RECIPROCAL JOB RECOMMENDER SYSTEM

KARAMJOT KAUR RITIKA MOHANTY STUTI AGGARWAL

RJRS

ISSUES ADDRESSED SO FAR

- Extract the information of jobseekers from their uploaded resumes
- Matching jobs to jobseekers based on skills of job seekers

NEW LEARNINGS

- Content-Based Recommendation
- N-gram Language model
- Count vectorization
- TF-IDF Vectorization

CONTENT BASED RECOMMENDATION

jobseeker resume

extracted skills

job description with similar requirement

Steps

01

Upload resumes from device

02

Parsing resumes with Resume Parser Library

03

Appending parsed resume to resume dataset

04

Extracting skills from resume

05

Create bag of words of various job description dataset

06

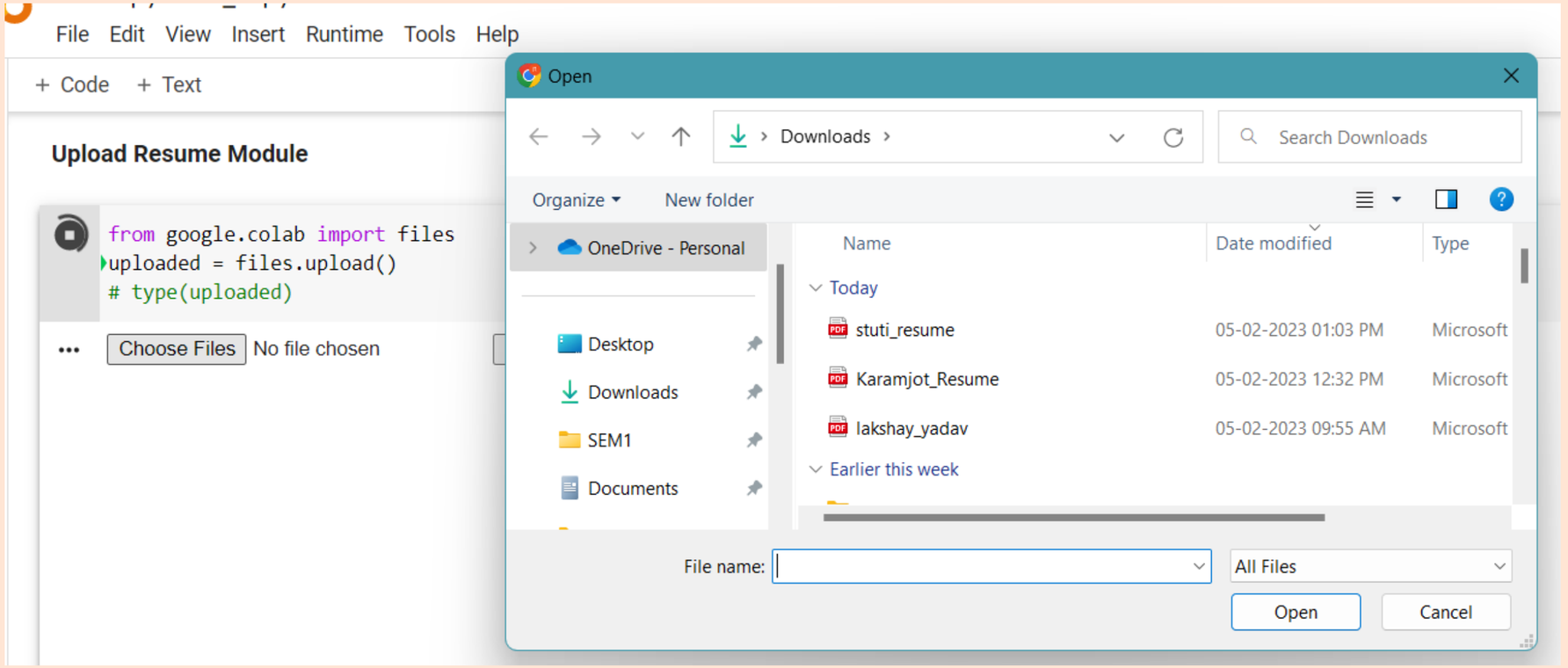
Recommend jobs based on similarity of skills and job

UPLOAD

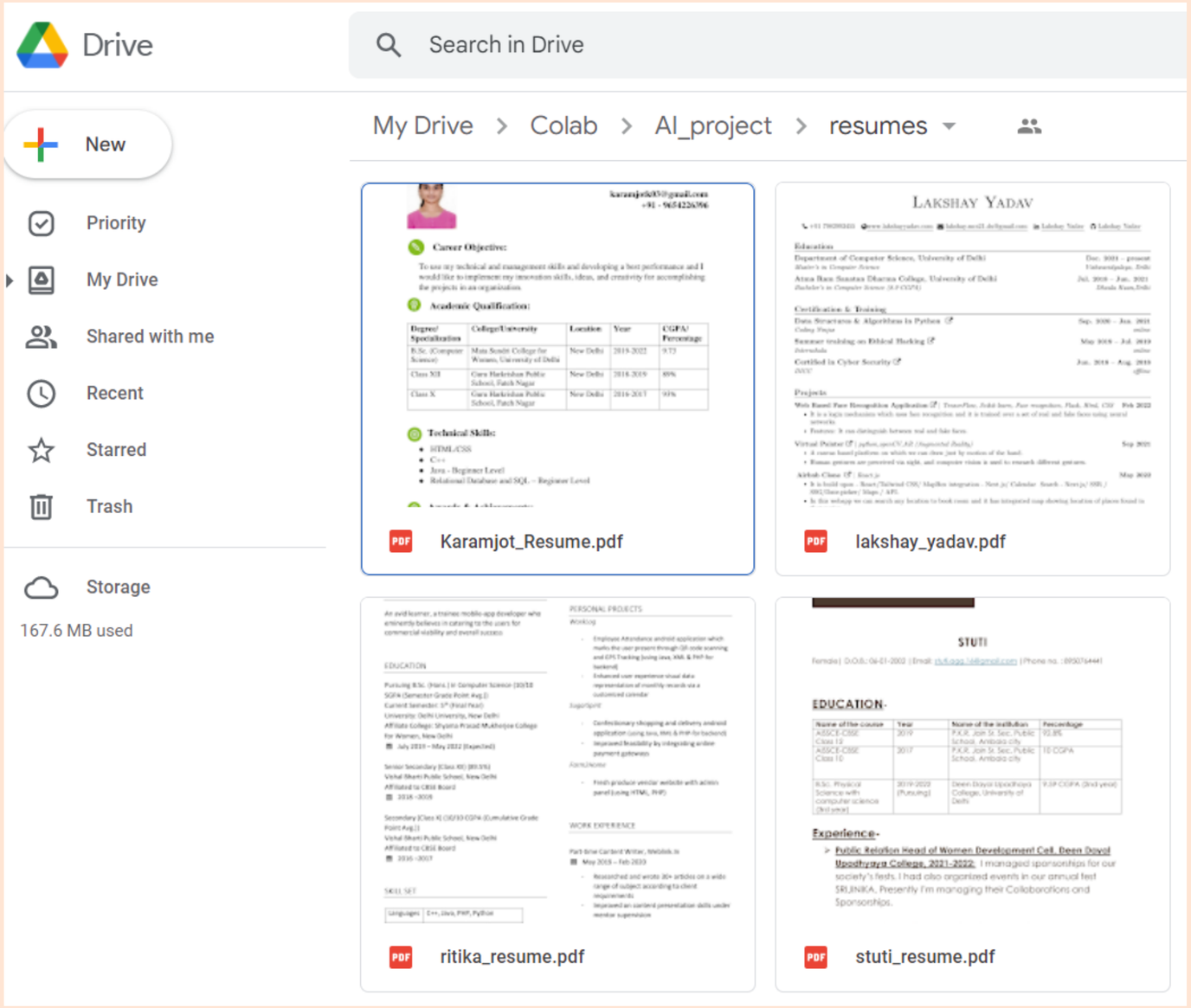
```
from google.colab import files
uploaded = files.upload()
# type(uploaded)
```

Choose Files No file chosen Upload widget is only availabl

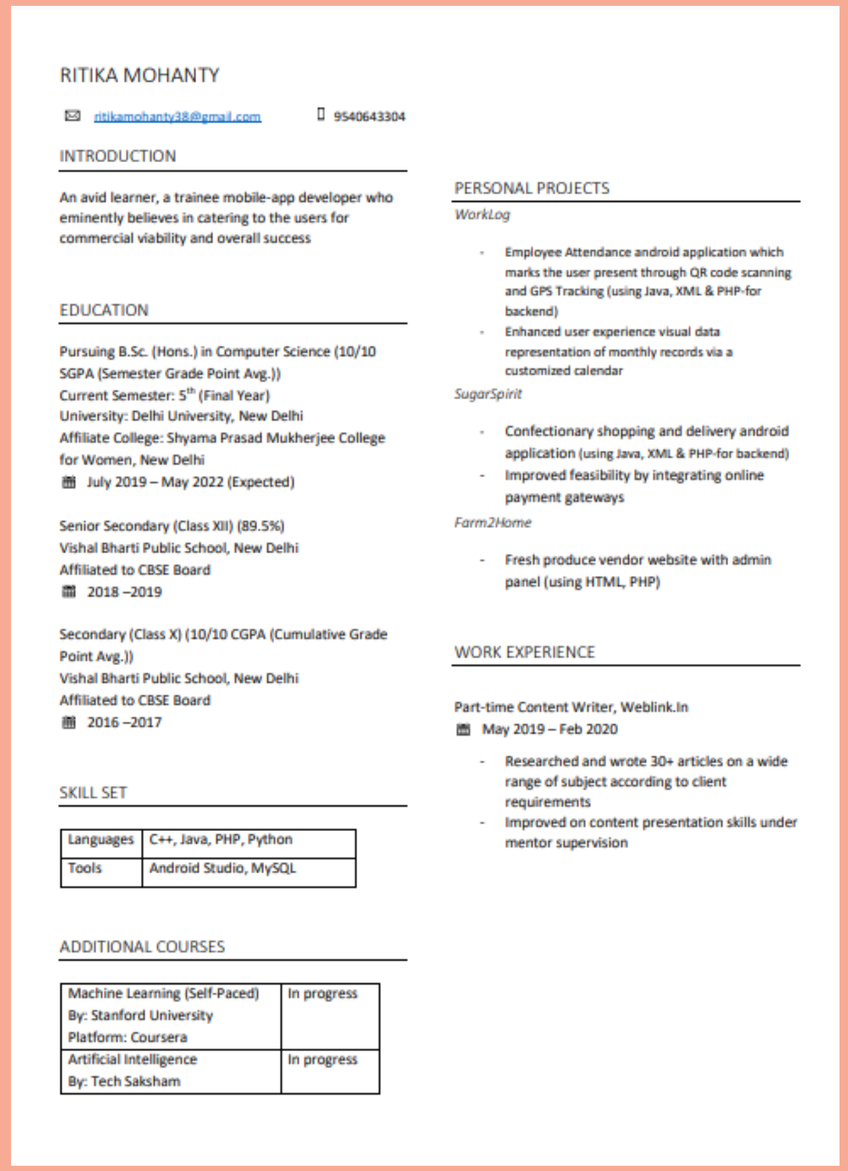
Saving resume.pdf to resume.pdf



STORE


Drive
Search in Drive
New
Priority
My Drive
Shared with me
Recent
Starred
Trash
Storage
167.6 MB used
My Drive > Colab > AI_project > resumes
Karamjot_Resume.pdf
lakshay_yadav.pdf
ritika_resume.pdf
stuti_resume.pdf

PARSE



RITIKA MOHANTY

ritikamohanty38@gmail.com 9540643304

INTRODUCTION

An avid learner, a trainee mobile-app developer who eminently believes in catering to the users for commercial viability and overall success

EDUCATION

Pursuing B.Sc. (Hons.) in Computer Science (10/10 SGPA (Semester Grade Point Avg.))
Current Semester: 5th (Final Year)
University: Delhi University, New Delhi
Affiliate College: Shyama Prasad Mukherjee College for Women, New Delhi
July 2019 – May 2022 (Expected)

Senior Secondary (Class XII) (89.5%)
Vishal Bharti Public School, New Delhi
Affiliated to CBSE Board
2018 –2019

Secondary (Class X) (10/10 CGPA (Cumulative Grade Point Avg.))
Vishal Bharti Public School, New Delhi
Affiliated to CBSE Board
2016 –2017

SKILL SET

Languages	C++, Java, PHP, Python
Tools	Android Studio, MySQL

ADDITIONAL COURSES

Machine Learning (Self-Paced) By: Stanford University Platform: Coursera	In progress
Artificial Intelligence By: Tech Saksham	In progress

PERSONAL PROJECTS

WorkLog

- Employee Attendance android application which marks the user present through QR code scanning and GPS Tracking (using Java, XML & PHP-for backend)
- Enhanced user experience visual data representation of monthly records via a customized calendar

SugarSpirit

- Confectionary shopping and delivery android application (using Java, XML & PHP-for backend)
- Improved feasibility by integrating online payment gateways

Farm2Home

- Fresh produce vendor website with admin panel (using HTML, PHP)

WORK EXPERIENCE

Part-time Content Writer, Weblink.In
May 2019 – Feb 2020

- Researched and wrote 30+ articles on a wide range of subject according to client requirements
- Improved on content presentation skills under mentor supervision

```
[127] # d1 ={}
      counter =0
      for key, val in data.items():
          print(key," : ",val)
          # d1.update( {key : listd1[counter]} )
          # counter = counter+1
```

```
name  :  RITIKA MOHANTY
email  :  ritikamohanty38@gmail.com
mobile_number  :  9540643304
skills  :  ['Content', 'Android', 'Java', 'Visual', 'Affilia
college_name  :  None
degree  :  None
designation  :  ['Machine Learning (Self-Paced) \nBy: Stanfo
experience  :  ['Part-time Content Writer, Weblink.In', 'May
company_names  :  None
no_of_pages  :  1
total_experience  :  0.75
```

APPEND

```
df =pd.read_csv('/content/drive/MyDrive/Colab/AI_project/resumes/resume_dataset.csv')
df.head()
```

	name	email	mobile_number	skills	college_name	degree	designation	
0	Lakshay Yadav	lakshay.mcs21.du@gmail.com	7982993455	['Engineering', 'Java', 'Tensorflow', 'Researc...	NaN	['Department of Computer Science', 'Bachelor's...	NaN	
1	STUTI EDUCATION-	stuti.agg.16@gmail.com	8950764441	['Digital marketing', 'Excel', 'Photoshop', 'P...	NaN	NaN	NaN	['rea 'Ac
2	KARAMJOT KAUR	karamjotk03@gmail.com	9654226396	['Engineering', 'Java', 'Database', 'Documenta...	NaN	NaN	NaN	
3	RITIKA MOHANTY	ritikamohanty38@gmail.com	9540643304	['Content', 'Android', 'Java', 'Visual', 'Affi...	NaN	NaN	['Machine Learning (Self-Paced) \nBy: Stanford...	['Part Write

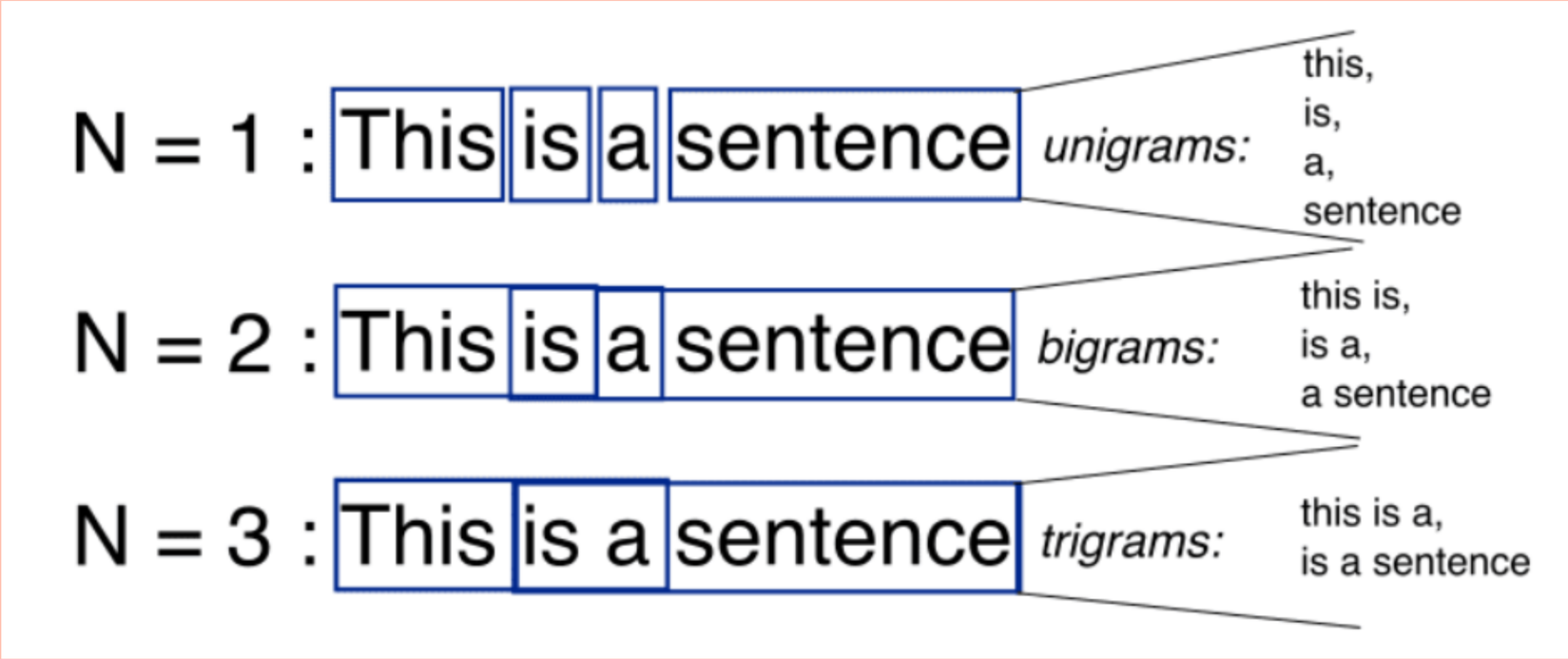
PREPROCESS

```
all_jobs_df['test']=all_jobs_df['Job_Description'].apply(string_process)
all_jobs_df['test']
```

```
0        about the company: smart food safe solutions ...
1        location : bangalore experience : 4+ years jo...
2        open systems international inc osi wwwosiicom...
3        about the job software testing engineer job d...
4        location: bangalore experience: 3 to 6years s...
                              ...
1919     skills and qualifications: 2+ years of experi...
1920     job id : th10519_13189 posted on: 29th of may...
1921     job description we spend 90 percent of our li...
1922     job number: 1905027 job title web developer u...
1923     we marry design and engineering language in w...
Name: test, Length: 1923, dtype: object
```

N-GRAM Language Model

- Language modeling is used to determine the probability of the word's sequence.
- An N-gram model is one type of a Language Model (LM), which is about finding the probability distribution over word sequences.



https://www.kdnuggets.com/2022/06/ngram-language-modeling-natural-language-processing.html

NGRAMS on Jobseeker Skills

```
# print(data['skills'])

data_lower = []
for s in data['skills']:
    data_lower.append(s.lower())

jobseeker_skills = " "
jobseeker_skills = jobseeker_skills.join(
j=1
for i in jobseeker_skills:
  if j%40:
    print(i, end='')
  else:
    print(i)
  j = j+1
```

```
affiliate visual content html python com
puter science mobile android website pre
sentation mysql php c++ java xml
```

```
<class 'dict'>
29
{'affiliate': 0,
 'visual': 24,
 'content': 6,
 'html': 8,
 'python': 20,
 'computer': 4,
 'science': 22,
 'mobile': 12,
 'android': 2,
 'website': 26,
 'presentation': 18,
 'mysql': 14,
 'php': 16,
 'java': 10,
 'xml': 28,
 'affiliate visual': 1,
 'visual content': 25,
 'content html': 7,
 'html python': 9,
 'python computer': 21,
 'computer science': 5,
 'science mobile': 23,
 'mobile android': 13,
 'android website': 3,
 'website presentation': 27,
 'presentation mysql': 19,
 'mysql php': 15,
 'php java': 17,
 'java xml': 11}
```

COUNT Vectorization

Count Vectorization involves counting the number of occurrences each words appears in a document

```
from sklearn.feature_extraction.text import CountVectorizer

vtest = CountVectorizer()
vtest = CountVectorizer(ngram_range=(1,2))
vtest.fit(["I am a robot who loves to play with people"])
print("Length of frequency matrix :",len(vtest.vocabulary_))
count = 1
print(" ")
for key, val in vtest.vocabulary_.items():
  if count%5:
    print(key,":", val, end=" ")
  else:
    print(key,":", val)
  count = count+1
tf_count = vtest.transform(["I am a robot"])
print(" ")
print("Count Vector for sentence 'I am a robot' :")
tf_count.toarray()
```

```
Length of frequency matrix : 15

am : 0 robot : 7 who : 11 loves : 2 to : 9
play : 5 with : 13 people : 4 am robot : 1 robot who : 8
who loves : 12 loves to : 3 to play : 10 play with : 6 with people : 14

Count Vector for sentence 'I am a robot' :
array([[1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]])
```

TF-MATRIX of Token Counts

```
tf_count = v.transform(all_jobs_df['test'])
tf_ud = tf_count.toarray()
print(type(tf_count))
print(tf_ud.shape)
print(tf_ud)
```

```
<class 'scipy.sparse.csr.csr_matrix'>
(1923, 233)
[[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]]
```

TF-IDF Vectorization

Term Frequency-Inverse document frequency refers to how common or rare a term appears in a document

```
vtest = TfidfVectorizer()
vtest = TfidfVectorizer(ngram_range=(1,2))
vtest.fit(["I am a robot who loves to play with people"])
print("Length of frequency matrix :",len(vtest.vocabulary_))
count = 1
print(" ")
for key, val in vtest.vocabulary_.items():
  if count%5:
    print(key,":", val, end=" ")
  else:
    print(key,":", val)
  count = count+1
tf_count = vtest.transform(["I am a robot"])
print(" ")
print("Count Vector for sentence 'I am a robot' :")
tf_count.toarray()
```

```
Length of frequency matrix : 15

am : 0 robot : 7 who : 11 loves : 2 to : 9
play : 5 with : 13 people : 4 am robot : 1 robot who : 8
who loves : 12 loves to : 3 to play : 10 play with : 6 with people : 14

Count Vector for sentence 'I am a robot' :
array([[0.57735027, 0.57735027, 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.57735027, 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ]])
```

TF-MATRIX of TF-IDF Features

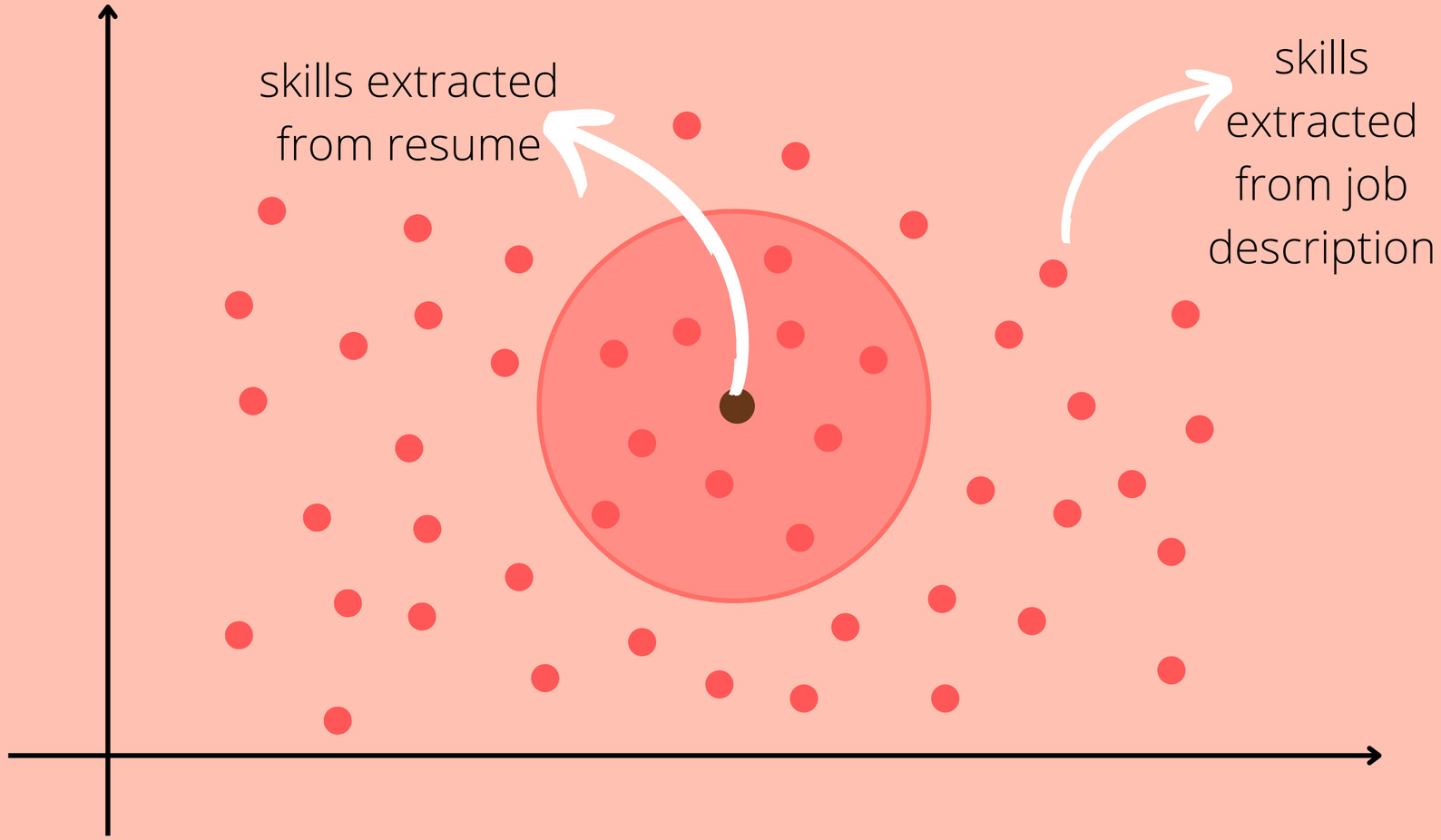
```
tfidf = vectorizer.transform(all_jobs_df['test'])
print(type(tfidf))
tfidf_arr = tfidf.toarray()
print(tfidf_arr.shape)

print(vectorizer.get_feature_names_out())
print(tfidf.indices)
print(tfidf_arr)
```

```
<class 'scipy.sparse.csr.csr_matrix'>
(1923, 29)
['affiliate' 'affiliate visual' 'android' 'android website' 'computer'
 'computer science' 'content' 'content html' 'html' 'html python' 'java'
 'java xml' 'mobile' 'mobile android' 'mysql' 'mysql php' 'php' 'php java'
 'presentation' 'presentation mysql' 'python' 'python computer' 'science'
 'science mobile' 'visual' 'visual content' 'website'
 'website presentation' 'xml']
[22  5  4 ... 10  6 10]
[[0.        0.        0.        ... 0.        0.        0.       ]
 [0.        0.        0.        ... 0.        0.        0.       ]
 [0.        0.        0.        ... 0.        0.        0.       ]
 ...
 [0.        0.        0.        ... 0.        0.        0.       ]
 [0.        0.        0.        ... 0.2236068 0.        0.2236068]
 [0.        0.        0.        ... 0.        0.        0.       ]]
```

Top-K Recommendation using KNN

KNN algorithm is used to classify by finding the K nearest matches in training data and then using the label of closest matches to predict.



MATCH SKILLS to JOBS

```
[153] nbrs = NearestNeighbors(n_neighbors=10, n_jobs=-1).fit(tfidf)
```

```
[157] def getNearestN(query):
        q_tf_count = vectorizer.transform(query)
        distances, indices = nbrs.kneighbors(q_tf_count)
        return distances, indices
```

```
[156] jobseeker_skills
```

```
'affiliate visual content html python computer science mobile androi
```

```
[158] distances, indices = getNearestN([jobseeker_skills])
```

```
print("Distances            ", "Indices")
for i in range(1,len(distances[0])):
  print(distances[0][i],"    ", indices[0][i])
```

```
Distances             Indices
0.9744478113897219      710
0.9744478113897219      1426
0.9744478113897219      1524
0.9960890925411551      1780
0.9960890925411551      1779
0.9999999999999998      686
0.9999999999999998      667
0.9999999999999998      655
0.9999999999999998      703
```

RECOMMEND

```
[161] type(indices)
      for i in indices:
          rec_jobs = all_jobs_df[['Position', 'Company','Location']].iloc[i]
      rec_jobs
```

	Position	Company	Location
1900	Full Stack Developer	Indian Angel network	Bengaluru
710	Staff Engineer, Data Engineering	Western Digital	â€“ Bengaluru
1427	Front-end Developer	Toolyt	Bengaluru
1525	Full stack UI developer	Netskope	Bengaluru
1781	UI Developer (Angular)	Bengaluru	Bengaluru
1780	Software Engineering - Senior Java Web Developer	J.P. Morgan	Bengaluru
686	Lead - Global Advanced Analytics Data Scientist-2	Diageo	â€“ Bengaluru
667	Data Analyst	SmartCoin	â€“ Bengaluru
655	Data Scientist (3-8 Years) for a Leading Softw...	Zyoin	â€“ Bengaluru
703	Big Data Engineer	Ganit	â€“ Bengaluru

NEXT STEPS

1. Input module for Recruiters.
2. Mapping jobs to resumes.
3. Checking and improving the accuracy of the model.
4. Building the user interface for the project
5. Trying different recommendation technologies and create a hybrid recommendation model

REFERENCES

- https://www.analyticssteps.com/blogs/what-content-based-recommendation-system-machine-learning
- https://www.kdnuggets.com/2022/06/ngram-language-modeling-natural-language-processing.html
- NLP Resume Parser Python | Satyajit Pattnaik, https://youtu.be/X83cDfwtFpw
- https://devopedia.org/n-gram-model
- https://towardsdatascience.com/text-classification-using-k-nearest-neighbors-46fa8a77acc5