Contents

List of Figures: -

Information Retrieval System Development using NLP

Name: Engr. Ahmad Ali

Company: Rex Technologies

I. Problem Statement: -

The Objective of this task is to make an Information Retrieval System(IRS) on the Data which we have to get from the online platform KAGGLE, the platform where we get any type of data for our Model Training.

II. Introduction: -

In this modern era the world is moving towards the automation of things for their ease. Everything we see today from our Home to Shopping, is automated through Technology. It has vital role in our daily life, the social media platforms and use of technology is increase day after the day. It's not wrong to say we are now depending on technology for our daily work. The peak of technology is after the 20's century in which people introduce new methods and techniques to solve the human problems. Artificial Intelligence is one of them which solve many Human problems which seems to be difficult. Now days AI play its role in almost all the sectors like Medical, Retail, Electrical and Social Media. It analyzes the human behavior through Deep Learning and perform task which human could not do. Through the availability of data, the Computer is trained by humans and then that train model performs certain tasks which if human do will take many time. The analysis of the emotion or to predict a person feeling is difficult for humans some time because we can't judge perfectly about the others meanings perfectly sometimes so here comes the AI which solve this problem for humans.

III. Information Retrieval System(IRS): -

The first question arises is what is actually a IRS? **Finding required documents from a collection of documents is known as information retrieval.** The user enters his need as a text query in the information retrieval system to make it work. After processing this query, the system searches the corpus of already-existing documents for the pertinent documents. The user is then supplied these pertinent documents in diminishing relevancy sequence. The quality of our results in this entire procedure depends on the rank of the documents returned. It is not the task in information retrieval issues to simply return pertinent materials. Instead, you must return the documents in order of relevance, starting with the ones that are most pertinent. **Document ranking** is the term for this. There are several methods for ranking documents for a query, however in this tutorial we'll just utilise the TF-IDF and Cosine score.

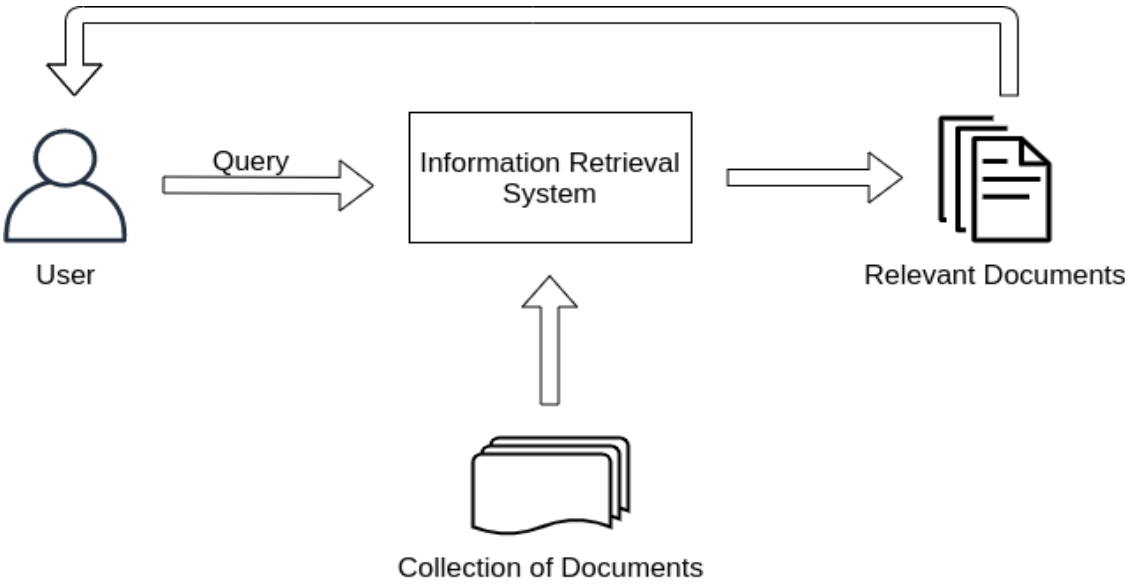

Figure 1 IRS Basic System Layout

I. Steps in IRS: -

- Getting Data and Cleaning
- Visualize Data(EDA)
- Pre-processing
- Vector making using TF-IDF
- Query Processing
- Cosine score to fetch relevant Documents
- Testing Model

IV. Getting Data and Cleaning Data: -

That step is the most Crucial step for any kind of problem or Model Training. All the other steps are depending on this step. We get our problem set data from the KAGGLE and then clean the data. The selection of data involves following steps:

- Identify Problem
- Choose Data according to that problem
- Clean the data (Remove duplicated, Fill Null values with Mode, Mean etc.)
- Removing Punctuation and number's (Text Analysis case)
- Visualize Data
- Correlation hunting

df

	Index	Author	Date published	Category	Section	Url	Headline	Description	Keywords	Second headline	Article text
0	0	Jacopo Prisco, CNN	2021-07-15 02:46:59	news	world	https://www.cnn.com/2021/07/14/world/tusimple-...	There's a shortage of truckers, but TuSimple t...	The e-commerce boom has exacerbated a global t...	world, There's a shortage of truckers, but TuS...	There's a shortage of truckers, but TuSimple t...	(CNN)Right now, there's a shortage of truck d...
1	2	Stephanie Bailey, CNN	2021-05-12 07:52:09	news	world	https://www.cnn.com/2021/05/12/world/ironhand-...	Bioservo's robotic 'Ironhand' could protect fa...	Working in a factory can mean doing the same t...	world, Bioservo's robotic 'Ironhand' could pro...	A robotic 'Ironhand' could protect factory wor...	(CNN)Working in a factory or warehouse can me...
2	3	Words by Stephanie Bailey, video by Zahra Jamshed	2021-06-16 02:51:30	news	asia	https://www.cnn.com/2021/06/15/asia/swarm-robo...	This swarm of robots gets smarter the more it ...	In a Hong Kong warehouse, a swarm of autonomou...	asia, This swarm of robots gets smarter the mo...	This swarm of robots gets smarter the more it ...	(CNN)In a Hong Kong warehouse, a swarm of aut...
3	4	Paul R. La Monica, CNN Business	2022-03-15 09:57:36	business	investing	https://www.cnn.com/2022/03/15/investing/brics...	Russia is no longer an option for investors. T...	For many years, the world's most popular emerg...	investing, Russia is no longer an option for i...	Russia is no longer an option for investors. T...	New York (CNN Business)For many years, the wor...
4	7	Reuters	2022-03-15 11:27:02	business	business	https://www.cnn.com/2022/03/15/business/russia...	Russian energy investment ban part of new EU s...	The European Union formally approved on Tuesda...	business, Russian energy investment ban part o...	EU bans investment in Russian energy in new sa...	The European Union formally approved on Tuesda...

Figure 2 New Articles data for IRS

The data which we need for our problem is News article which have a lot of articles related to Sports and business articles and other type of articles. The data set have 11 columns and about 4000 samples the data set is shown above Figure. During the Cleaning I first check whether there is any NULL value or Duplicates in the data through the following command,

```
[3]  df.isnull().sum()

     Index              0
     Author             0
     Date published     0
     Category           0
     Section            0
     Url                0
     Headline           0
     Description        0
     Keywords           0
     Second headline    0
     Article text       0
     dtype: int64

[4]  df.duplicated().sum()

     0

[5]  df.shape

     (4076, 11)
```

Figure 3 Checking NULL and Duplicated in data set

V. Visualization(EDA): -

In this process we have to Visualize our data which give us an insight knowledge about our data. It is not possible for humans to read all the rows and columns of data set. So to make it easy we use Graphs to see the behavior or pattern in the data. It helps us in following,

- Data Exploration
- Feature Selection for model
- Outlier Detection
- Model Selection
- Hyper-parameter tuning for model
- Pattern Recognition

We use **Count Plot** method to visualize that what type of news articles and in which category they fall. The code and graph are shown below which show we have majority News articles for Sports and Business.

```
import matplotlib.pyplot as plt
a=df['Category'].value_counts()
plt.bar(a.index,a.values)
plt.xlabel('Category')
plt.ylabel('Count')
plt.title('Top 8 Article Category')
plt.xticks(rotation=0)
plt.show()
```

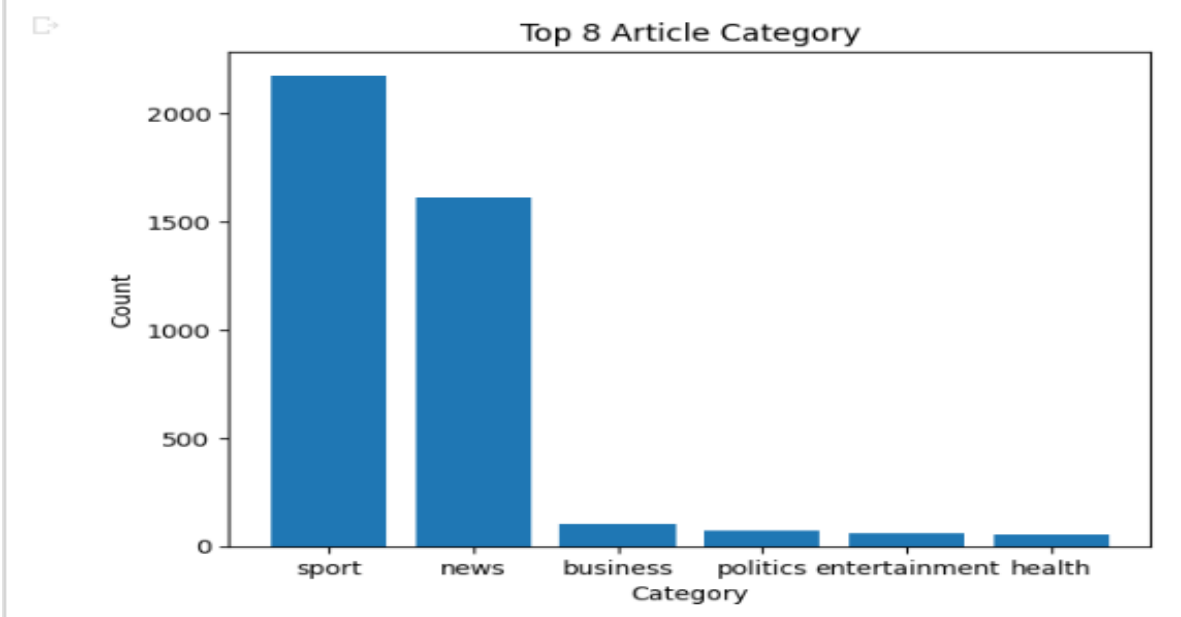

Figure 4 Count plot for Categories in data

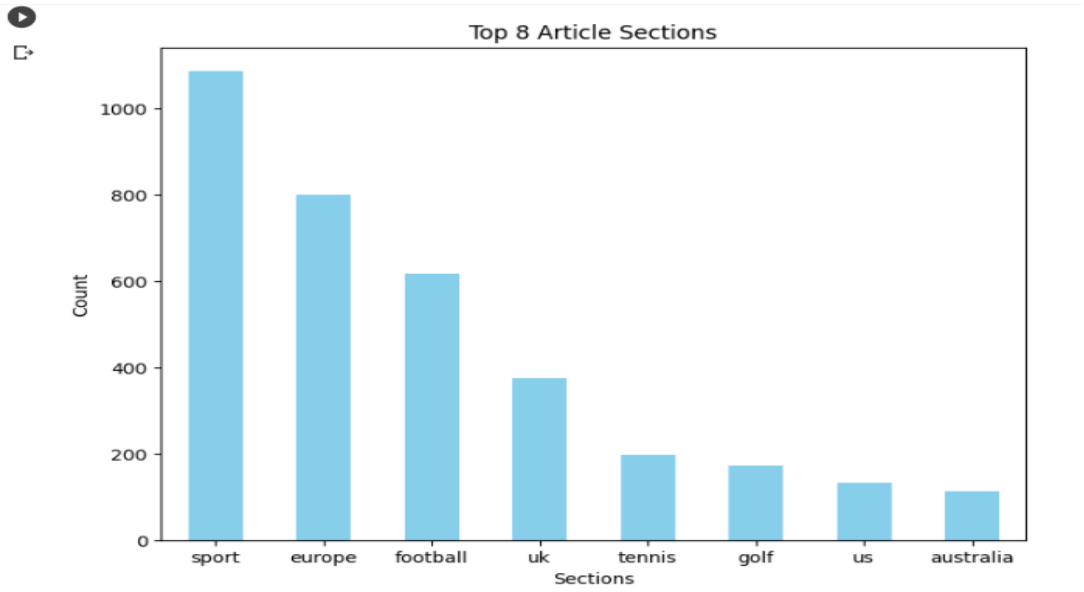

Figure 5 Count Plot for Section in Articles

VI. Pre-Processing Data: -

There are Steps in pre-processing which involve Removing Punctuation and numbers, Tokenization, Stemming, Stop words. First I select the column where we have to apply Pre-processing. We have many columns like Author name, Author ID, date and many others. But main column is **Article Text** which we use to make **Vectors** and compare with User Query vector and Fetch top 10 Documents for user. A key text processing task in natural language processing (NLP) is **tokenization**. It entails dividing a text into tokens, which are shorter pieces of text. Depending on the level of granularity necessary for a certain NLP activity, tokens can be individual words, sub words, or even characters. Tokenization is a preprocessing procedure used in a variety of NLP applications to facilitate working with text input and it also help the Computer to understand the Word or Sentence. Machine assign a specific number to each word which help in processing. It also helps in counting frequency of the words in the text. I apply Tokenization to my data as shown below. The next step is **Stemming**. Stemming also called Lemmatization is the Process to Normalize the words or text in to its **root word**. For example, if we have these words in our long text Affected, Affects, Affection and Affectation then these all word has same **Root** word which is **Affect.** So Stemming Perform that Task on out text and it also reduce the Length of our text which has no effect on model. The Main goal is to just get the Most important words which show sentiments or felling of a person. Stemming perform common prefixes and suffixes on the text or words to find the root words. After the Stemming our next step is to remove the irrelevant words like “is”, “am” or “there” etc. That is achieve by using **Stop words** module of nltk. That module has almost 120 above words which are part of English language in order to complete sentence but in NLP we do not need that words for our Model. It reduces the Complexity of our text and also reduce computational time and cost.

```
def data_preprocessing(text):
  # Removing punctuatoins:

  punctuation_pattern = r'[!@#$%^&*()_+{}\[\]:;"\'<>,.?/\|\\-]'
  text=re.sub(punctuation_pattern,'', text)
  # Tokenize and lower case conversion:

  words=word_tokenize(text.lower())

  # Remove numbers using regular expression
  words = [word for word in words if not re.match(r'\d+', word)]
  # Applying stopwords:

  stop_words = set(stopwords.words('english'))
  words=[word for word in words if word not in stop_words]
  # Applying Stemming:

  stemmer = PorterStemmer()
  words = [stemmer.stem(word) for word in words]
  processed_text = ' '.join(words)

  return processed_text
df['Description']=df['Description'].apply(data_preprocessing)
df['Headline']=df['Headline'].apply(data_preprocessing)
df['Keywords']=df['Keywords'].apply(data_preprocessing)
df['Second headline']=df['Second headline'].apply(data_preprocessing)
```

Figure 6 Pre-Processing Code for IRS Model

```
[11] df
```

	Index	Author	Date published	Category	Section	Url	Headline	Description	Keywords	Second headline	Article text
0	0	Jacopo Prisco, CNN	2021-07-15 02:46:59	news	world	https://www.cnn.com/2021/07/14/world/tusimple-...	there shortag trucker tusimpl think solut driv...	ecommerc boom exacerb global truck driver shor...	world there shortag trucker tusimpl think solu...	there shortag trucker tusimpl think solut driv...	(CNN)Right now, there's a shortage of truck d...
1	2	Stephanie Bailey, CNN	2021-05-12 07:52:09	news	world	https://www.cnn.com/2021/05/12/world/ironhand-...	bioservo robot ironhand could protect factori ...	work factori mean task could lead chronic inju...	world bioservo robot ironhand could protect fa...	robot ironhand could protect factori worker in...	(CNN)Working in a factory or warehouse can me...
2	3	Words by Stephanie Bailey, video by Zahra Jamshed	2021-06-16 02:51:30	news	asia	https://www.cnn.com/2021/06/15/asia/swarm-robo...	swarm robot get smarter work cnn	hong kong warehous swarm autonom robot work th...	asia swarm robot get smarter work cnn	swarm robot get smarter work	(CNN)In a Hong Kong warehouse, a swarm of aut...
3	4	Paul R. La Monica, CNN Business	2022-03-15 09:57:36	business	investing	https://www.cnn.com/2022/03/15/investing/brics...	russia longer option investor emerg market cnn	mani year world popular emerg market socal bri...	invest russia longer option investor emerg mar...	russia longer option investor emerg market	New York (CNN Business)For many years, the wor...

Figure 7 Data After Pre-processing

VII. Feature Extraction through TF-IDF: -

Feature extraction is important part in NLP Projects and it involves extraction of the words which have meaning in the text. I use TF-IDF method which is Term frequency inverse Document Frequency. It is the model of Sklearn which performs following tasks,

- Convert text into Tokens
- Lower case the text and remove punctuations by default
- Calculate how many times a word occurs in document
- Find the Importance of that word in the Document
- Make a Vector based on above operations which have score of each word

In IRS system we use Article text column and apply TF-IDF method to convert into Vector which we then compare with User Query Vectors and find Cosine Score and display top 10 Documents for user.

VIII. Query Processing: -

Our next step is to process the user query and convert it into Vector and based on that query we fetch top 10 documents for user. We not only want to fetch but also rank the documents, the documents are shown in descending orders. The code is shown below,

Processing User Query for search:-

```
[14] def process_user_query(user_query):
         query_vector = tfidf_vectorizer.transform([user_query])
         return query_vector
```

Getting top 10 Ranked Articles from search:-

```
[15] def get_search_results(user_query, num_results=10):
         query_vector = process_user_query(user_query)
         cosine_similarities = cosine_similarity(query_vector, dtm)
         top_indices = cosine_similarities.argsort()[0][-num_results:][::-1]
         search_results = df.iloc[top_indices]
         return search_results
```

Displaying results with cosine_score:-

```
[16] user_query = input("Enter your Query tro search:- ")
     # Retrieve and display the top N documents as search results
     search_results = get_search_results(user_query)
     retrieved_paper_ids=[]
     for idx, row in search_results.iterrows():
         retrieved_paper_ids.append(row.Index)
         cosine_score = cosine_similarity(process_user_query(user_query), dtm[idx])
         print(f"Title: {row['Headline']}")
         print(f"index of Articles: {row['Index']}")
         print(f"Author: {row['Author']}")
         print(f"Date Published: {row['Date published']}")
         print(f"Description: {row['Description']}")
         print(f"URL For paper: {row['Url']}")
         print(f"Cosine Similarity Score: {float(cosine_score[0]):.4f}")
         print()
```

Figure 8 Query processing

```
Enter your Query tro search:- golf
Title: random golf club filmmak look solv problem feel unwelcom golf cnn
index of Articles: 1704
Author: Ben Morse, CNN
Date Published: 2021-02-15 09:35:56
Description: golf tradit suffer reput tough sport access
URL For paper: https://www.cnn.com/2021/02/15/golf/erik-anders-lang-random-golf-club-exeter-golf-spc-cmd-spt-intl/index.html
Cosine Similarity Score: 0.5447

Title: climat crisi golf cours borrow time earth weather pattern becom wild cnn
index of Articles: 4572
Author: CNN Sports staff
Date Published: 2021-12-08 09:01:51
Description: golf cours salt lake counti utah drink around nine million gallon water day stay pristin green that olympics swim pool
URL For paper: https://www.cnn.com/2021/12/08/golf/climate-change-sustainability-spt-intl-cmd/index.html
Cosine Similarity Score: 0.4691

Title: iceland could reshap world golf cnn
index of Articles: 584
Author: Sean Coppack, CNN
Date Published: 2021-11-04 09:46:52
Description: iceland may seem like unlik golf power
URL For paper: https://www.cnn.com/2021/11/04/golf/iceland-reshape-living-golf-spt-intl/index.html
Cosine Similarity Score: 0.4178

Title: hideki matsuyama becam japan new nation hero cnn
index of Articles: 2197
Author: Ben Morse, CNN
Date Published: 2021-04-17 23:26:09
Description: one small putt hideki matsuyama ascend profession golfer nation treasur
```

Figure 9 Output for Fetching Documents.

IX. Performance Evaluation: -

The next step is to evaluate the performance. For that we have to define test Queries and get the Documents ID against that related queries. Then we test our model on that query and get the ID which our model gives us. Then we compare and find precision. We make ground-truth name data frame for that using the codes below,

Perfomance Evaluation:-

```
[17] a=df[df['Section']=='golf']
     doc_golf_index=[]
     for i in a.index.values:
       doc_golf_index.append(i)

     b=df[df['Section']=='tennis']
     doc_tennis_index=[]
     for j in b.index.values:
       doc_tennis_index.append(j)

     c=df[df['Section']=='football']
     doc_football_index=[]
     for k in c.index.values:
       doc_football_index.append(k)

     d=df[df['Section']=='uk']
     doc_uk_index=[]
     for l in d.index.values:
       doc_uk_index.append(l)

     e=df[df['Section']=='business']
     doc_business_index=[]
     for m in e.index.values:
       doc_business_index.append(m)
```

Figure 10 Getting Documents ID.

```
f=df[df['Section']=='media']
doc_media_index=[]
for p in f.index.values:
  doc_media_index.append(p)
```

```
[19] import pandas as pd

     # Create a sample ground truth DataFrame
     data = {'Query': ['Playing golf', 'news about football',
                       'tennis papers','information about uk','is it good carrier to became a football player',
                       'how to start invest in business','media role in society'],
             'Relevant_Documents_ID': [doc_golf_index,doc_football_index,doc_tennis_index,
                                       doc_uk_index,doc_football_index,doc_business_index,doc_media_index]}
     ground_truth_df = pd.DataFrame(data)
```

Figure 11 Code to make Data Frame.

```
ground_truth_df
```

	Query	Relevant_Documents_ID
0	Playing golf	[76, 83, 90, 92, 93, 94, 308, 309, 310, 311, 3...
1	news about football	[81, 84, 95, 96, 97, 125, 142, 291, 292, 293, ...
2	tennis papers	[89, 91, 98, 99, 100, 306, 307, 778, 791, 808,...
3	information about uk	[770, 773, 779, 783, 794, 804, 812, 816, 821, ...
4	is it good carrier to became a football player	[81, 84, 95, 96, 97, 125, 142, 291, 292, 293, ...
5	how to start invest in business	[4, 11, 15, 17, 20, 24, 26, 32, 35, 37, 38, 39...
6	media role in society	[5, 6]

Figure 12 Ground-truth Data frame

Now next is to find Document ID for each test query. In order to achieve it we apply for Loop as shown in below code.

```
[21]  # Initialize empty lists to store results
     all_retrieved_paper_ids = []
     for i in ground_truth_df['Query']:
         user_query = i

         # Retrieve and display the top N documents as search results
         search_results = get_search_results(user_query)

         retrieved_paper_ids = []  # List to store retrieved document IDs
         for idx, row in search_results.iterrows():
             retrieved_paper_ids.append(row.Index)
             cosine_score = cosine_similarity(process_user_query(user_query), dtm[idx])

         # Store the retrieved document IDs and scores for this query
         all_retrieved_paper_ids.append(retrieved_paper_ids)

     # Add the obtained document IDs and scores to the ground truth DataFrame
     ground_truth_df['Obtained_Doc_ID'] = all_retrieved_paper_ids
```

Figure 13 Code to get Documents ID.

```
ground_truth_df
```

	Query	Relevant_Documents_ID	Obtained_Doc_ID
0	Playing golf	[76, 83, 90, 92, 93, 94, 308, 309, 310, 311, 3...	[1704, 4572, 584, 2197, 2260, 1835, 2237, 2276...
1	news about football	[81, 84, 95, 96, 97, 125, 142, 291, 292, 293, ...	[2166, 4028, 3897, 2171, 2079, 2185, 2817, 409...
2	tennis papers	[89, 91, 98, 99, 100, 306, 307, 778, 791, 808,...	[4165, 1432, 2978, 2859, 2375, 1736, 1439, 471...
3	information about uk	[770, 773, 779, 783, 794, 804, 812, 816, 821, ...	[1952, 1683, 2679, 3856, 3943, 1401, 1418, 283...
4	is it good carrier to became a football player	[81, 84, 95, 96, 97, 125, 142, 291, 292, 293, ...	[2125, 1577, 2418, 4227, 3897, 4028, 4018, 216...
5	how to start invest in business	[4, 11, 15, 17, 20, 24, 26, 32, 35, 37, 38, 39...	[35, 346, 962, 1030, 19, 40, 21, 182, 1928, 4]
6	media role in society	[5, 6]	[4399, 208, 2133, 4236, 2577, 4459, 1398, 4568...

Figure 14 Data Frame for ground truth

Next step is to find the precision for the model, for that we find True Positive and True Negative values by comparing the id in the ground-truth data frame. The code is shown below which compare each retrieved id with original document id and then find precision using precision formula. We get the low precision because we have only 4000 samples.

```
TP = 0
FP = 0
FN = 0

# Calculate TP, FP, and FN
for j in ground_truth_df.index:
    for doc_id in ground_truth_df['Obtained_Doc_ID'][j]:
        relevant_docs = ground_truth_df['Relevant_Documents_ID'][j]
        if doc_id in relevant_docs:
            TP += 1
        else:
            FP += 1

for M in ground_truth_df.index:
    for id in ground_truth_df['Relevant_Documents_ID'][M]:
        obtained_docs = ground_truth_df['Obtained_Doc_ID'][M]
        if id not in obtained_docs:
            FN += 1

# Calculate Precision, Recall, and F1-Score
Precision = TP / (TP + FP)
Recall = TP / (TP + FN)
# Calculate Precision, Recall, and F1-Score
if (Precision +Recall) != 0:
    f1_score = 2 * (Precision * Recall) / (Precision+ Recall)
else:
    f1_score = 0.0

print(f"Precision: {Precision:.2f}")
print(f"Recall: {Recall:.2f}")
print(f"F1-Score: {f1_score:.2f}")
```

Figure 15 Code to find precision

```
Precision: 0.07
Recall: 0.00
F1-Score: 0.00
```