Hybrid System for Research Paper Recommendation and Text Summarization

Chitransh Bose
chitransh22096@iiitd.ac.in
IIITD
Ranit Pal
ranit22119@iiitd.ac.in
IIITD
Soumyajyoti Das
soumyajyoti22075@iiitd.ac
.in
IIITD
Varun Jhunjhunwala
varun22087@iiitd.ac.in
IIITD

1. Updated problem formation and literature review

1.1 Problem formation

We are working with a “Hybrid system for Research paper recommendation and Text summarization” that will be basically recommend the user with the top ‘k’ research paper based on the user input along with the short summary. Thus our work will be divided into two parts first will be recommendation of relevant research papers and the other will be summarization of text. We are currently working with the Content-Based filtering system and have tried to extend it using collaborative filtering. The previous works have primarily used content-based filtering and collaborative-based filtering alone for the recommendations and we are trying to incorporate both of them to get better results.

1.2 Literature review

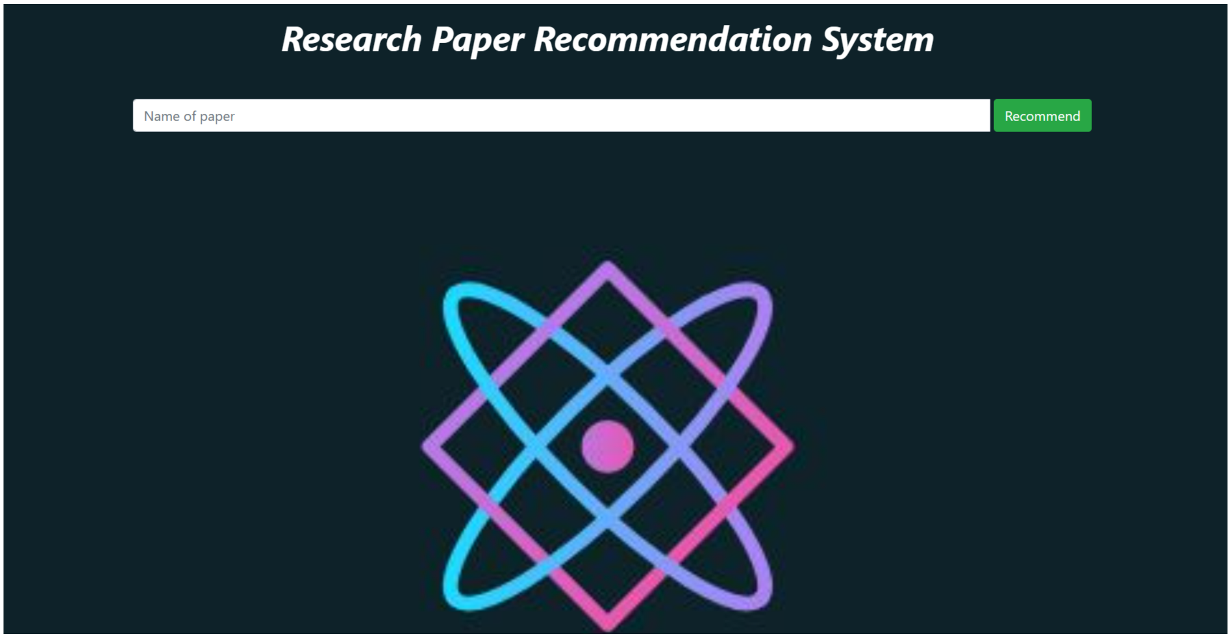
Joeran Beal et al. in their paper have provided a survey based on a research paper recommendation system, in which they have identified the methods which are mostly used for recommendations along with the various shortcomings that are present in various papers like the choice of evaluation metrics, etc. Joonseok Lee et al., have proposed a Personalized Academic Research Paper recommendation system that recommends the use of articles and papers related to the interest of the user by finding the similarity between the text using collaborative filtering. As there is a lot of information present over the internet these days, the extraction of relevant documents and further the relevant information is very important. Thus, S.A. Babar et al. in their paper have presented a system for text summarization. They have discussed various text summarization methods like TF-IDF, Graph-theoretic approach, Machine Learning approach, and Automatic Text Summarization based on Fuzzy logic. Julian Kupiec et al. proposes a trainable document summarizer that uses a neural network architecture to generate summaries of input documents. They evaluated their approach on a benchmark dataset and compared it to existing summarization systems. S.M. Kamruzzaman et al. in their work have proposed a new algorithm for the classification of text that is based on the word relation rather than the word only using a lesser number of documents for training. Further, it uses Naïve Bayes for extracting the features. Arun Kumar Yadav et.al has used reinforcement learning as their base model and on the basis of that they have proposed their algorithm which they have evaluated using Bilingual Evaluation Understudy (BLEU) and Recall-Oriented Understudy for Gisting Evaluation (ROUGE).

2. Updated baseline results

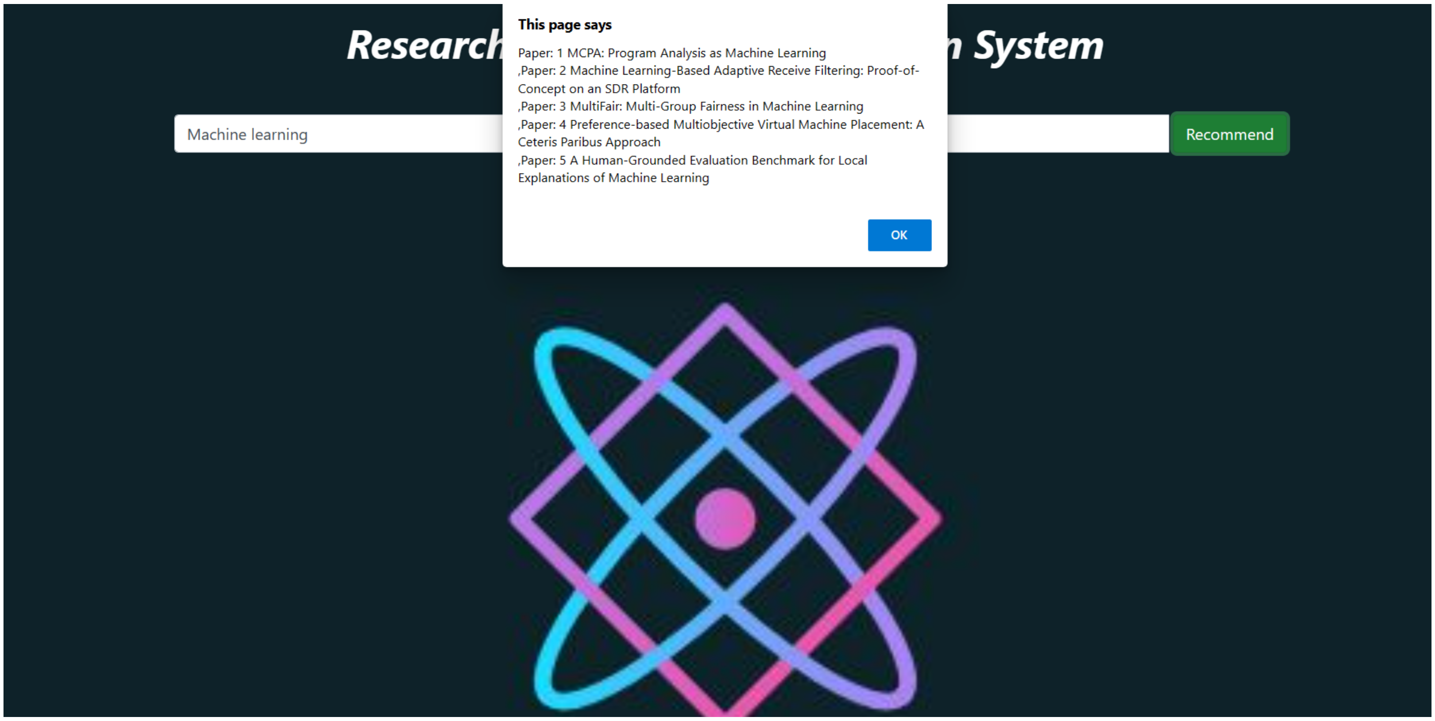
Our baseline results were based on the dataset that contained 2000 entries of files in a CSV format and now we have moved to a large dataset which is around 10000 files and is close to the original dataset that a system may get i.e. PDFs. We have extracted the title and abstract from around 8000 files as of now and are working on the extraction of complete dataset and have applied content-based filtering to that data. As a part of the evaluation, we have done user testing on around 10 users as of now and have found that we are able to recommend 3-4 correct recommendations out of five recommendations to each user. Also as of now, we have developed a static user interface which we will convert to dynamic if we convert our project to engineering based. We have also tried to recommend the papers using collaborative based filtering but we have observed that the results were very poor as compared to content based filtering. This might be because there is no approach to connect two or more users based on their search interests. Thus it is not possible to use it as of now.

3. Proposed method

As there are challenges with the recommendation system using collaborative filtering, therefore, we will primarily be using content based filtering method as it is more relevant to our dataset and work. We will also apply ranking methods such as Tf-IDF, Tf-ICF, and nDCG. Then on that output, the system will recommend the top results based on the threshold. Those recommendations will work as input for the text summarization framework which will be part of our final submission. We have also developed a prototype(static UI) if we change our project to engineering based in the future. It will take the name of the topic from the user and recommend the most related papers to the user along with a short summary of the complete paper.



Prototype static screen.

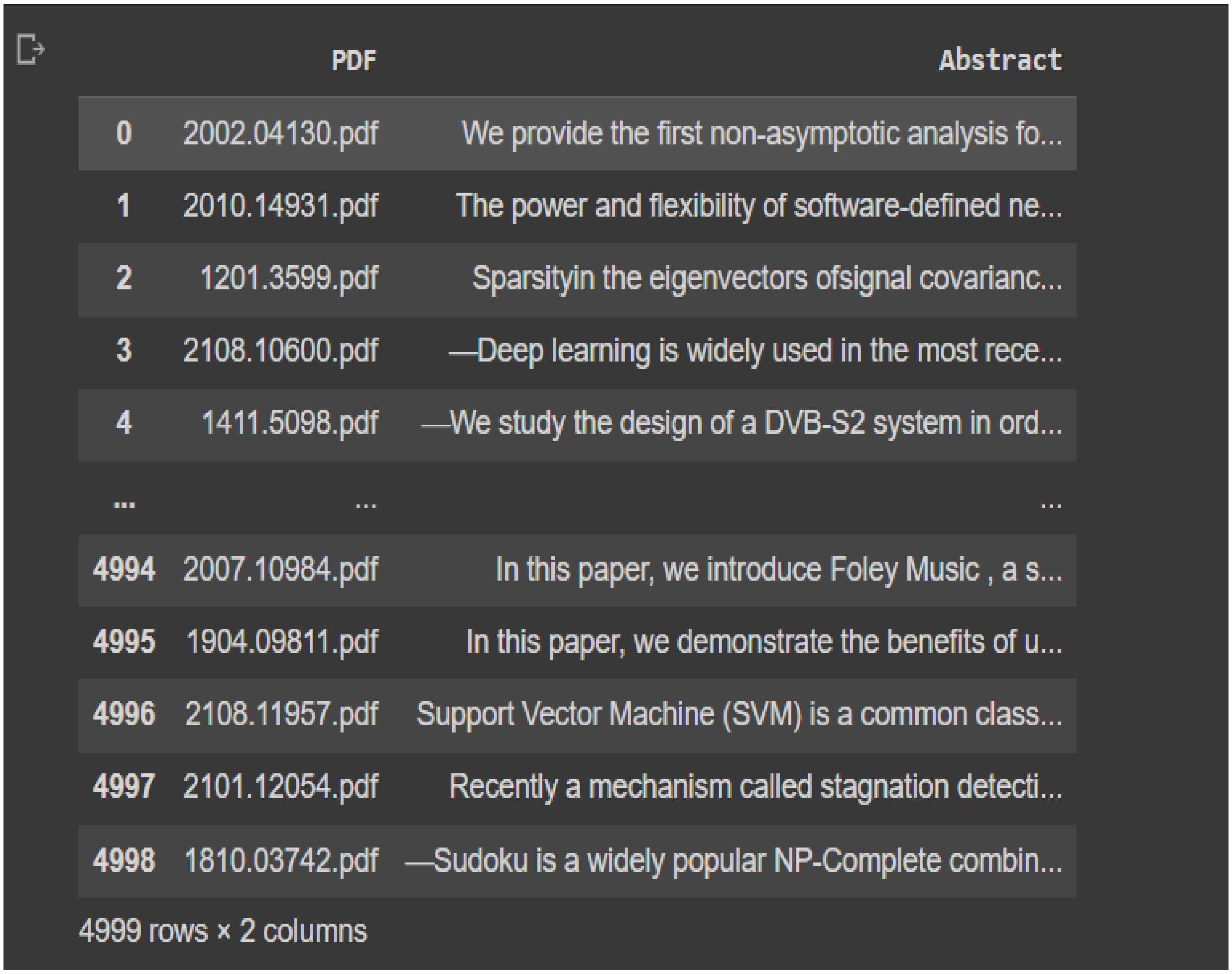


Machine learning

This page says

Paper: 1 MCPA: Program Analysis as Machine Learning
,Paper: 2 Machine Learning-Based Adaptive Receive Filtering: Proof-of-Concept on an SDR Platform
,Paper: 3 MultiFair: Multi-Group Fairness in Machine Learning
,Paper: 4 Preference-based Multiobjective Virtual Machine Placement: A Ceteris Paribus Approach
,Paper: 5 A Human-Grounded Evaluation Benchmark for Local Explanations of Machine Learning

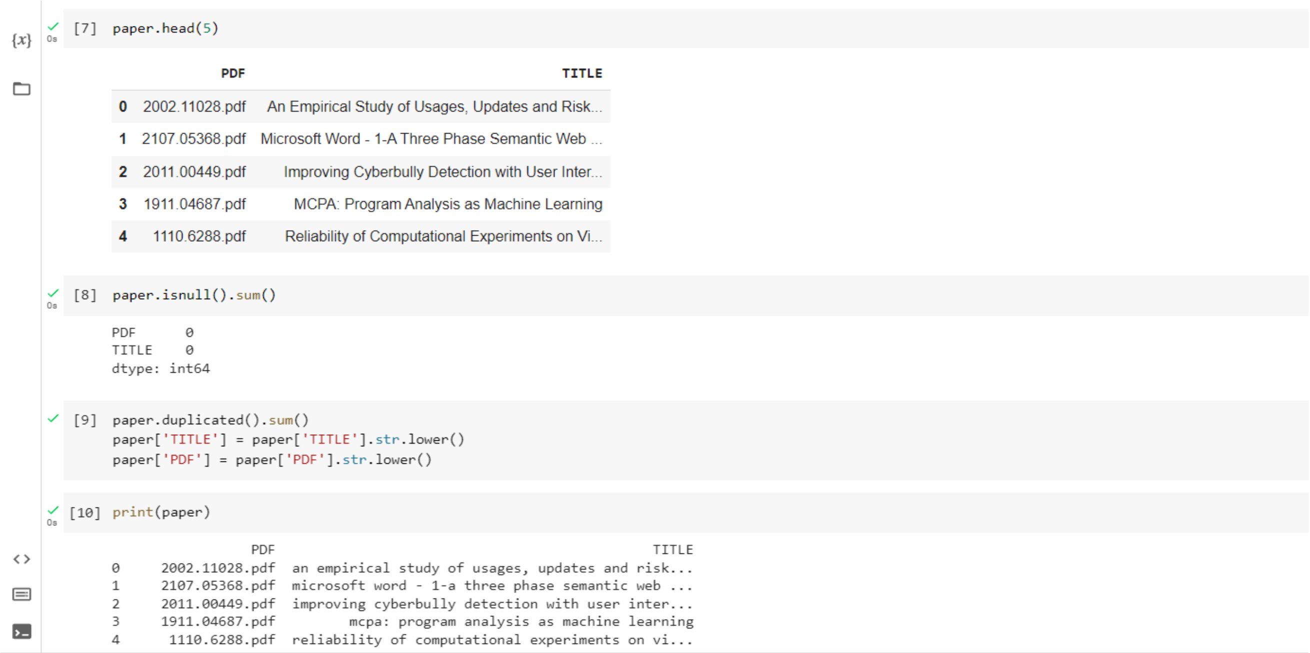
Sample output from static database.



```
                                                      PDF                                              Abstract
   0     2002.04130.pdf      We provide the first non-asymptotic analysis fo...
   1     2010.14931.pdf      The power and flexibility of software-defined ne...
   2      1201.3599.pdf      Sparsityin the eigenvectors ofsignal covarianc...
   3     2108.10600.pdf      —Deep learning is widely used in the most rece...
   4      1411.5098.pdf      —We study the design of a DVB-S2 system in ord...
  ...              ...                                                   ...
4994     2007.10984.pdf      In this paper, we introduce Foley Music , a s...
4995     1904.09811.pdf      In this paper, we demonstrate the benefits of u...
4996     2108.11957.pdf      Support Vector Machine (SVM) is a common class...
4997     2101.12054.pdf      Recently a mechanism called stagnation detecti...
4998     1810.03742.pdf      —Sudoku is a widely popular NP-Complete combin...
4999 rows × 2 columns
```

Sample extracted abstracts

RECOMMENDATION CODE:



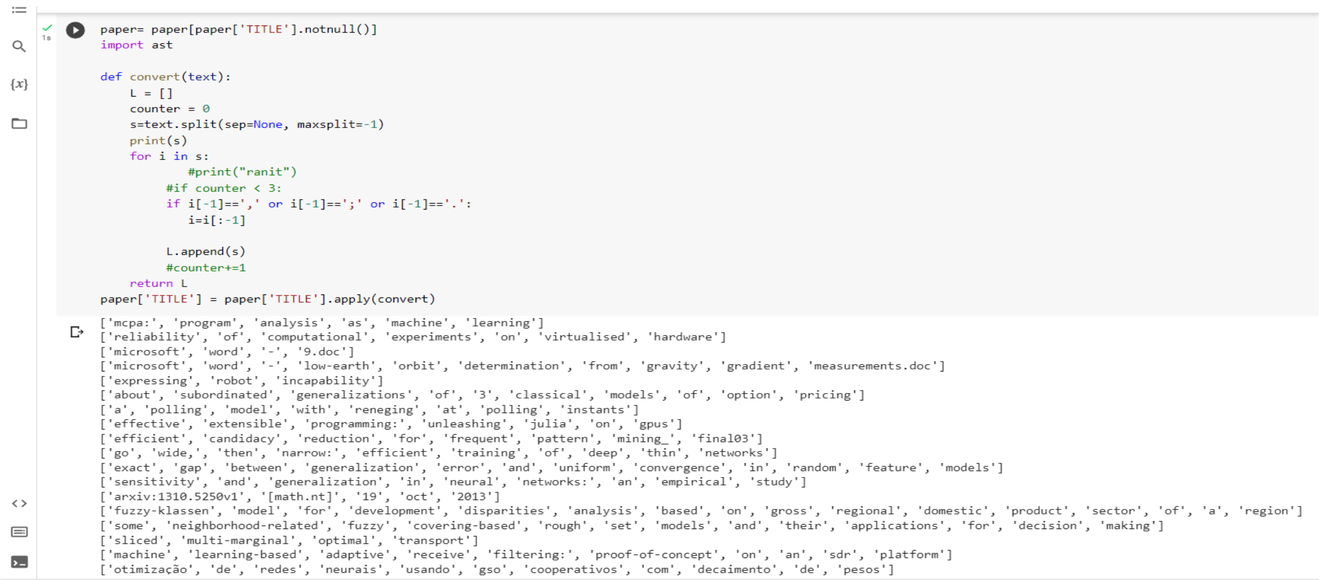
```
[7] paper.head(5)
```

	PDF	TITLE
0	2002.11028.pdf	An Empirical Study of Usages, Updates and Risk...
1	2107.05368.pdf	Microsoft Word - 1-A Three Phase Semantic Web ...
2	2011.00449.pdf	Improving Cyberbully Detection with User Inter...
3	1911.04687.pdf	MCPA: Program Analysis as Machine Learning
4	1110.6288.pdf	Reliability of Computational Experiments on Vi...

```
[8] paper.isnull().sum()

PDF      0
TITLE    0
dtype: int64
```

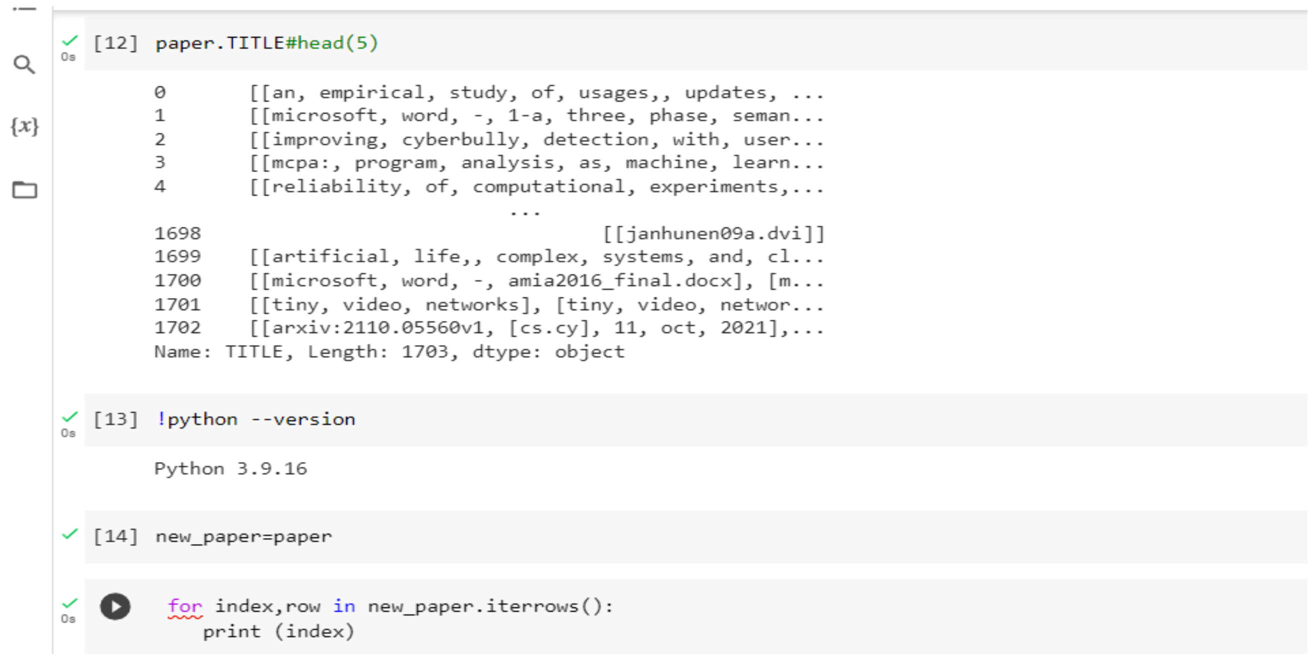
```
[9] paper.duplicated().sum()
    paper['TITLE'] = paper['TITLE'].str.lower()
    paper['PDF'] = paper['PDF'].str.lower()
```

```
[10] print(paper)

                 PDF                                              TITLE
0     2002.11028.pdf  an empirical study of usages, updates and risk...
1     2107.05368.pdf  microsoft word - 1-a three phase semantic web ...
2     2011.00449.pdf  improving cyberbully detection with user inter...
3     1911.04687.pdf         mcpa: program analysis as machine learning
4      1110.6288.pdf  reliability of computational experiments on vi...
```



```
paper= paper[paper['TITLE'].notnull()]
import ast

def convert(text):
    L = []
    counter = 0
    s=text.split(sep=None, maxsplit=-1)
    print(s)
    for i in s:
            #print("ranit")
        #if counter < 3:
        if i[-1]==',' or i[-1]==';' or i[-1]=='.':
            i=i[:-1]

        L.append(s)
        #counter+=1
    return L
paper['TITLE'] = paper['TITLE'].apply(convert)
```

```
['mcpa:', 'program', 'analysis', 'as', 'machine', 'learning']
['reliability', 'of', 'computational', 'experiments', 'on', 'virtualised', 'hardware']
['microsoft', 'word', '-', '9.doc']
['microsoft', 'word', '-', 'low-earth', 'orbit', 'determination', 'from', 'gravity', 'gradient', 'measurements.doc']
['expressing', 'robot', 'incapability']
['about', 'subordinated', 'generalizations', 'of', '3', 'classical', 'models', 'of', 'option', 'pricing']
['a', 'polling', 'model', 'with', 'reneging', 'at', 'polling', 'instants']
['effective', 'extensible', 'programming:', 'unleashing', 'julia', 'on', 'gpus']
['efficient', 'candidacy', 'reduction', 'for', 'frequent', 'pattern', 'mining_', 'final03']
['go', 'wide,', 'then', 'narrow:', 'efficient', 'training', 'of', 'deep', 'thin', 'networks']
['exact', 'gap', 'between', 'generalization', 'error', 'and', 'uniform', 'convergence', 'in', 'random', 'feature', 'models']
['sensitivity', 'and', 'generalization', 'in', 'neural', 'networks:', 'an', 'empirical', 'study']
['arxiv:1310.5250v1', '[math.nt]', '19', 'oct', '2013']
['fuzzy-klassen', 'model', 'for', 'development', 'disparities', 'analysis', 'based', 'on', 'gross', 'regional', 'domestic', 'product', 'sector', 'of', 'a', 'region']
['some', 'neighborhood-related', 'fuzzy', 'covering-based', 'rough', 'set', 'models', 'and', 'their', 'applications', 'for', 'decision', 'making']
['sliced', 'multi-marginal', 'optimal', 'transport']
['machine', 'learning-based', 'adaptive', 'receive', 'filtering:', 'proof-of-concept', 'on', 'an', 'sdr', 'platform']
['otimização', 'de', 'redes', 'neurais', 'usando', 'gso', 'cooperativos', 'com', 'decaimento', 'de', 'pesos']
```

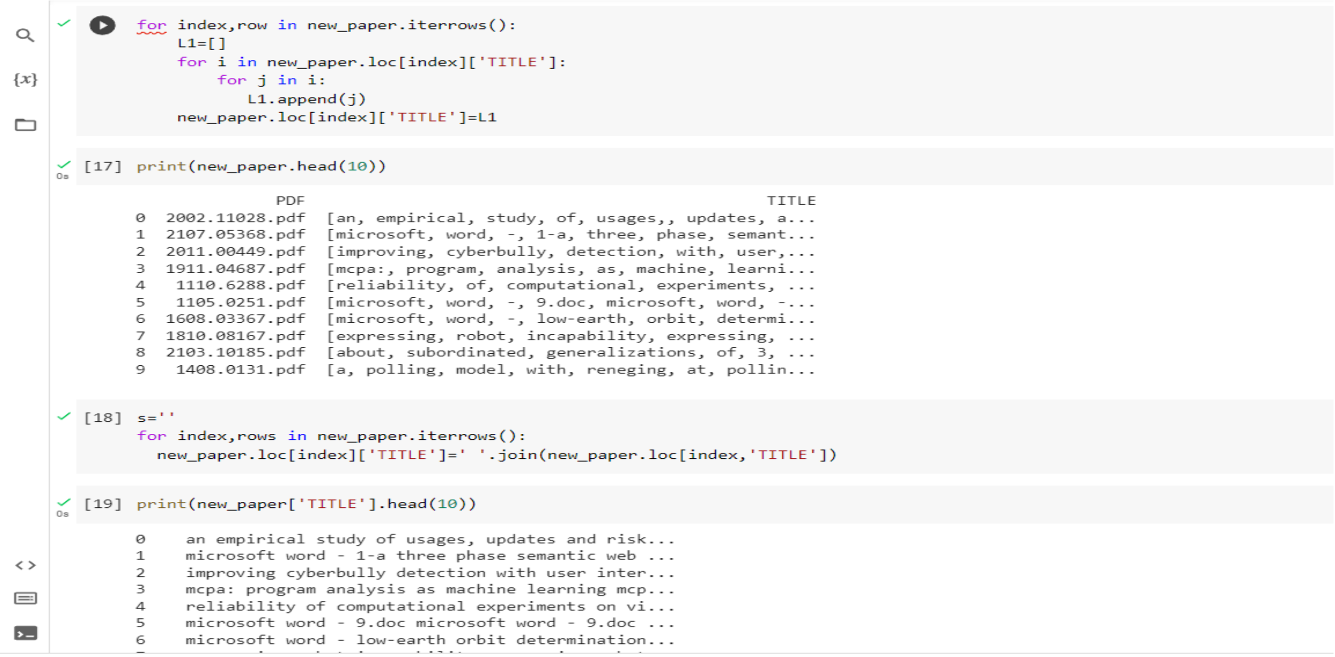


```
[12] paper.TITLE#head(5)

0       [[an, empirical, study, of, usages,, updates, ...
1       [[microsoft, word, -, 1-a, three, phase, seman...
2       [[improving, cyberbully, detection, with, user...
3       [[mcpa:, program, analysis, as, machine, learn...
4       [[reliability, of, computational, experiments,...
                              ...
1698                                  [[janhunen09a.dvi]]
1699    [[artificial, life,, complex, systems, and, cl...
1700    [[microsoft, word, -, amia2016_final.docx], [m...
1701    [[tiny, video, networks], [tiny, video, networ...
1702    [[arxiv:2110.05560v1, [cs.cy], 11, oct, 2021],...
Name: TITLE, Length: 1703, dtype: object
```

```
[13] !python --version

Python 3.9.16
```

```
[14] new_paper=paper
```

```
for index,row in new_paper.iterrows():
   print (index)
```

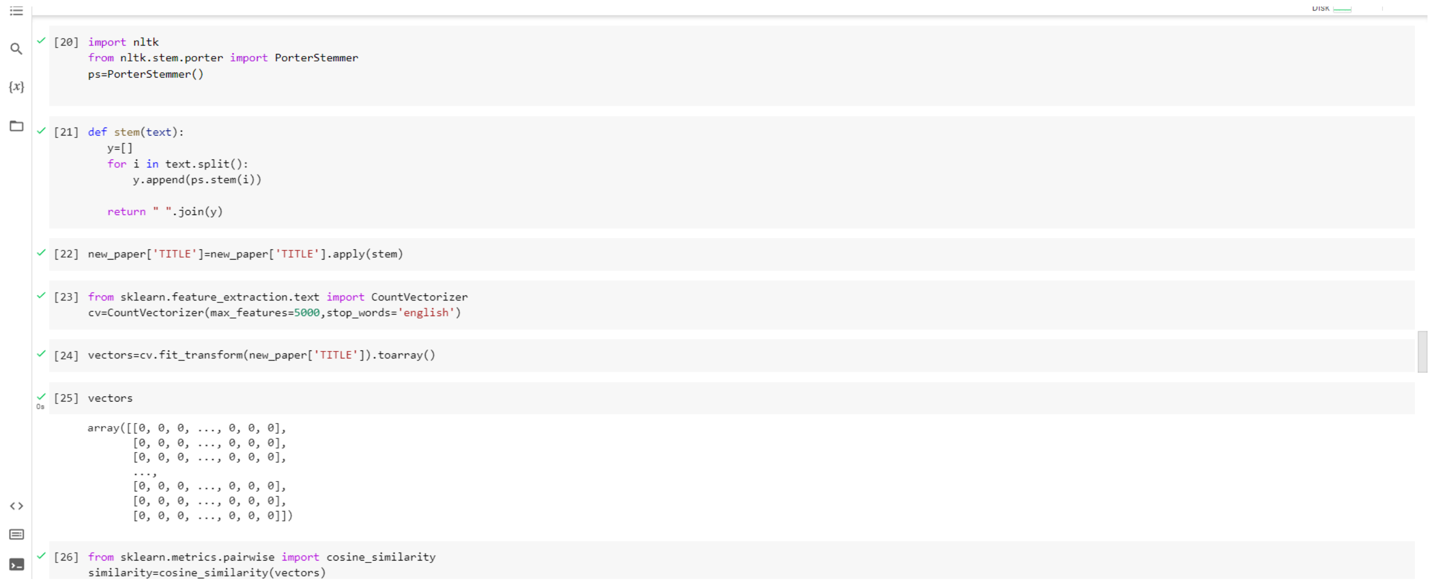


```
for index,row in new_paper.iterrows():
    L1=[]
    for i in new_paper.loc[index]['TITLE']:
        for j in i:
            L1.append(j)
    new_paper.loc[index]['TITLE']=L1
```

```
[17] print(new_paper.head(10))

                PDF                                              TITLE
0  2002.11028.pdf  [an, empirical, study, of, usages,, updates, a...
1  2107.05368.pdf  [microsoft, word, -, 1-a, three, phase, semant...
2  2011.00449.pdf  [improving, cyberbully, detection, with, user,...
3  1911.04687.pdf  [mcpa:, program, analysis, as, machine, learni...
4   1110.6288.pdf  [reliability, of, computational, experiments, ...
5   1105.0251.pdf  [microsoft, word, -, 9.doc, microsoft, word, -...
6  1608.03367.pdf  [microsoft, word, -, low-earth, orbit, determi...
7  1810.08167.pdf  [expressing, robot, incapability, expressing, ...
8  2103.10185.pdf  [about, subordinated, generalizations, of, 3, ...
9   1408.0131.pdf  [a, polling, model, with, reneging, at, pollin...
```

```
[18] s=''
     for index,rows in new_paper.iterrows():
       new_paper.loc[index]['TITLE']=' '.join(new_paper.loc[index,'TITLE'])
```

```
[19] print(new_paper['TITLE'].head(10))

0     an empirical study of usages, updates and risk...
1     microsoft word - 1-a three phase semantic web ...
2     improving cyberbully detection with user inter...
3     mcpa: program analysis as machine learning mcp...
4     reliability of computational experiments on vi...
5     microsoft word - 9.doc microsoft word - 9.doc ...
6     microsoft word - low-earth orbit determination...
```



```
[20] import nltk
     from nltk.stem.porter import PorterStemmer
     ps=PorterStemmer()
```

```
[21] def stem(text):
       y=[]
       for i in text.split():
           y.append(ps.stem(i))

       return " ".join(y)
```

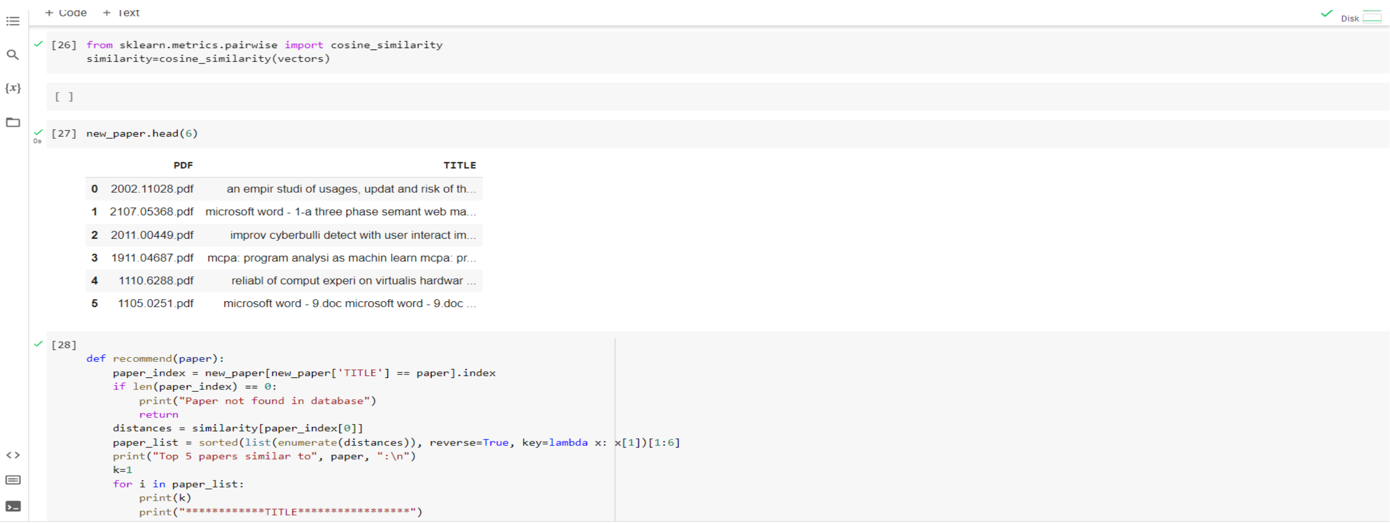
```
[22] new_paper['TITLE']=new_paper['TITLE'].apply(stem)
```

```
[23] from sklearn.feature_extraction.text import CountVectorizer
     cv=CountVectorizer(max_features=5000,stop_words='english')
```

```
[24] vectors=cv.fit_transform(new_paper['TITLE']).toarray()
```

```
[25] vectors

     array([[0, 0, 0, ..., 0, 0, 0],
            [0, 0, 0, ..., 0, 0, 0],
            [0, 0, 0, ..., 0, 0, 0],
            ...,
            [0, 0, 0, ..., 0, 0, 0],
            [0, 0, 0, ..., 0, 0, 0],
            [0, 0, 0, ..., 0, 0, 0]])
```

```
[26] from sklearn.metrics.pairwise import cosine_similarity
     similarity=cosine_similarity(vectors)
```



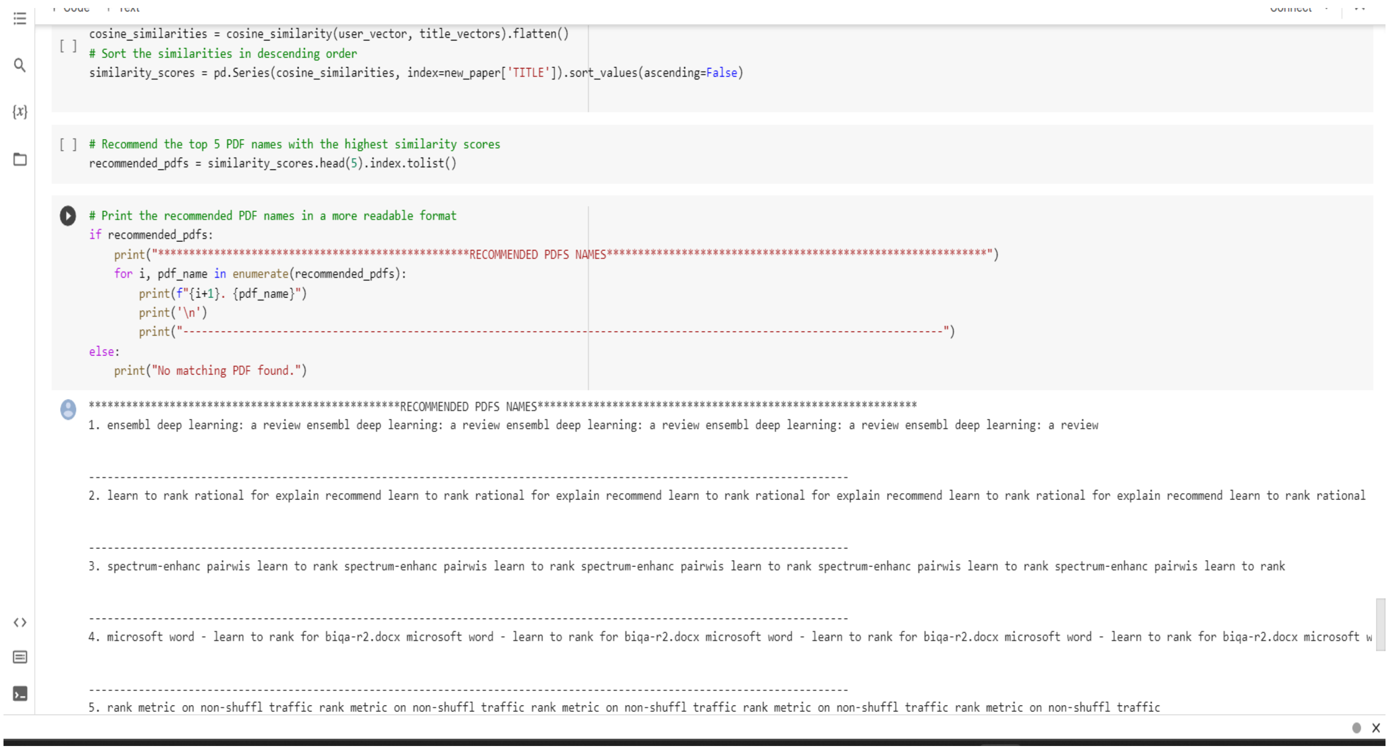
```
[26] from sklearn.metrics.pairwise import cosine_similarity
     similarity=cosine_similarity(vectors)
```

```
[ ]
```

```
[27] new_paper.head(6)
```

	PDF	TITLE
0	2002.11028.pdf	an empir studi of usages, updat and risk of th...
1	2107.05368.pdf	microsoft word - 1-a three phase semant web ma...
2	2011.00449.pdf	improv cyberbulli detect with user interact im...
3	1911.04687.pdf	mcpa: program analysi as machin learn mcpa: pr...
4	1110.6288.pdf	reliabl of comput experi on virtualis hardwar ...
5	1105.0251.pdf	microsoft word - 9.doc microsoft word - 9.doc ...

```
[28]
def recommend(paper):
    paper_index = new_paper[new_paper['TITLE'] == paper].index
    if len(paper_index) == 0:
        print("Paper not found in database")
        return
    distances = similarity[paper_index[0]]
    paper_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]
    print("Top 5 papers similar to", paper, ":\n")
    k=1
    for i in paper_list:
        print(k)
        print("************TITLE*****************")
```



```
[ ]
cosine_similarities = cosine_similarity(user_vector, title_vectors).flatten()
# Sort the similarities in descending order
similarity_scores = pd.Series(cosine_similarities, index=new_paper['TITLE']).sort_values(ascending=False)
```

```
[ ]
# Recommend the top 5 PDF names with the highest similarity scores
recommended_pdfs = similarity_scores.head(5).index.tolist()
```

```
# Print the recommended PDF names in a more readable format
if recommended_pdfs:
    print("**************************************************RECOMMENDED PDFS NAMES**************************************************************")
    for i, pdf_name in enumerate(recommended_pdfs):
        print(f"{i+1}. {pdf_name}")
        print('\n')
        print("----------------------------------------------------------------------------------------------------------------------------")
else:
    print("No matching PDF found.")
```

```
**************************************************RECOMMENDED PDFS NAMES**************************************************************
1. ensembl deep learning: a review ensembl deep learning: a review ensembl deep learning: a review ensembl deep learning: a review ensembl deep learning: a review


----------------------------------------------------------------------------------------------------------------------------
2. learn to rank rational for explain recommend learn to rank rational for explain recommend learn to rank rational for explain recommend learn to rank rational for explain recommend learn to rank rational


----------------------------------------------------------------------------------------------------------------------------
3. spectrum-enhanc pairwis learn to rank spectrum-enhanc pairwis learn to rank spectrum-enhanc pairwis learn to rank spectrum-enhanc pairwis learn to rank spectrum-enhanc pairwis learn to rank


----------------------------------------------------------------------------------------------------------------------------
4. microsoft word - learn to rank for biqa-r2.docx microsoft word - learn to rank for biqa-r2.docx microsoft word - learn to rank for biqa-r2.docx microsoft word - learn to rank for biqa-r2.docx microsoft w


----------------------------------------------------------------------------------------------------------------------------
5. rank metric on non-shuffl traffic rank metric on non-shuffl traffic rank metric on non-shuffl traffic rank metric on non-shuffl traffic rank metric on non-shuffl traffic
```