2-Generating data required for further analysis

Importing Necessary Libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import _pickle as pickle
```

Importing Database

In [2]:
```
df = pd.read_csv('Complete_database.csv')
```

In [3]:
```
df[df.Country=='India']
```

Out[3]:

	Unnamed: 0	Authors	Title	Year	Cited by	Country	Funding_Details
23567	0	Jayadeva, Khemchandani, R., Chandra, S.	Twin support vector machines for pattern class...	2007	958.0	India	0
23568	1	Ravi, K., Ravi, V.	A survey on opinion mining and sentiment analy...	2015	617.0	India	0
23569	2	Varma, M., Zisserman, A.	A statistical approach to material classificat...	2009	479.0	India	University of Oxford\n \nEuropean Commission
23570	3	Nayak, P.C., Sudheer, K.P., Rangan, D.M., Rama...	A neuro-fuzzy computing technique for modeling...	2004	466.0	India	0
23571	4	Kale, A., Sundaresan, A., Rajagopalan, A.N., C...	Identification of humans using gait	2004	445.0	India	0
...	...	...	...	...	...	...	...
28945	5378	Narasimhan, R.	Artificial intelligence in fifth generation co...	1986	0.0	India	0
28946	5379	Ramani, S., Chandrasekar, R.	Partitioning computations and parallel processing	1986	0.0	India	0
28947	5380	Krishna, M.H., Murty, N.M.	A conceptual clustering scheme for frame-based...	1986	0.0	India	0
28948	5381	Krishnamurthy, E.V., Subramanian, K., Mahadeva...	Formal description, compression and transforma...	1974	0.0	India	0
28949	5382	Aggarwal, G.K.	On negative character of information	1968	0.0	India	0

5383 rows × 7 columns

In [4]:
```
df.head()
```

Out[4]:

	Unnamed: 0	Authors	Title	Year	Cited by	Country	Funding_Details
0	0	Soares, J.V.B., Leandro, J.J.G., Cesar Jr., R....	Retinal vessel segmentation using the 2-D Gabo...	2006	1083.0	Australia	0
1	1	Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuch...	The graph neural network model	2009	1031.0	Australia	0
2	2	Karantonis, D.M., Narayanan, M.R., Mathie, M.,...	Implementation of a real-time human movement c...	2006	908.0	Australia	0
3	3	Mirjalili, S.	Dragonfly algorithm: a new meta-heuristic opti...	2016	865.0	Australia	0
4	4	Naseem, I., Togneri, R., Bennamoun, M.	Linear regression for face recognition	2010	768.0	Australia	0

In [5]:
```
df.drop('Unnamed: 0',axis='columns',inplace=True)
```

In [6]:
```
df = df.rename(columns={'Cited by':'Cited_by'})
```

In [7]:
```
df.head()
```

Out[7]:

	Authors	Title	Year	Cited_by	Country	Funding_Details
0	Soares, J.V.B., Leandro, J.J.G., Cesar Jr., R....	Retinal vessel segmentation using the 2-D Gabo...	2006	1083.0	Australia	0
1	Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuch...	The graph neural network model	2009	1031.0	Australia	0
2	Karantonis, D.M., Narayanan, M.R., Mathie, M.,...	Implementation of a real-time human movement c...	2006	908.0	Australia	0
3	Mirjalili, S.	Dragonfly algorithm: a new meta-heuristic opti...	2016	865.0	Australia	0
4	Naseem, I., Togneri, R., Bennamoun, M.	Linear regression for face recognition	2010	768.0	Australia	0

In [8]:
```
df_without_countries = df.drop('Country',axis='columns')
```

In [9]:
```
df_without_countries = df_without_countries.drop_duplicates(subset=['Title'],keep='first')
```

In [10]:
```
df_without_countries.shape
```

Out[10]:
```
(59215, 5)
```

Generating Author:database dictonary

In [11]:
```
authors_lst = list(df_without_countries.loc[:,'Authors'].values)
```

In [12]:
```
print(authors_lst[0])
```

```
Soares, J.V.B., Leandro, J.J.G., Cesar Jr., R.M., Jelinek, H.F., Cree, M.J.
```

In [13]:
```
len(authors_lst)
```

Out[13]:
```
59215
```

In [14]:
```
set_authors = ['Jayadeva', 'Khemchandani, R.', 'Chandra, S.']
for i in range(len(authors_lst)):
    authors_sub_lst = authors_lst[i].split(',')
    authors_sub_lst_mod = []
    if authors_sub_lst == 'Jayadeva, Khemchandani, R., Chandra, S.'.split(','):
        continue
    for j in range(0,len(authors_sub_lst)-1,2):
        authors_sub_lst_mod.append(authors_sub_lst[j].strip()+','+authors_sub_l
st[j+1])

    for author in authors_sub_lst_mod:
        if(author not in set_authors):
            set_authors.append(author)
            # print(f'{i}\t{author}')
```

Storing Authors Names to a file so that it can be easily availabe afterwards

In [15]:
```
with open('Authors_list.txt','w') as filehandle:
        filehandle.writelines("%s\n" % author for author in set_authors)
filehandle.close()
```

Reading Authors from Above file and storing into a python list

In [16]:
```
set_authors = []
with open('Authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_authors.append(author)
```

In [17]:
```
len(set_authors)
```

Out[17]:
```
120161
```

Creating Author:Database Dictonary

In [18]:
```
dct_author_database = {}
count=0
for author in set_authors:
    # print(f'{count}\t{author}')

    df_auth = pd.DataFrame(columns = ['Authors', 'Title', 'Year', 'Cited_by','F
unding Details'] )

    filt= df_without_countries['Authors'].str.contains(author, na=False)
    df_auth= df_without_countries.loc[filt,'Authors':].reset_index(drop=True)

    dct_author_database[author] = df_auth

    count += 1
```

```
/home/deshabhakt/.local/lib/python3.8/site-packages/pandas/core/strings/accesso
r.py:101: UserWarning: This pattern has match groups. To actually get the group
s, use str.extract.
  return func(self, *args, **kwargs)
```

Alterative to above code

''' dct_author_database = {}

count = 0 for author in set_authors:

```
    # print(f'{count}\t{author}')

    df_auth = pd.DataFrame(columns = ['Authors', 'Title', 'Year', 'Cited_by'] )

    for authors in authors_lst:

        if author in authors:
            df_tmp = df_without_countries[df_without_countries.Authors==authors]
            df_auth = df_auth.append(df_tmp,ignore_index=True)

    dct_author_database[author] = df_auth

    count += 1
```

'''

Storing Author:Database dictonary to file so that it can reused again easily

In [19]:
```
with open('Author_database_dictonary.txt','wb') as file:
    file.write(pickle.dumps(dct_author_database))
file.close()
```

Reading Author:Database dictonary from Author_database_dictonary.txt file

In [ ]:
```
dct_author_database = {}
with open('Author_database_dictonary.txt','rb') as file:
    dct_author_database = pickle.load(file)
file.close()
```

In [20]:
```
print(type(dct_author_database))
```

```
<class 'dict'>
```

In [21]:
```
print(set_authors[10])
dct_author_database[set_authors[10]]
```

```
Tsoi, A.C.
```

Out[21]:

	Authors	Title	Year	Cited_by	Funding_Details
0	Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuch...	The graph neural network model	2009	1031.0	0
1	Shilton, A., Palaniswami, M., Ralph, D., Tsoi,...	Incremental training of support vector machines	2005	159.0	0
2	Scarselli, F., Tsoi, A.C., Hagenbuchner, M., N...	Solving graph data issues using a layered arch...	2013	10.0	0
3	Scarselli, F., Tsoi, A.C., Hagenbuchner, M.	The Vapnik–Chervonenkis dimension of graph and...	2018	4.0	0
4	Pucci, A., Gori, M., Hagenbuchner, M., Scarsel...	Investigations into the application of Graph N...	2006	3.0	0

Creating indian authors list and foreign authors list and storing those in file

Creating Authors_collective list from Indian database

In [22]:
```
authors_from_ind_database = list(df[df.Country=='India']['Authors'].unique())
```

separating individual authors from authors_collective list

In [25]:
```
set_of_authors_from_indian_database = ['Jayadeva', 'Khemchandani, R.', 'Chandr
a, S.']

count = 0
for authors in authors_from_ind_database:

    authors_sub_lst = authors.split(',')

    if authors_sub_lst == 'Jayadeva, Khemchandani, R., Chandra, S.'.split(','):
        continue

    authors_sub_lst_mod = []
    for i in range(0,len(authors_sub_lst)-1,2):
            authors_sub_lst_mod.append(authors_sub_lst[i].strip()+','+authors_s
ub_lst[i+1])

    for author in authors_sub_lst_mod:
        if(author not in set_of_authors_from_indian_database):
            set_of_authors_from_indian_database.append(author)
    # print(f'{count}\t{authors}')
    count += 1
```

Storing Indian Authors names to file

In [26]:
```
with open('Indian_authors_list.txt','w') as filehandle:
        filehandle.writelines("%s\n" % author for author in set_of_authors_from
_indian_database)
filehandle.close()
```

After this step we manually removed non-indian authors from the Indian authors list

Reading Indian Authors names to file

In [27]:
```
set_of_indian_authors_from_file = []
with open('Indian_authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_of_indian_authors_from_file.append(author)
```

In [28]:
```
len(set_of_indian_authors_from_file)
```

Out[28]:
```
9267
```

Creating Foreign authors list

In [30]:
```
set_of_foreign_authors = []

count = 0
for author in set_authors:
    if author not in set_of_indian_authors_from_file:
        set_of_foreign_authors.append(author)
        # print(f'{count}\t{author}')
        count += 1
```

Storing Foreign authors in txt file so that it can be reused again easily

In [32]:
```
with open('Foreign_authors_list.txt','w') as filehandle:
        filehandle.writelines("%s\n" % author for author in set_of_foreign_auth
ors)
filehandle.close()
```

Reading Foreign authors from foreign authors list file

In [33]:
```
set_of_foreign_authors_from_file = []
with open('Foreign_authors_list.txt','r') as filehandle:
    filecontents = filehandle.readlines()

    for line in filecontents:
        # remove linebreak which is the last character of the string
        author = line[:-1]

        # add item to the list
        set_of_foreign_authors_from_file.append(author)
```

In [34]:
```
print(len(set_of_foreign_authors_from_file))
```

```
110892
```

Creating a dictonary with foreign author as key and number of paper published by him with india authors as value

In [35]:
```
dct_foreign_author_coauth_count = {}
for foreign_author in set_of_foreign_authors_from_file:
    dct_foreign_author_coauth_count[foreign_author] = 0
```

In [37]:
```
count = 0
for foreign_author in set_of_foreign_authors_from_file:
    row_foreign_auth_list = dct_author_database[foreign_author].iloc[:,0].value
s
    for indian_author in set_of_indian_authors_from_file:
        for authors in row_foreign_auth_list:
            if indian_author in authors:
                dct_foreign_author_coauth_count[foreign_author] += 1
    # print(f'{count}     {foreign_author}        {dct_foreign_author_coauth_co
unt[foreign_author]}')
    count += 1
```

Storing above dictonary in file so that it can be easily reused again

In [38]:
```
with open('Foreign_auth_and_their_publication_count_with_india_authors_dct.txt
','wb') as file:
    file.write(pickle.dumps(dct_foreign_author_coauth_count))
file.close()
```

Reading above stored dictonary from file and storing it in a python dictonary variable

In [ ]:
```
dct_foreign_author_coauth_count_from_file = {}
with open('Foreign_auth_and_their_publication_count_with_india_authors_dct.txt
','rb') as file:
    dct_foreign_author_coauth_count_from_file = pickle.load(file)
file.close()
```