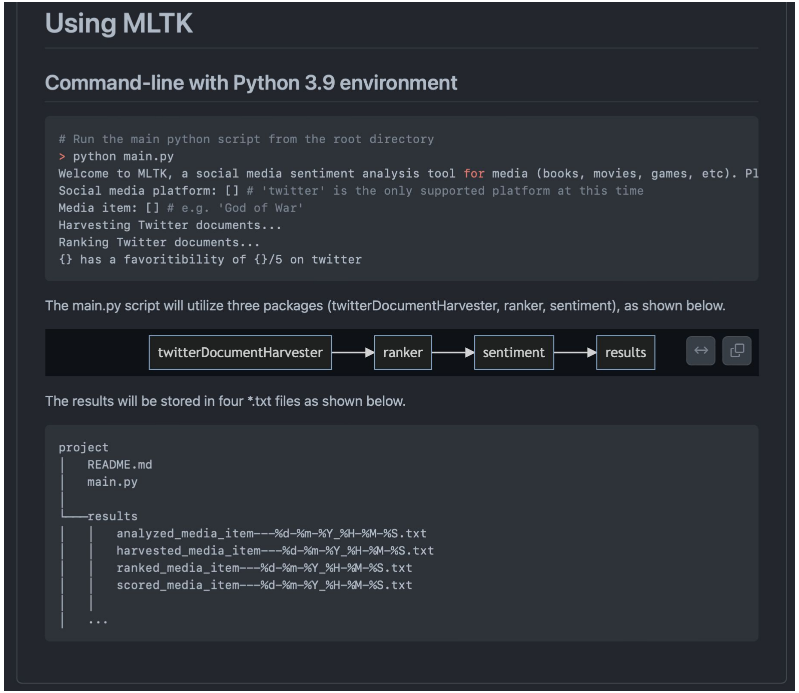
```
# Ensure you have the following dependencies up-to-date
pip install --upgrade pip
pip install metapy pytoml pandas
```

Update header of rankerScript.py to use any python 3.7 environment (example shown below):

```
#!/Users/wally/opt/anaconda3/envs/python=3.7/bin/python
```

Python 3.9 Environment

```
# Ensure you have the following dependencies up-to-date
pip install --upgrade pip
pip install nltk numpy snscrape pandas
```



Using MLTK

Command-line with Python 3.9 environment

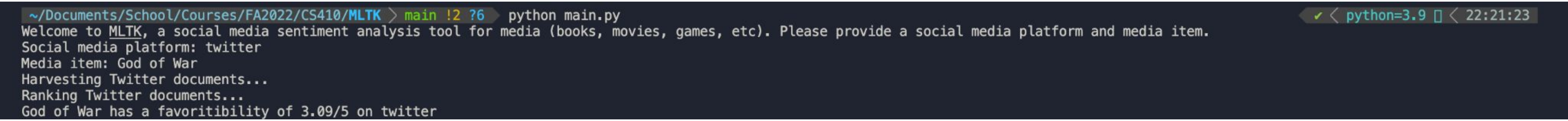
```
# Run the main python script from the root directory
> python main.py
Welcome to MLTK, a social media sentiment analysis tool for media (books, movies, games, etc). Pl
Social media platform: [] # 'twitter' is the only supported platform at this time
Media item: [] # e.g. 'God of War'
Harvesting Twitter documents...
Ranking Twitter documents...
{} has a favoritibility of {}/5 on twitter
```

The main.py script will utilize three packages (twitterDocumentHarvester, ranker, sentiment), as shown below.

twitterDocumentHarvester → ranker → sentiment → results

The results will be stored in four *.txt files as shown below.

```
project
│   README.md
│   main.py
│
└───results
│   │   analyzed_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   harvested_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   ranked_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │   scored_media_item---%d-%m-%Y_%H-%M-%S.txt
│   │
│   ...
```



```
~/Documents/School/Courses/FA2022/CS410/MLTK > main !2 ?6  python main.py                                         ✓ < python=3.9 < 22:21:23
Welcome to MLTK, a social media sentiment analysis tool for media (books, movies, games, etc). Please provide a social media platform and media item.
Social media platform: twitter
Media item: God of War
Harvesting Twitter documents...
Ranking Twitter documents...
God of War has a favoritibility of 3.09/5 on twitter
```