CP 468-A - Artificial Intelligence

Course Project

Web Document Classifier

AUTHORS:

ISHA RAULJI	: 170639370
ISHMEET SINGH	: 186800780
MOHAMMED PEERA	: 173242560
SAMBHAV JAIN	: 173146450

Table Of Content

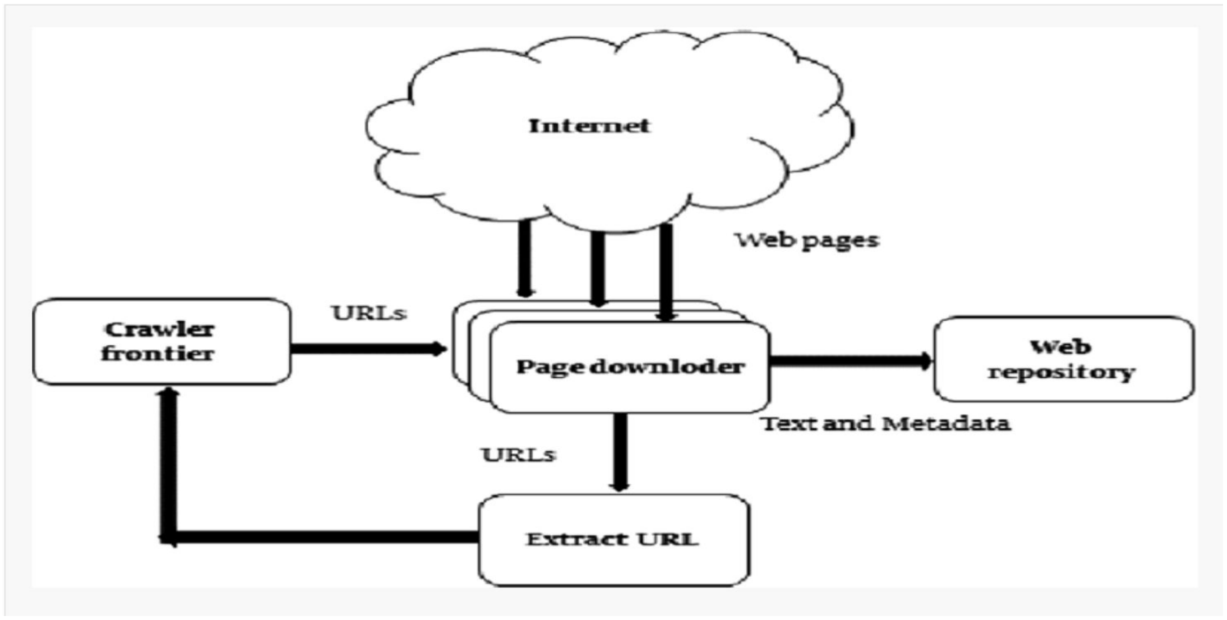
INTRODUCTION

[2] A web crawler is a program that searches the World Wide Web to collect data from different pages and is then used for analysis. Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine, that will index the downloaded pages to provide fast searches.Crawlers can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code.Also, crawlers can be used to gather specific types of information from Web pages, such as harvesting email addresses.

What Is Web Scraping?

[3] The process of capturing information from the internet in the form of web documents. The word “web scraping” actually means a process that requires automation. They are used for a variety of reasons. The pages are scraped to capture important information like comparing the company data. Big companies use this method to capture customer’s review as it makes it more efficient for them to review the data.

Crawler Architecture:



Step 0 — Understanding What Data to Scrape

[5] If we look at different websites we can consider them as a piece of HTML code along with CSS for designing. Our task is to download the page via Python and scrape data from it. Downloading the page means the HTML code of the page.

Topics :

Topics Selected by Our Group

1) Artificial Intelligence
2) Leadership
3) Bees
4) Machine Learning
5) Mathematics
6) Operating System
7) Python

PROJECT DESCRIPTION:

Phase 1 includes a web crawler program that is written in python language. It is then used to gather data from different sets of topics. Phase 2 includes the creation of feature vectors by using different approaches like the boolean attributes and TFIDF model that has been used in phase 3. Phase 3 uses the same approach and the datasets created in phase 1 and feature vectors in phase 2 to compare different models like price match , reviews. The data is then used to compare with other companies to do a price analysis, revenue checks etc.. In this project, we investigated the process of tagging/classifying web pages by applying data classification techniques for auto-tagging/classifying web pages as subject categories.

In addition, responses were filtered by search engine or web page ranking based on relevance to the topics specified.

OBJECTIVE:

- Create web-crawler program for phase 1 and scrape the topics provided by the user
- Feature Vectors are created in phase 2 using feature extraction.
- Different approaches like the boolean attributes and TF-IDF were used for this program to create feature vectors.

PHASE 1:

- The web scraping program was written in python language. The program then asks for input on different topics which the user wants to crawl.
- As we input the topics, the program automatically creates a folder called a dataset in which other folders with the topics names are created.
- The program adds the scraped content in the form of txt files inside the document. The user can select the number of topics that he wants to crawl and it should be minimum 3 as per the project requirements.
- At the end of this phase the user will be having several web documents created inside the dataset folder.

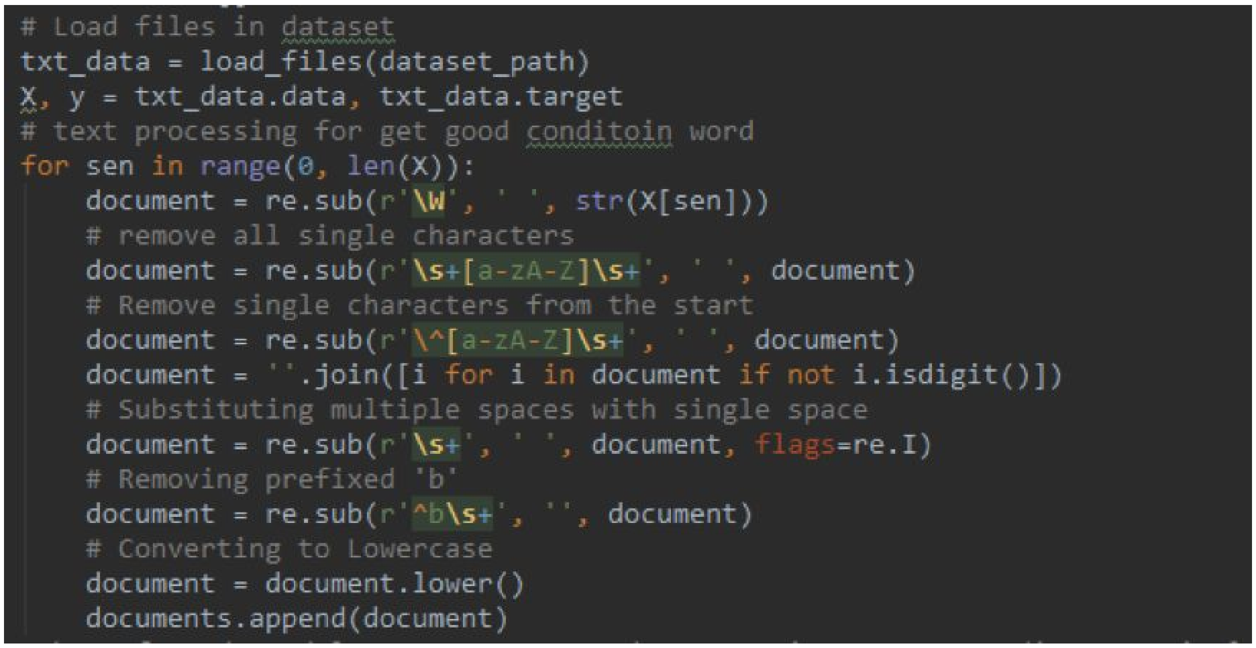
PHASE 2:

- Given the dataset results from Phase 1, we now had a collection of documents under several topics. However, there is not a lot of insight we can gain from just a collection of documents.
- Phase 2 focused on taking the documents from the dataset and created vectors via feature extraction.
- Feature extraction are methods that will select keywords from text and create features to effectively describe the data from the dataset.
- For this to happen, we created a program that would read files, and used the TF-IDF approach and Boolean attributes to compute a score for each feature, this approach would also be eliminating common words like “a, the, will, not, etc.” to keep the vectors as efficient and precise.
- At the end of phase 2, we aimed to have feature vectors for each collected document and to be exported into the dataset directory.

PHASE 3:

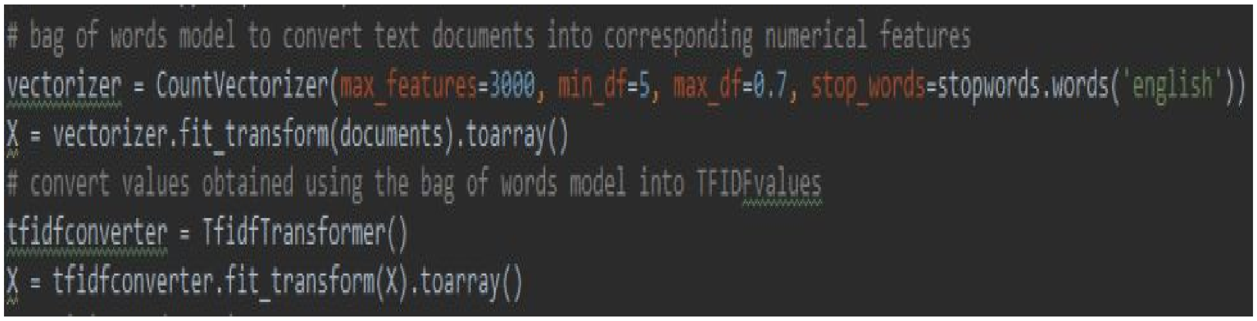
- Phase 3 involved using Machine Learning algorithms that can take Phase 2 a step forward.
- We aimed to create models from our dataset that can be used for new data input and have the data sorted with the feature vectors provided from Phase 2.
- We used Random Forest classification for our Phase 3 which is a supervised learning approach.
- At the end of Phase 3, we take the vector dataset and then create a confusion matrix, classification report, and an accuracy score.

Import dataset files and process texts with condition.



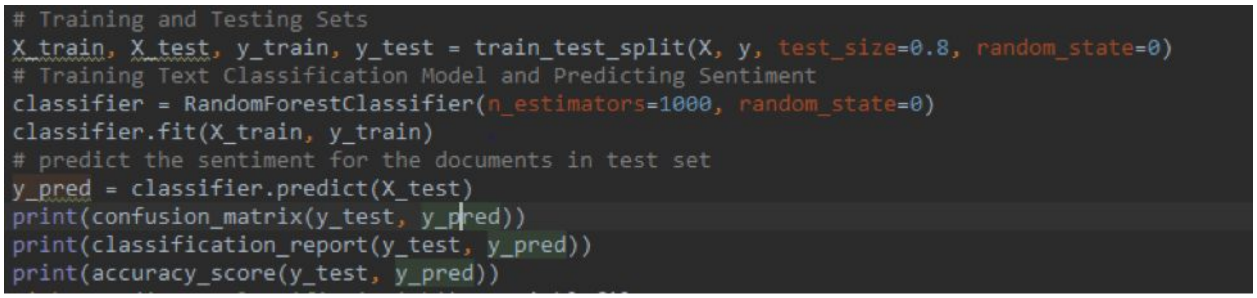
```
# Load files in dataset
txt_data = load_files(dataset_path)
X, y = txt_data.data, txt_data.target
# text processing for get good conditoin word
for sen in range(0, len(X)):
    document = re.sub(r'\W', ' ', str(X[sen]))
    # remove all single characters
    document = re.sub(r'\s+[a-zA-Z]\s+', ' ', document)
    # Remove single characters from the start
    document = re.sub(r'\^[a-zA-Z]\s+', ' ', document)
    document = ''.join([i for i in document if not i.isdigit()])
    # Substituting multiple spaces with single space
    document = re.sub(r'\s+', ' ', document, flags=re.I)
    # Removing prefixed 'b'
    document = re.sub(r'^b\s+', '', document)
    # Converting to Lowercase
    document = document.lower()
    documents.append(document)
```

Converting Text to Numbers and finding TF-IDF



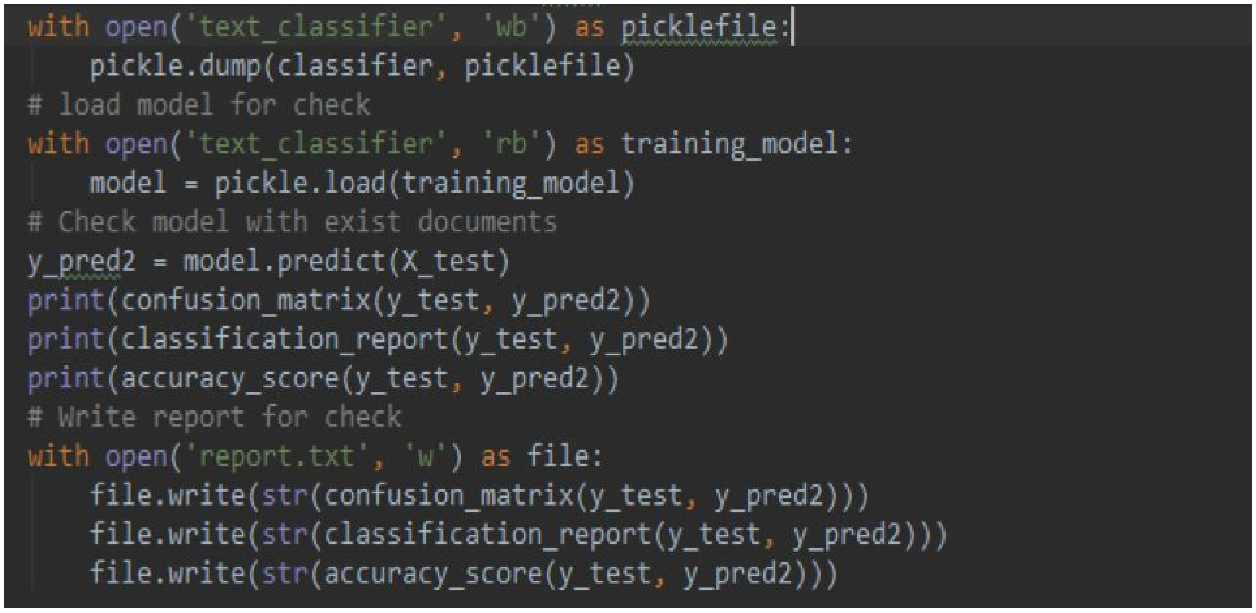
```
# bag of words model to convert text documents into corresponding numerical features
vectorizer = CountVectorizer(max_features=3000, min_df=5, max_df=0.7, stop_words=stopwords.words('english'))
X = vectorizer.fit_transform(documents).toarray()
# convert values obtained using the bag of words model into TFIDFvalues
tfidfconverter = TfidfTransformer()
X = tfidfconverter.fit_transform(X).toarray()
```

Training,Testing Sets and out predict result



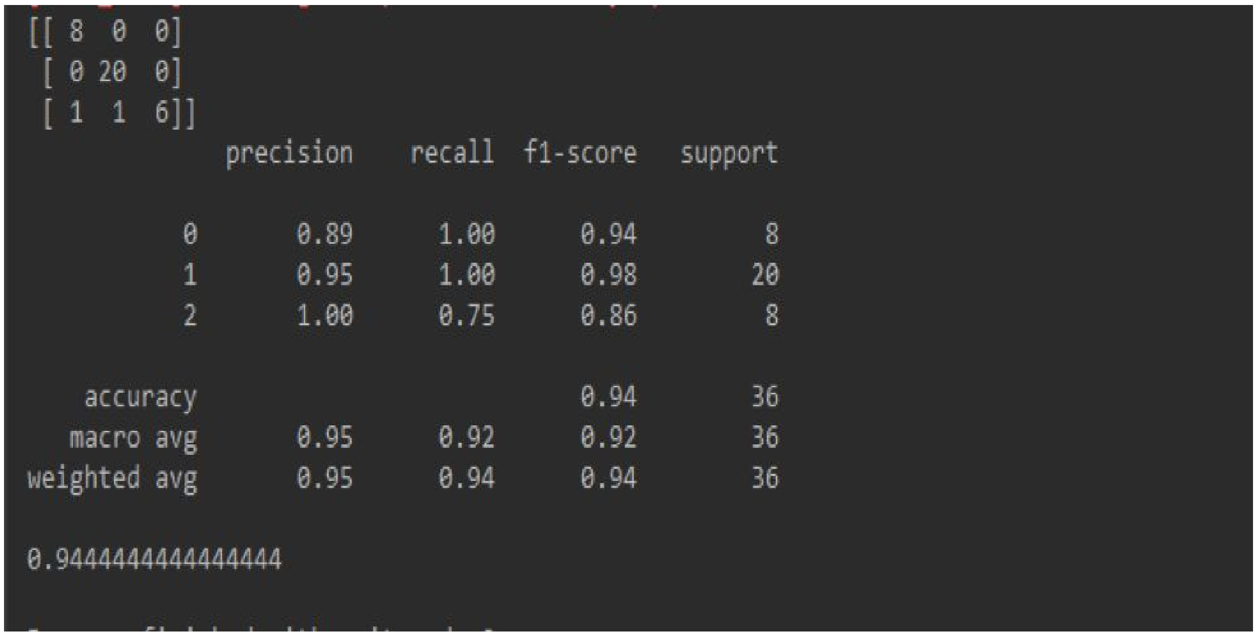
```
# Training and Testing Sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.8, random_state=0)
# Training Text Classification Model and Predicting Sentiment
classifier = RandomForestClassifier(n_estimators=1000, random_state=0)
classifier.fit(X_train, y_train)
# predict the sentiment for the documents in test set
y_pred = classifier.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
print(accuracy_score(y_test, y_pred))
```

Save model in file and load model for check



```
with open('text_classifier', 'wb') as picklefile:
    pickle.dump(classifier, picklefile)
# load model for check
with open('text_classifier', 'rb') as training_model:
    model = pickle.load(training_model)
# Check model with exist documents
y_pred2 = model.predict(X_test)
print(confusion_matrix(y_test, y_pred2))
print(classification_report(y_test, y_pred2))
print(accuracy_score(y_test, y_pred2))
# Write report for check
with open('report.txt', 'w') as file:
    file.write(str(confusion_matrix(y_test, y_pred2)))
    file.write(str(classification_report(y_test, y_pred2)))
    file.write(str(accuracy_score(y_test, y_pred2)))
```

Result is model.py and text_classifier file



```
[[ 8  0  0]
 [ 0 20  0]
 [ 1  1  6]]
              precision    recall  f1-score   support

           0       0.89      1.00      0.94         8
           1       0.95      1.00      0.98        20
           2       1.00      0.75      0.86         8

    accuracy                           0.94        36
   macro avg       0.95      0.92      0.92        36
weighted avg       0.95      0.94      0.94        36

0.9444444444444444
```

INSTALLATION

- **Install XCode**

  In your Terminal app, run the following command to install XCode and its command-line tools:

```
$ xcode-select --install
```

  It is a large program so this may take a while to download. Make sure to click through all the confirmation prompts that XCode requires.

- **Install Homebrew**

  Next install Homebrew by copy/pasting the following command into Terminal and then type Enter:

```
$ ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

  To confirm Homebrew installed correctly, run the command:

```
$ brew doctor

Your system is ready to brew.
```

- **Install Python 3**

  Now we can install the latest version of Python 3. Type the following command into Terminal and press Enter:

```
$ brew install python3
```

To confirm which version of Python 3 was installed, run the following command in Terminal:

```
$ python3 --version
Python 3.7.7
```

- **Install Google**

As you will run the program, it will die with an error saying cannot import name 'search' from 'googlesearch'. Run the following command in Terminal:

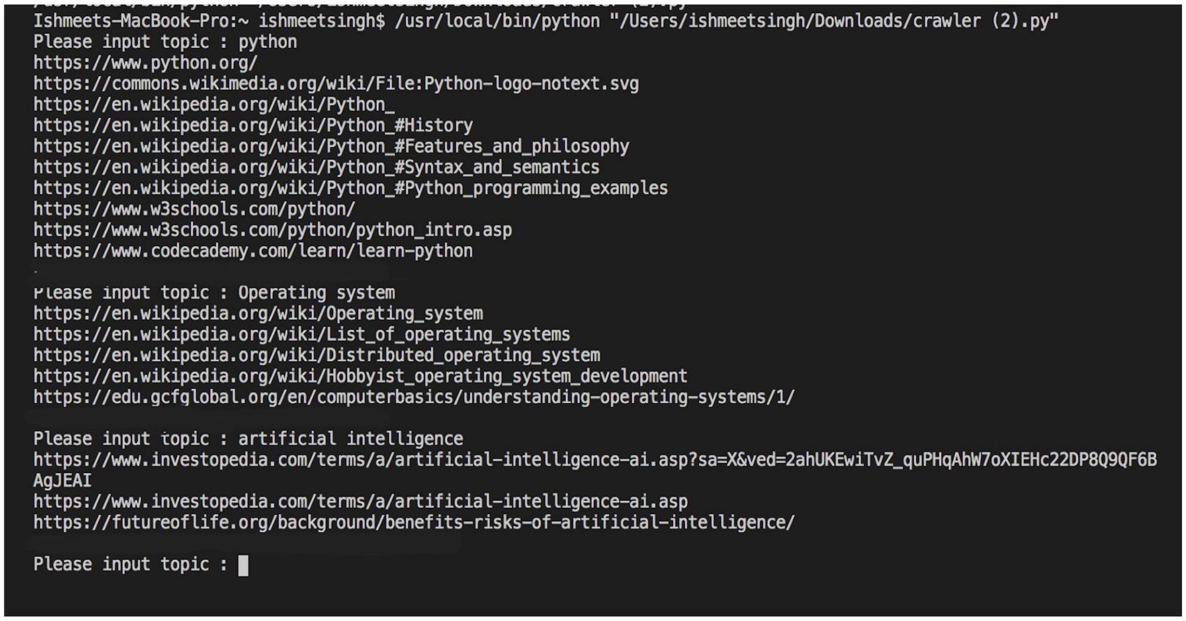
HOW TO RUN GUIDE:

1) Phase 1 : Download the project on your desktop and type the commands given below on Terminal:

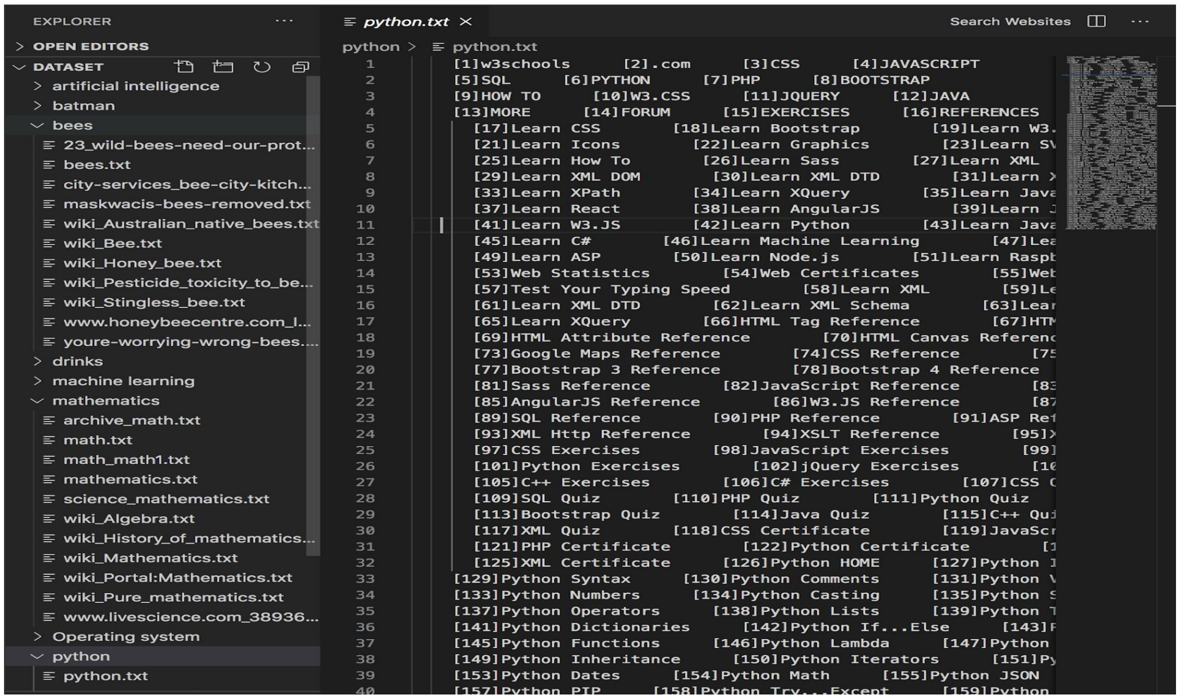
```
cd Desktop

Python3 <name of the program including the extension for example crawler.py>
```

2) It will then ask to input the topic and when you type in the topic it will create the folder and add the scraped documents in the dataset folder
3) Phase 2 : Use the same command to run phase 2 as shown in point 1, it will automatically create a folder dataset_vectors with the boolean and TD-IDF values
4) Phase 3: Use the same command mentioned in point 1 to run phase 3. It will print the output in terminal indicating the values.

OUTPUT:

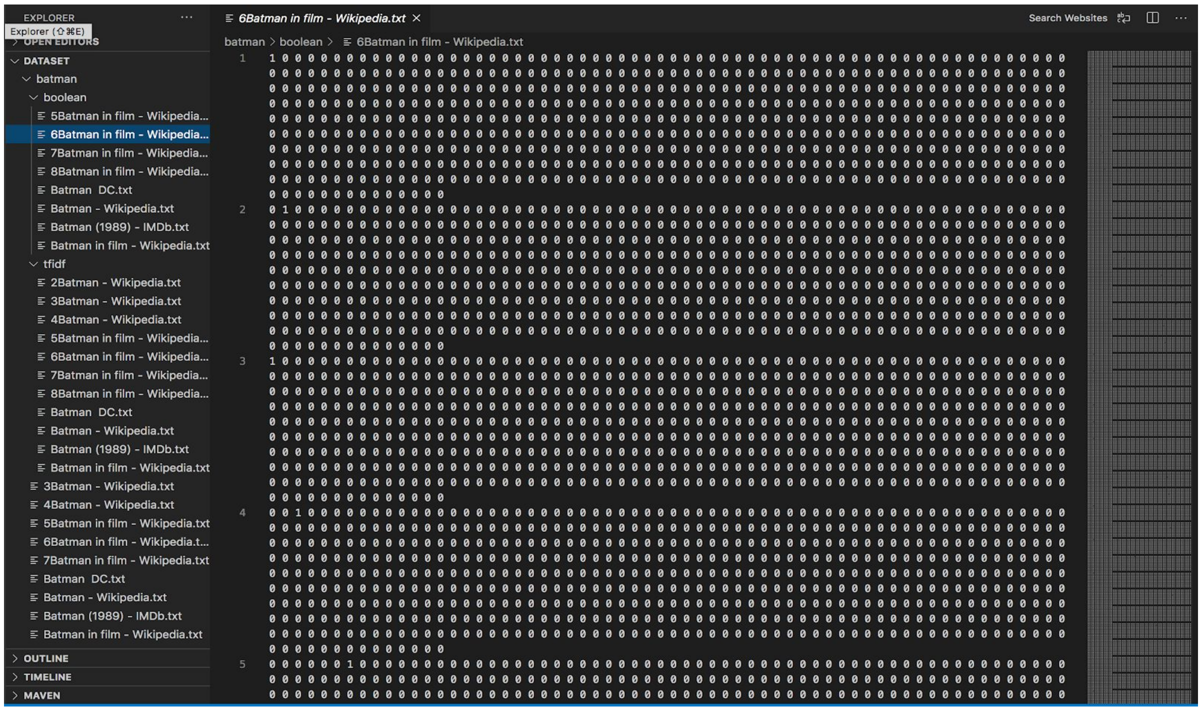


```
Ishmeets-MacBook-Pro:~ ishmeetsingh$ /usr/local/bin/python "/Users/ishmeetsingh/Downloads/crawler (2).py"
Please input topic : python
https://www.python.org/
https://commons.wikimedia.org/wiki/File:Python-logo-notext.svg
https://en.wikipedia.org/wiki/Python_
https://en.wikipedia.org/wiki/Python_#History
https://en.wikipedia.org/wiki/Python_#Features_and_philosophy
https://en.wikipedia.org/wiki/Python_#Syntax_and_semantics
https://en.wikipedia.org/wiki/Python_#Python_programming_examples
https://www.w3schools.com/python/
https://www.w3schools.com/python/python_intro.asp
https://www.codecademy.com/learn/learn-python

Please input topic : Operating system
https://en.wikipedia.org/wiki/Operating_system
https://en.wikipedia.org/wiki/List_of_operating_systems
https://en.wikipedia.org/wiki/Distributed_operating_system
https://en.wikipedia.org/wiki/Hobbyist_operating_system_development
https://edu.gcfglobal.org/en/computerbasics/understanding-operating-systems/1/

Please input topic : artificial intelligence
https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp?sa=X&ved=2ahUKEwiTvZ_quPHqAhW7oXIEHc22DP8Q9QF6B
AgJEAI
https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp
https://futureoflife.org/background/benefits-risks-of-artificial-intelligence/

Please input topic :
```

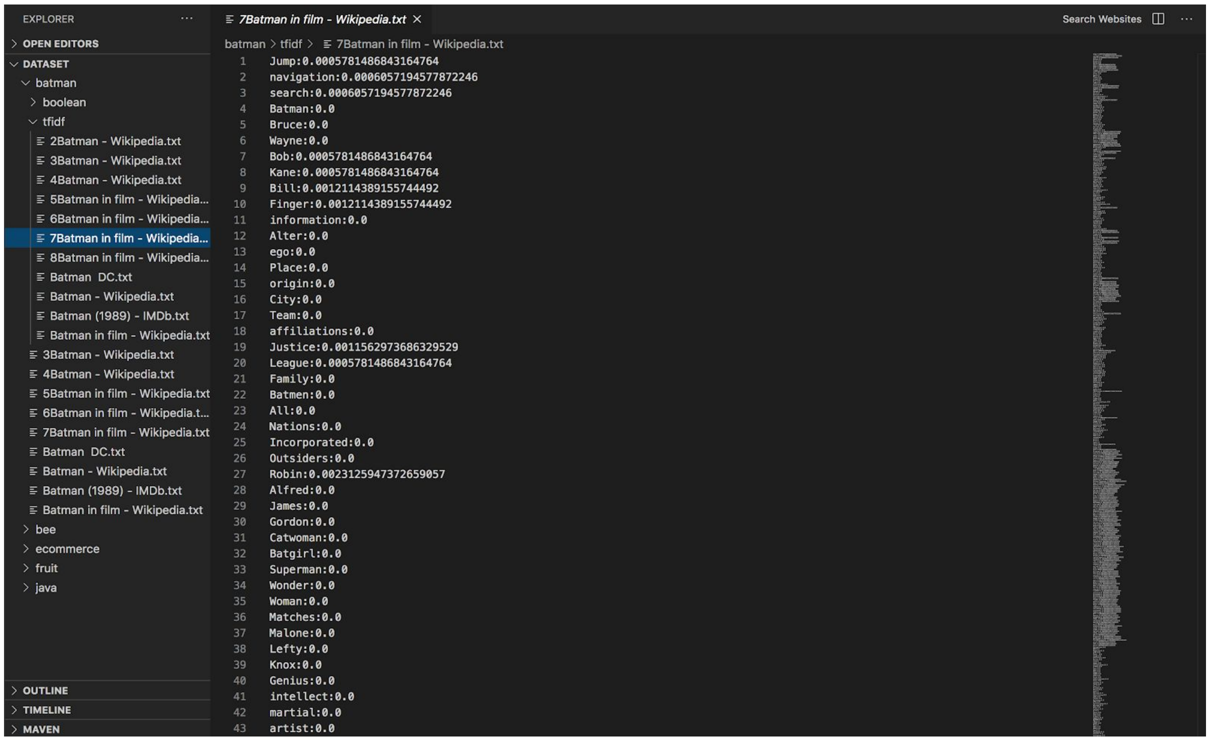


```
> artificial intelligence
> batman
v bees
    23_wild-bees-need-our-prot...
    bees.txt
    city-services_bee-city-kitch...
    maskwacis-bees-removed.txt
    wiki_Australian_native_bees.txt
    wiki_Bee.txt
    wiki_Honey_bee.txt
    wiki_Pesticide_toxicity_to_be...
    wiki_Stingless_bee.txt
    www.honeybeecentre.com_l...
    youre-worrying-wrong-bees....
> drinks
> machine learning
v mathematics
    archive_math.txt
    math.txt
    math_math1.txt
    mathematics.txt
    science_mathematics.txt
    wiki_Algebra.txt
    wiki_History_of_mathematics...
    wiki_Mathematics.txt
    wiki_Portal:Mathematics.txt
    wiki_Pure_mathematics.txt
    www.livescience.com_38936...
> Operating system
v python
    python.txt
```

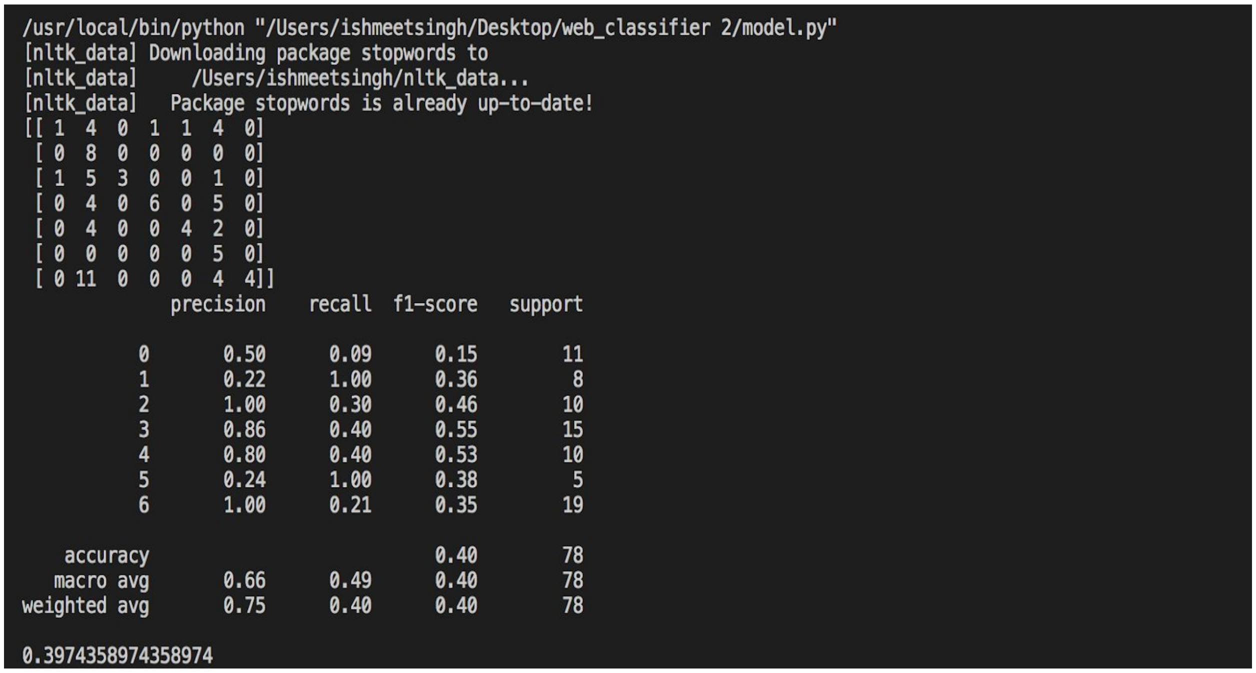
```
python > python.txt
[1]w3schools      [2].com      [3]CSS      [4]JAVASCRIPT
[5]SQL      [6]PYTHON      [7]PHP      [8]BOOTSTRAP
[9]HOW TO      [10]W3.CSS      [11]JQUERY      [12]JAVA
[13]MORE      [14]FORUM      [15]EXERCISES      [16]REFERENCES
  [17]Learn CSS        [18]Learn Bootstrap        [19]Learn W3.
  [21]Learn Icons        [22]Learn Graphics        [23]Learn SV
  [25]Learn How To        [26]Learn Sass        [27]Learn XML
  [29]Learn XML DOM        [30]Learn XML DTD        [31]Learn X
  [33]Learn XPath        [34]Learn XQuery        [35]Learn Java
  [37]Learn React        [38]Learn AngularJS        [39]Learn J
  [41]Learn W3.JS        [42]Learn Python        [43]Learn Java
  [45]Learn C#        [46]Learn Machine Learning        [47]Lea
  [49]Learn ASP        [50]Learn Node.js        [51]Learn Raspb
  [53]Web Statistics        [54]Web Certificates        [55]Web
  [57]Test Your Typing Speed        [58]Learn XML        [59]Le
  [61]Learn XML DTD        [62]Learn XML Schema        [63]Lear
  [65]Learn XQuery        [66]HTML Tag Reference        [67]HTM
  [69]HTML Attribute Reference        [70]HTML Canvas Referenc
  [73]Google Maps Reference        [74]CSS Reference        [75
  [77]Bootstrap 3 Reference        [78]Bootstrap 4 Reference
  [81]Sass Reference        [82]JavaScript Reference        [83
  [85]AngularJS Reference        [86]W3.JS Reference        [87
  [89]SQL Reference        [90]PHP Reference        [91]ASP Ref
  [93]XML Http Reference        [94]XSLT Reference        [95]X
  [97]CSS Exercises        [98]JavaScript Exercises        [99]
  [101]Python Exercises        [102]jQuery Exercises        [10
  [105]C++ Exercises        [106]C# Exercises        [107]CSS C
  [109]SQL Quiz        [110]PHP Quiz        [111]Python Quiz
  [113]Bootstrap Quiz        [114]Java Quiz        [115]C++ Qui
  [117]XML Quiz        [118]CSS Certificate        [119]JavaScr
  [121]PHP Certificate        [122]Python Certificate        [1
  [125]XML Certificate        [126]Python HOME        [127]Python I
[129]Python Syntax      [130]Python Comments      [131]Python V
[133]Python Numbers      [134]Python Casting      [135]Python S
[137]Python Operators      [138]Python Lists      [139]Python T
[141]Python Dictionaries      [142]Python If...Else      [143]P
[145]Python Functions      [146]Python Lambda      [147]Python
[149]Python Inheritance      [150]Python Iterators      [151]Py
[153]Python Dates      [154]Python Math      [155]Python JSON
[157]Python PIP      [158]Python Try...Except      [159]Python
```


EXPLORER
Explorer (⇧⌘E)
OPEN EDITORS
DATASET
batman
boolean
5Batman in film - Wikipedia...
6Batman in film - Wikipedia...
7Batman in film - Wikipedia...
8Batman in film - Wikipedia...
Batman DC.txt
Batman - Wikipedia.txt
Batman (1989) - IMDb.txt
Batman in film - Wikipedia.txt
tfidf
2Batman - Wikipedia.txt
3Batman - Wikipedia.txt
4Batman - Wikipedia.txt
5Batman in film - Wikipedia...
6Batman in film - Wikipedia...
7Batman in film - Wikipedia...
8Batman in film - Wikipedia...
Batman DC.txt
Batman - Wikipedia.txt
Batman (1989) - IMDb.txt
Batman in film - Wikipedia.txt
3Batman - Wikipedia.txt
4Batman - Wikipedia.txt
5Batman in film - Wikipedia.txt
6Batman in film - Wikipedia.t...
7Batman in film - Wikipedia.txt
Batman DC.txt
Batman - Wikipedia.txt
Batman (1989) - IMDb.txt
Batman in film - Wikipedia.txt
OUTLINE
TIMELINE
MAVEN
6Batman in film - Wikipedia.txt
Search Websites
batman > boolean > 6Batman in film - Wikipedia.txt

```
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```


EXPLORER
OPEN EDITORS
DATASET
batman
boolean
tfidf
2Batman - Wikipedia.txt
3Batman - Wikipedia.txt
4Batman - Wikipedia.txt
5Batman in film - Wikipedia...
6Batman in film - Wikipedia...
7Batman in film - Wikipedia...
8Batman in film - Wikipedia...
Batman DC.txt
Batman - Wikipedia.txt
Batman (1989) - IMDb.txt
Batman in film - Wikipedia.txt
3Batman - Wikipedia.txt
4Batman - Wikipedia.txt
5Batman in film - Wikipedia.txt
6Batman in film - Wikipedia.t...
7Batman in film - Wikipedia.txt
Batman DC.txt
Batman - Wikipedia.txt
Batman (1989) - IMDb.txt
Batman in film - Wikipedia.txt
bee
ecommerce
fruit
java
OUTLINE
TIMELINE
MAVEN
7Batman in film - Wikipedia.txt
Search Websites
batman > tfidf > 7Batman in film - Wikipedia.txt

```
Jump:0.0005781486843164764
navigation:0.0006057194577872246
search:0.0006057194577872246
Batman:0.0
Bruce:0.0
Wayne:0.0
Bob:0.0005781486843164764
Kane:0.0005781486843164764
Bill:0.0012114389155744492
Finger:0.0012114389155744492
information:0.0
Alter:0.0
ego:0.0
Place:0.0
origin:0.0
City:0.0
Team:0.0
affiliations:0.0
Justice:0.0011562973686329529
League:0.0005781486843164764
Family:0.0
Batmen:0.0
All:0.0
Nations:0.0
Incorporated:0.0
Outsiders:0.0
Robin:0.0023125947372659057
Alfred:0.0
James:0.0
Gordon:0.0
Catwoman:0.0
Batgirl:0.0
Superman:0.0
Wonder:0.0
Woman:0.0
Matches:0.0
Malone:0.0
Lefty:0.0
Knox:0.0
Genius:0.0
intellect:0.0
martial:0.0
artist:0.0
```



```
/usr/local/bin/python "/Users/ishmeetsingh/Desktop/web_classifier 2/model.py"
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/ishmeetsingh/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[[ 1  4  0  1  1  4  0]
 [ 0  8  0  0  0  0  0]
 [ 1  5  3  0  0  1  0]
 [ 0  4  0  6  0  5  0]
 [ 0  4  0  0  4  2  0]
 [ 0  0  0  0  0  5  0]
 [ 0 11  0  0  0  4  4]]
              precision    recall  f1-score   support

           0       0.50      0.09      0.15        11
           1       0.22      1.00      0.36         8
           2       1.00      0.30      0.46        10
           3       0.86      0.40      0.55        15
           4       0.80      0.40      0.53        10
           5       0.24      1.00      0.38         5
           6       1.00      0.21      0.35        19

    accuracy                           0.40        78
   macro avg       0.66      0.49      0.40        78
weighted avg       0.75      0.40      0.40        78

0.3974358974358974
```

CONCLUSION

- Phase 1 provides us with a [1] **Web Crawler** program that traverses the Web and downloads the web document that the user needs. Web crawlers were used in this project to retrieve the web documents in txt files so it can be used to compare things like prices review etc. which is shown in Phase 3.
- Phase 2 provides us with features vectors using different via feature extraction. Different approaches like **boolean attribute** and **TFIDF** were applied on the dataset folder to convert them to feature vectors.
- Phase 3 uses **RandomForestClassifier** data classification algorithm to create models that calculate the accuracy of the topics.

Work Partitioning

Work Done	**Name**
Progress Report	ISHMEET SINGH
Phase 1	ISHMEET SINGH
Phase 2	ISHA RAULJI
Phase 3	ISHMEET SINGH
Power-Point Presentation	ISHA RAULJI
Video Presentation	ISHA RAULJI
Final Report	ISHMEET SINGH

REFERENCES

[1] https://www.researchgate.net/publication/258789938_Web_Crawler_A_Review

[2] https://www.sciencedaily.com/terms/web_crawler.htm

[3] https://realpython.com/beautiful-soup-web-scraper-python/

[4] https://towardsdatascience.com/introduction-to-web-scraping-with-beautifulsoup-e87a06c2b857

[5] https://towardsdatascience.com/web-scraping-5649074f3ead