In [1]:

```
import csv

import json

import pandas as pd

import numpy as np

import seaborn as sns
sns.set()

import re

from my_nlp import *
```

In [65]:

```
import random
```

functions

clean names of NGOs

In [2]:

```
def name_cleaner(s, StopWords):
    s = depunctuate(s).lower()
    Tokens = tokenize(s)
    UnStopped = [t for t in Tokens if t not in StopWords]
    sL = list(set(UnStopped))
    sL.sort()
    CleanedName = ' '.join(sL)
    return CleanedName
```

In [3]:

```
s = 'I am a rabbit! @ home with the bunnies? how are of the
 they/them cow and jump Over Moon. rabbit Rabbit they them a
re'
cleans = name_cleaner(s, SomeStopWordsV)
print(cleans)
```

```
am are bunnies cow home i jump moon over rabbit
them they with
```

WIP pincode extract

this is only the HQ pincode, not where they are active

In [4]:

```
def extract_regex(regx, st):
    """ regx = raw string so no escape backslash"""
    pin_re = re.compile(regx)
    m = pin_re.search(st)
    if m:
        pinS = m.group()
    else:
        pinS = ""
    return pinS
```

In [5]:

```
#pincodes starting with '0' don't exist, those starting with
'9' are APOs (Army Post Offices)
rgx = r"[1-8]\d{5}" # raw string so no escape backslash
```

In [6]:

```
# no match
strng = r"https:// hey 103 feminisminindia.com/"
print(extract_regex(rgx, strng))
```

In [7]:

```
# match
strng = r"some or the other address, bangloure, india, 56723
8, karnataka"
print(extract_regex(rgx, strng))
```

```
567238
```

In [8]:

```
# no '0' start pincodes
strng = r"#2 dhoop chhaon tree, bangloure, india, 067238, ka
rnataka"
print(extract_regex(rgx, strng))
```

In [9]:

```
# no '9' start pincodes which are Army POs
strng = r"some of the other address, bangloure, india, 96723
8, karnataka"
print(extract_regex(rgx, strng))
```

In [10]:

```
# no less than 6 digits
strng = r"some of the other address, 534f67 bangloure, indi
a, 67238, karnataka"
print(extract_regex(rgx, strng))
```

In [11]:

```
# multiple pincodes extracts first valid only
strng = r"#2 dhoop chhaon tree,  bangloure 441007, india, 66
7238, karnataka"
print(extract_regex(rgx, strng))
```

```
441007
```

Darpan21

In [12]:

```
Darpan21DF = pd.read_csv('42621 Final_Data_ngodarpan.csv', d
type={"Mobile": 'string', "UniqueID": "string"\
                        , "fcrano": "string", "president mob
ile": "string", "Chairman mobile": "string", "Secretary mobi
le": "string"\
                        , "Asisstant Secretary mobile": "str
ing", "Board Member mobile": "string", "Vice Chairman mobil
e": "string", "Member mobile": "string"\
                        , "issues working db": "string"}) #,
low_memory = False)
```

In [13]:

```
Darpan21DF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111929 entries, 0 to 111928
Data columns (total 42 columns):
 #   Column                    Non-Null Count
Dtype
---  ------                    --------------
-----
 0   Name                      111929 non-null
object
 1   ngo url                   25787 non-null
object
 2   Mobile                    111897 non-null
string
 3   UniqueID                  111929 non-null
string
 4   Off phone1                16527 non-null
object
 5   Email                     111929 non-null
object
 6   Major Activities1         84618 non-null
object
 7   operational states db     88890 non-null
object
 8   issues working db         89292 non-null
string
 9   operational district db   88890 non-null
object
 10  reg name                  111929 non-null
object
 11  fcrano                    22060 non-null
string
 12  nr regNo                  111926 non-null
object
 13  nr add                    111929 non-null
object
 14  nr orgName                111929 non-null
object
 15  ngo reg date              111929 non-null
object
 16  nr actName                110613 non-null
object
 17  nr city                   111715 non-null
```

```
object
 18  TypeDescription              111929 non-null
object
 19  StateName                    111929 non-null
object
 20  status                       0 non-null
float64
 21  president name               59409 non-null
object
 22  president email              59409 non-null
object
 23  president mobile             59409 non-null
string
 24  Chairman name                29803 non-null
object
 25  Chairman email               29797 non-null
object
 26  Chairman mobile              29792 non-null
string
 27  Secretary name               74508 non-null
object
 28  Secretary email              74486 non-null
object
 29  Secretary mobile             74482 non-null
string
 30  Asisstant Secretary name     1037 non-null
object
 31  Asisstant Secretary email    1037 non-null
object
 32  Asisstant Secretary mobile  1037 non-null
string
 33  Board Member name            5001 non-null
object
 34  Board Member email           5001 non-null
object
 35  Board Member mobile          5001 non-null
string
 36  Vice Chairman name           5451 non-null
object
 37  Vice Chairman email          5448 non-null
object
 38  Vice Chairman mobile         5449 non-null
```

```
string
 39  Member name                 29572 non-null
object
 40  Member email                29558 non-null
object
 41  Member mobile               29560 non-null
string
dtypes: float64(1), object(30), string(11)
memory usage: 35.9+ MB
```

reg name has city name sometimes, do not use

StateName clean and use with Name for matching

In [14]:

```
Darpan21DF.drop(['reg name', 'status'], axis='columns', inpl
ace=True)
```

In [15]:

```
Darpan21DF.describe()
```

Out[15]:

	Name	ngo url	Mobile	UniqueID	Off phone1
count	111929	25787	111897	111929	16527
unique	109682	24252	111430	111929	15363
top	CATHOLIC CHURCH	http://	9422471767	PB/2017/0159714	00000-000000
freq	25	859	8	1	73

4 rows × 40 columns

histograms

too long

sns.countplot(Darpan21DF['Name'], color='gray') Darpan21DF.groupby('Name').size().plot(kind='bar') excel pivot table shows freq = 31 "CATHOLIC CHURCH", and a unique "catholic church" in addition to many other uniques with the name of the parish

In [16]:

```
Darpan21DF['CleanName'] = Darpan21DF.apply(lambda row: name_
cleaner(row.Name, SomeStopWordsV), axis = 1)
Darpan21DF['CleanState'] = Darpan21DF.apply(lambda row: name
_cleaner(row.StateName, SomeStopWordsV), axis = 1)
Darpan21DF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111929 entries, 0 to 111928
Data columns (total 42 columns):
 #   Column                       Non-Null Count
Dtype
---  ------                       --------------
-----
 0   Name                         111929 non-null
object
 1   ngo url                      25787 non-null
object
 2   Mobile                       111897 non-null
string
 3   UniqueID                     111929 non-null
string
 4   Off phone1                   16527 non-null
object
 5   Email                        111929 non-null
object
 6   Major Activities1            84618 non-null
object
 7   operational states db        88890 non-null
object
 8   issues working db            89292 non-null
string
 9   operational district db      88890 non-null
object
 10  fcrano                       22060 non-null
string
 11  nr regNo                     111926 non-null
object
 12  nr add                       111929 non-null
object
 13  nr orgName                   111929 non-null
object
 14  ngo reg date                 111929 non-null
object
 15  nr actName                   110613 non-null
object
 16  nr city                      111715 non-null
object
 17  TypeDescription              111929 non-null
```

```
object
 18  StateName                      111929 non-null
object
 19  president name                 59409 non-null
object
 20  president email                59409 non-null
object
 21  president mobile               59409 non-null
string
 22  Chairman name                  29803 non-null
object
 23  Chairman email                 29797 non-null
object
 24  Chairman mobile                29792 non-null
string
 25  Secretary name                 74508 non-null
object
 26  Secretary email                74486 non-null
object
 27  Secretary mobile               74482 non-null
string
 28  Asisstant Secretary name       1037 non-null
object
 29  Asisstant Secretary email      1037 non-null
object
 30  Asisstant Secretary mobile     1037 non-null
string
 31  Board Member name              5001 non-null
object
 32  Board Member email             5001 non-null
object
 33  Board Member mobile            5001 non-null
string
 34  Vice Chairman name             5451 non-null
object
 35  Vice Chairman email            5448 non-null
object
 36  Vice Chairman mobile           5449 non-null
string
 37  Member name                    29572 non-null
object
 38  Member email                   29558 non-null
```

```
object
 39  Member mobile                  29560 non-null
string
 40  CleanName                      111929 non-null
object
 41  CleanState                     111929 non-null
object
dtypes: object(31), string(11)
memory usage: 35.9+ MB
```

FCRA

In [17]:

```
FcraDF = pd.read_csv('FCRA - Sheet1.csv', dtype = {'S.No.':
'string', 'Registration': 'string'}) #, low_memory = False)
```

In [18]:

```
FcraDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19903 entries, 0 to 19902
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Location         19903 non-null  object
 1   S.No.            19894 non-null  string
 2   Registration     19892 non-null  string
 3   AssociationName  19892 non-null  object
 4   Address          19794 non-null  object
 5   Nature           19890 non-null  object
dtypes: object(4), string(2)
memory usage: 933.1+ KB
```

Note: "Nature" contains "Religious(Hindu) ,Cultural ,Educational ," etc info if we want to use it

In [19]:

```
FcraDF.dropna(subset = ['Registration', 'AssociationName',
'Nature'], inplace = True)
```

In [20]:

```
FcraDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19890 entries, 0 to 19891
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Location         19890 non-null  object
 1   S.No.            19890 non-null  string
 2   Registration     19890 non-null  string
 3   AssociationName  19890 non-null  object
 4   Address          19792 non-null  object
 5   Nature           19890 non-null  object
dtypes: object(4), string(2)
memory usage: 1.1+ MB
```

In [21]:

```
FcraDF.describe()
```

Out[21]:

	Location	S.No.	Registration	AssociationName	Addres
count	19890	19890	19890	19890	1979
unique	29	3264	19411	18923	1915
top	Tamil Nadu	1	125410003	Society of the Mission of Sisters of Ajmer	, Dist Ajmei Rajasthan
freq	3264	496	469	469	46

FCRA DATA, address is too non-standard to parse easily, so just use "location" -which is state name- to match

In [22]:

```
FcraDF['CleanName'] = FcraDF.apply(lambda row: name_cleaner(
row.AssociationName, SomeStopWordsV), axis = 1)
FcraDF['CleanState'] = FcraDF.apply(lambda row: name_cleaner
(row.Location, SomeStopWordsV), axis = 1)
FcraDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19890 entries, 0 to 19891
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Location         19890 non-null  object
 1   S.No.            19890 non-null  string
 2   Registration     19890 non-null  string
 3   AssociationName  19890 non-null  object
 4   Address          19792 non-null  object
 5   Nature           19890 non-null  object
 6   CleanName        19890 non-null  object
 7   CleanState       19890 non-null  object
dtypes: object(6), string(2)
memory usage: 1.4+ MB
```

Merge the two datasets

In [23]:

```
DarpanFcraDF = Darpan21DF.merge(FcraDF, on = ['CleanName',
'CleanState'], how = 'left')
```

In [24]:

```
DarpanFcraDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 112863 entries, 0 to 112862
Data columns (total 48 columns):
 #   Column                        Non-Null Count
Dtype
---  ------                        --------------
-----
 0   Name                          112863 non-null
object
 1   ngo url                       25916 non-null
object
 2   Mobile                        112831 non-null
string
 3   UniqueID                      112863 non-null
string
 4   Off phone1                    16623 non-null
object
 5   Email                         112863 non-null
object
 6   Major Activities1             85372 non-null
object
 7   operational states db         89740 non-null
object
 8   issues working db             90138 non-null
string
 9   operational district db       89740 non-null
object
 10  fcrano                        22801 non-null
string
 11  nr regNo                      112860 non-null
object
 12  nr add                        112863 non-null
object
 13  nr orgName                    112863 non-null
object
 14  ngo reg date                  112863 non-null
object
 15  nr actName                    111539 non-null
object
 16  nr city                       112649 non-null
object
 17  TypeDescription               112863 non-null
```

```
object
 18  StateName                     112863 non-null
object
 19  president name                59866 non-null
object
 20  president email               59866 non-null
object
 21  president mobile              59866 non-null
string
 22  Chairman name                 29913 non-null
object
 23  Chairman email                29907 non-null
object
 24  Chairman mobile               29902 non-null
string
 25  Secretary name                74957 non-null
object
 26  Secretary email               74935 non-null
object
 27  Secretary mobile              74931 non-null
string
 28  Asisstant Secretary name    1050 non-null
object
 29  Asisstant Secretary email   1050 non-null
object
 30  Asisstant Secretary mobile  1050 non-null
string
 31  Board Member name             5020 non-null
object
 32  Board Member email            5020 non-null
object
 33  Board Member mobile           5020 non-null
string
 34  Vice Chairman name            5463 non-null
object
 35  Vice Chairman email           5460 non-null
object
 36  Vice Chairman mobile          5461 non-null
string
 37  Member name                   30023 non-null
object
 38  Member email                  30009 non-null
```

```
object
 39  Member mobile                  30011 non-null
string
 40  CleanName                      112863 non-null
object
 41  CleanState                     112863 non-null
object
 42  Location                       11048 non-null
object
 43  S.No.                          11048 non-null
string
 44  Registration                   11048 non-null
string
 45  AssociationName                11048 non-null
object
 46  Address                        10988 non-null
object
 47  Nature                         11048 non-null
object
dtypes: object(35), string(13)
memory usage: 42.2+ MB
```

In [25]:

```
DarpanFcraDF[['CleanName', 'CleanState', 'fcrano', 'Registra
tion']].head()
```

Out[25]:

	CleanName	CleanState	fcrano	Registrati
0	prayas	orissa	105100015	1051000
1	pondicherrywomensconference	puducherry	<NA>	<N
2	samaj samiti sewa shabri	madhya pradesh	<NA>	<N
3	anand ganga samajik samiti siksha	pradesh uttar	<NA>	<N
4	gram himaliyan samiti vikas	uttarakhand	347990011	<N

In [26]:

```
DarpanFcraDF[['CleanName', 'CleanState', 'fcrano', 'Registra
tion']].tail()
```

Out[26]:

	CleanName	CleanState	fcrano	Registration
112858	hariom samaj samiti vikas	pradesh uttar	<NA>	<NA>
112859	hoshangabad jan jeevan kalyaan narmadanchal na...	madhya pradesh	<NA>	<NA>
112860	gramodyog mathura prasad sansthan	pradesh uttar	<NA>	<NA>
112861	education shree swaminarayan trust	gujarat	<NA>	<NA>
112862	sansthan srijan	rajasthan	<NA>	<NA>

In [27]:

```
DarpanFcraDF[['CleanName', 'CleanState', 'fcrano', 'Registra
tion']][DarpanFcraDF.fcrano.isnull() & DarpanFcraDF.Registra
tion.notnull()]
```

Out[27]:

	CleanName	CleanState	fcrano	Registration
40	foundation sarthak	pradesh uttar	<NA>	136550502
56	lakshya	bihar	<NA>	31170349
220	education environmental foundation natural res...	pradesh uttar	<NA>	136580052
364	dakshin durgapur kshudiram sangha smriti	bengal west	<NA>	147110375
381	gramin kendra vikas	jharkhand	<NA>	337670001
...	...	...	...	...
112251	birds charitable nest trust	nadu tamil	<NA>	75901470
112277	contemporary culture for foundation g5a	maharashtra	<NA>	83781636
112334	development educational mata rural sri trust	karnataka	<NA>	94360017
112455	charity down town trust	assam	<NA>	20780101
112568	kalyan mushar sangh seva zati	bihar	<NA>	31250029

1252 rows × 4 columns

In [28]:

```
DarpanFcraDF['FCRA'] = DarpanFcraDF['Registration'].fillna(D
arpanFcraDF['fcrano'])
DarpanFcraDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 112863 entries, 0 to 112862
Data columns (total 49 columns):
 #   Column                       Non-Null Count
Dtype
---  ------                       --------------
-----
 0   Name                         112863 non-null
object
 1   ngo url                      25916 non-null
object
 2   Mobile                       112831 non-null
string
 3   UniqueID                     112863 non-null
string
 4   Off phone1                   16623 non-null
object
 5   Email                        112863 non-null
object
 6   Major Activities1            85372 non-null
object
 7   operational states db        89740 non-null
object
 8   issues working db            90138 non-null
string
 9   operational district db      89740 non-null
object
 10  fcrano                       22801 non-null
string
 11  nr regNo                     112860 non-null
object
 12  nr add                       112863 non-null
object
 13  nr orgName                   112863 non-null
object
 14  ngo reg date                 112863 non-null
object
 15  nr actName                   111539 non-null
object
 16  nr city                      112649 non-null
object
 17  TypeDescription              112863 non-null
```

```
object
 18  StateName                    112863 non-null
object
 19  president name               59866 non-null
object
 20  president email              59866 non-null
object
 21  president mobile             59866 non-null
string
 22  Chairman name                29913 non-null
object
 23  Chairman email               29907 non-null
object
 24  Chairman mobile              29902 non-null
string
 25  Secretary name               74957 non-null
object
 26  Secretary email              74935 non-null
object
 27  Secretary mobile             74931 non-null
string
 28  Asisstant Secretary name    1050 non-null
object
 29  Asisstant Secretary email   1050 non-null
object
 30  Asisstant Secretary mobile  1050 non-null
string
 31  Board Member name           5020 non-null
object
 32  Board Member email          5020 non-null
object
 33  Board Member mobile         5020 non-null
string
 34  Vice Chairman name          5463 non-null
object
 35  Vice Chairman email         5460 non-null
object
 36  Vice Chairman mobile        5461 non-null
string
 37  Member name                  30023 non-null
object
 38  Member email                 30009 non-null
```

```
object
 39   Member mobile                  30011 non-null
string
 40   CleanName                      112863 non-null
object
 41   CleanState                     112863 non-null
object
 42   Location                       11048 non-null
object
 43   S.No.                          11048 non-null
string
 44   Registration                   11048 non-null
string
 45   AssociationName                11048 non-null
object
 46   Address                        10988 non-null
object
 47   Nature                         11048 non-null
object
 48   FCRA                           24053 non-null
string
dtypes: object(35), string(14)
memory usage: 43.1+ MB
```

Write csv

In [29]:

```
DarpanFcraDF.to_csv('Darpan21FCRA.csv', index=False)
# DarpanFcraDF.to_excel('Darpan21FCRA.xlsx', index=False)
```

not needed

DarpanFcraDF.reset_index(inplace=True) DarpanFcraDF = DarpanFcraDF.rename(columns = {'index':'new column name'})

Make sets of IDs for all tags

In [30]:

```
TagsToIDVL = []
TagsToIDD = {}
```

All NGO IDs

In [31]:

```
AllIDV = set(DarpanFcraDF.UniqueID)
```

In [32]:

```
TagsToIDVL.append({'tag': 'All', 'IDSet': AllIDV})
TagsToIDD['All'] = list(AllIDV)
len(AllIDV)
```

Out[32]:

```
111929
```

FCRA number exists?

In [33]:

```
FCRAV = set(DarpanFcraDF.UniqueID[DarpanFcraDF.FCRA.notnull
()])
```

In [34]:

```
TagsToIDVL.append({'tag': 'FCRA', 'IDSet': FCRAV})
TagsToIDD['FCRA'] = list(FCRAV)
len(FCRAV)
```

Out[34]:

```
23119
```

URL exists?

In [35]:

```
URLV = set(DarpanFcraDF.UniqueID[DarpanFcraDF['ngo url'].not
null()])
```

In [36]:

```
TagsToIDVL.append({'tag': 'URL', 'IDSet': URLV})
TagsToIDD['URL'] = list(URLV)
len(URLV)
```

Out[36]:

```
25787
```

Major activities exists?

In [37]:

```
MA1V = set(DarpanFcraDF.UniqueID[DarpanFcraDF['Major Activit
ies1'].notnull()])
```

In [38]:

```
TagsToIDVL.append({'tag': 'MajorActivities', 'IDSet': MA1V})
TagsToIDD['MA1'] = list(MA1V)
len(MA1V)
```

Out[38]:

```
84618
```

Write csv for tags and IDs

https://www.geeksforgeeks.org/how-to-save-a-python-dictionary-to-a-csv-file/ (https://www.geeksforgeeks.org/how-to-save-a-python-dictionary-to-a-csv-file/)

In [39]:

```
field_names = ['tag', 'IDSet']
with open('TagsToIDV.csv', 'w') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames = field_name
s)
    writer.writeheader()
    writer.writerows(TagsToIDVL)
```

write json for tags and IDs

https://www.geeksforgeeks.org/how-to-convert-python-dictionary-to-json/ (https://www.geeksforgeeks.org/how-to-convert-python-dictionary-to-json/)

In [40]:

```
# TagsToIDD[tag] = [UniqueID]
with open("TagsToIDList.json", "w") as outfile:
    json.dump(TagsToIDD, outfile)
```

Make reverse look up dictionaries for Issues, States and Districts

Issues: "Agriculture,Environment & Forests,Health & Family Welfare," States: "UTTAR PRADESH, testingswss, UTTAR PRADESH, testingswss, UTTAR PRADESH," Need to strip, remove '' and 'testingswss' and dedupe.

In [41]:

```
IssueToIDD = {}
StateToIDD = {}
IDToStateDistD = {}
for index, row in DarpanFcraDF.iterrows():
    UniqueID = row['UniqueID']
    Issues = row['issues working db']
    States = row['operational states db']
    Dists =row['operational district db']

    # issues dict
    try:
        IssuesL = list(set(Issues.split(',')))
        IssuesL.remove('')
        for issue in IssuesL:
            if issue in IssueToIDD:
                IssueToIDD[issue].append(UniqueID)
            else:
                IssueToIDD[issue] = [UniqueID]
    except (AttributeError, ValueError):
        pass

    # states dict
    try:
        StatesL = list(set(map(lambda s: s.strip(), States.s
plit(','))))
        StatesL.remove('')
        StatesL.remove('testingswss')
        for state in StatesL:
            if state in StateToIDD:
                StateToIDD[state].append(UniqueID)
            else:
                StateToIDD[state] = [UniqueID]
    except (AttributeError, ValueError):
        pass

    # districts list
    try:
        Dists1 = Dists.replace('->', ',')
        DistL = list(map(lambda s: s.strip(), Dists1.split(
',')))
```

```
    except (AttributeError, ValueError):
        pass

    DistL = [elem for elem in DistL if elem != '']
    DistL = [elem for elem in DistL if elem != 'testingswss'
]

    # ID to states/districts
    IDToStateDistD[UniqueID] = {}
    for state in StatesL:
        IDToStateDistD[UniqueID][state] = []

    for location in DistL:
        if location in StatesL:
            state = location
        else:
            IDToStateDistD[UniqueID][state].append(location)

    for state in StatesL:
        IDToStateDistD[UniqueID][state] = list(set(IDToState
DistD[UniqueID][state]))
```

In [42]:

```
StateDistToIDD = {}
for ID in IDToStateDistD:
    for state in IDToStateDistD[ID]:
        if state in StateDistToIDD:
            pass
        else:
            StateDistToIDD[state] = {}
        for dist in IDToStateDistD[ID][state]:
            if dist in StateDistToIDD[state]:
                StateDistToIDD[state][dist].append(ID)
            else:
                StateDistToIDD[state][dist] = [ID]
```

In [43]:

```
StatesL = list(StateDistToIDD.keys())
StatesSer = pd.Series(StatesL)

IssuesSer = pd.Series(list(IssueToIDD.keys()))
```

write json for sets of NGOs in Issue, State and State, Dist

https://www.geeksforgeeks.org/how-to-convert-python-dictionary-to-json/ (https://www.geeksforgeeks.org/how-to-convert-python-dictionary-to-json/)

In [44]:

```
# IssueToIDD[issue] = [UniqueID]
with open("IssueToIDList.json", "w") as outfile:
    json.dump(IssueToIDD, outfile)

# StateToIDD[state] = [UniqueID]
with open("StateToIDList.json", "w") as outfile:
    json.dump(StateToIDD, outfile)

# StateDistToIDD[state][dist] = [UniqueID]
with open("StateDistToIDList.json", "w") as outfile:
    json.dump(StateDistToIDD, outfile)
```

Write csv for sets of NGOs by feature

Write csv for sets of NGOs by issue

In [45]:

```
IssueToIDVL = []
for issue in IssueToIDD:
    IssueToIDVL.append({'issue': issue, 'IDSet': IssueToIDD[
issue]})

field_names = ['issue', 'IDSet']
with open('IssueToIDV.csv', 'w') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames = field_name
s)
    writer.writeheader()
    writer.writerows(IssueToIDVL)
```

Write csv for sets of NGOs by State

In [46]:

```
StateToIDVL = []
for state in StateToIDD:
    StateToIDVL.append({'state': state, 'IDSet': StateToIDD[
state]})

field_names = ['state', 'IDSet']
with open('StateToIDV.csv', 'w') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames = field_name
s)
    writer.writeheader()
    writer.writerows(StateToIDVL)
```

Write csv for sets of NGOs by State and Dist

In [47]:

```
StateDistToIDVL = []
for state in StateDistToIDD:
    for dist in StateDistToIDD[state]:
        StateDistToIDVL.append({'state': state, 'dist': dist
, 'IDSet': StateDistToIDD[state][dist]})

field_names = ['state', 'dist', 'IDSet']
with open('StateDistToIDV.csv', 'w') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames = field_name
s)
    writer.writeheader()
    writer.writerows(StateDistToIDVL)
```

Filter NGOs

Select Issues

In [48]:

```
IssuesSer
```

Out[48]:

```
0                      Environment & Forests
1                                   HIV/AIDS
2                        Labour & Employment
3                                   Children
4                             Drinking Water
5                   Health & Family Welfare
6          Right to Information & Advocacy
7                             Panchayati Raj
8      Rural Development & Poverty Alleviation
9        Women's Development & Empowerment
10                             Youth Affairs
11                       Vocational Training
12                              Human Rights
13     Information & Communication Technology
14                              Civic Issues
15                               Agriculture
16                    New & Renewable Energy
17                      Education & Literacy
18                                    Sports
19                       Disaster Management
20                            Tribal Affairs
21                                   Housing
22                            Land Resources
23                      Micro Finance (SHGs)
24                     Legal Awareness & Aid
25                           Food Processing
26          Micro Small & Medium Enterprises
27                          Animal Husbandry
28                             Biotechnology
29                                 Nutrition
30                             Art & Culture
31                           Minority Issues
32                      Dairying & Fisheries
33                           Water Resources
34                      Science & Technology
35                          Differently Abled
36                          Prisoner's Issues
37                          Dalit Upliftment
38                              Aged/Elderly
39                                 Any Other
```

```
40    Urban Development & Poverty Alleviation
41            Scientific & Industrial Research
42                                     Tourism
43                           Skill Development
dtype: object
```

In [50]:

```
selection = input("Select index of (preferably) one issue (or indices of upto 3 Issues) you are interested in, separated by ',' ind1, ind2, ind3 from above list\n").split(',')

IDInIssuesV = set()
for ind in selection:
    print("Number of NGOs in Issue", IssuesSer[int(ind)], "=", len(IssueToIDD[IssuesSer[int(ind)]]))
    IDInIssuesV = IDInIssuesV.union(set(IssueToIDD[IssuesSer[int(ind)]]))
print("Number of NGOs in any of the Issues =", len(IDInIssuesV))
```

```
Select index of (preferably) one issue (or indic
es of upto 3 Issues) you are interested in, sepa
rated by ',' ind1, ind2, ind3 from above list
35, 37, 27
Number of NGOs in Issue Differently Abled = 1740
0
Number of NGOs in Issue Dalit Upliftment = 14953
Number of NGOs in Issue Animal Husbandry = 18173
Number of NGOs in any of the Issues = 30240
```

Select Region (States or Districts in a State)

In [55]:

```
DistrictsOrStates = str(input("To select up to 3 districts f
rom a single state, type '1', else '0' - you will have the c
hoice of selecting up to 3 states\n"))

if DistrictsOrStates == '1':
    print(StatesL,'\n')

    TheState = str(input("Select ONLY ONE state whose distri
cts you are interested in\n"))

    StateDistL = list(StateDistToIDD[TheState].keys())
    print('\n', StateDistL, '\n')

    selection = str(input("Select upto 3 districts you are i
nterested in from above list, separated by ','\n")).split(
',')

    IDInRegionV = set()
    for dist in selection:
        print("number of NGOs in", dist, "=", len(StateDistT
oIDD[TheState][dist.strip()]))
        IDInRegionV = IDInRegionV.union(set(StateDistToIDD[T
heState][dist.strip()]))
    print("number of NGOs in region = ", len(IDInRegionV))

else:
    print(StatesSer)
    selection = input("\nSelect indices of upto 3 states you
are interested in, separated by ',' ind1, ind2, ind3 from ab
ove list\n").split(',')

    IDInRegionV = set()
    for ind in selection:
        print("number of NGOs in", StatesSer[int(ind)], "=",
len(StateToIDD[StatesSer[int(ind)]]))
        IDInRegionV = IDInRegionV.union(set(StateToIDD[State
sSer[int(ind)]]))
    print("number of NGOs in region =", len(IDInRegionV))
```

```
To select up to 3 districts from a single state,
type '1', else '0' - you will have the choice of
selecting up to 3 states
1
['ORISSA', 'PUDUCHERRY', 'MADHYA PRADESH', 'UTTA
R PRADESH', 'UTTARAKHAND', 'WEST BENGAL', 'JAMMU
& KASHMIR', 'LADAKH', 'MANIPUR', 'TAMIL NADU',
'HARYANA', 'MAHARASHTRA', 'ANDHRA PRADESH', 'RAJ
ASTHAN', 'CHHATTISGARH', 'KARNATAKA', 'DELHI',
'BIHAR', 'KERALA', 'GUJARAT', 'GOA', 'ASSAM', 'T
RIPURA', 'PUNJAB', 'CHANDIGARH', 'NAGALAND', 'JH
ARKHAND', 'MIZORAM', 'SIKKIM', 'MEGHALAYA', 'ARU
NACHAL PRADESH', 'HIMACHAL PRADESH', 'TELANGAN
A', 'LAKSHADWEEP', 'ANDAMAN & NICOBAR ISLANDS',
'DAMAN & DIU', 'DADRA & NAGAR HAVELI']

Select ONLY ONE state whose districts you are in
terested in
JAMMU & KASHMIR

 ['Shupiyan', 'Anantnag', 'Badgam', 'Srinagar',
'Baramula', 'Ramban', 'Pulwama', 'Kulgam', 'Samb
a', 'Punch', 'Reasi', 'Rajouri', 'Doda', 'Gander
bal', 'Udhampur', 'Bandipore', 'Kishtwar', 'Jamm
u', 'Kathua', 'Kupwara']

Select upto 3 districts you are interested in fr
om above list, separated by ','
Baramula, Kishtwar, Kathua
number of NGOs in Baramula = 615
number of NGOs in  Kishtwar = 333
number of NGOs in  Kathua = 388
number of NGOs in region =  767
```

In [58]:

```python
FinalV = IDInIssuesV.intersection(IDInRegionV)
print("Number of NGOs in Issues and region =", len(FinalV))
```

```
Number of NGOs in Issues and region = 311
```

Select tags

In [59]:

```
FCRATag = str(input("Are you a looking to make a donation to an NGO in Foreign Currency?\n'1' for 'Yes' '0' for 'No'\n"))

FCRAReqV = AllIDV
if FCRATag == '1':
    FCRAReqV = FCRAV
```

```
Are you a looking to make a donation to an NGO i
n Foreign Currency?
'1' for 'Yes' '0' for 'No'
1
```

In [60]:

```
URLTag = str(input("Do you want to be able to explore the NGO's website?\n'1' for 'Yes' '0' for 'No'\n"))

URLReqV = AllIDV
if URLTag == '1':
    URLReqV = URLV
```

```
Do you want to be able to explore the NGO's webs
ite?
'1' for 'Yes' '0' for 'No'
1
```

In [61]:

```
MATag = str(input("Would you like to be able to see the NG
O's description of Major Activities?\n'1' for 'Yes' '0' for
 'No'\n"))

MAReqV = AllIDV
if MATag == '1':
    MAReqV = MA1V
```

```
Would you like to be able to see the NGO's descr
iption of Major Activities?
'1' for 'Yes' '0' for 'No'
1
```

Final set

In [62]:

```
FinalV = FinalV.intersection(FCRAReqV)
print("Number of filtered NGOs =", len(FinalV))
```

```
Number of filtered NGOs = 47
```

In [63]:

```
FinalV = FinalV.intersection(URLReqV)
print("Number of filtered NGOs =", len(FinalV))
```

```
Number of filtered NGOs = 33
```

In [64]:

```
FinalV = FinalV.intersection(MAReqV)
print("Number of filtered NGOs =", len(FinalV))
```

```
Number of filtered NGOs = 31
```

Show info for sample of 10

In [66]:

```
FinalL = list(FinalV)
sampleL = random.sample(FinalL, 10)
```

In [67]:

```
DarpanFcraDF.set_index("UniqueID", inplace=True)
```

In [70]:

```
DarpanFcraDF.loc[sampleL][['Name', 'ngo url', 'Email', 'Mobi
le', 'Major Activities1', 'Secretary name', 'Secretary mobil
e', 'Secretary email']]
```

Out[70]:

	Name	ngo url	
UniqueID			
JK/2009/0002051	KASHMIR RESEARCH INSTITUTE OF EDUCATION AND SOLAR	http://www.kriest.in	kries
TN/2016/0111485	Bright Light Society	http://blsngo.org	bl
RJ/2009/0009170	Narayan Sewa Sansthan	http://www.narayanseva.org	pro
JK/2009/0005953	Jammu and Kashmir Habakhatoon Foundation	http://hkf.defindia.org	jk
DL/2017/0161747	INDO GLOBAL SOCIAL SERVICE SOCIETY	http://www.igsss.org	
JK/2017/0115767	Voluntary medicare society	http://voluntarymedicare.org	
GJ/2017/0119442	Shri Vadilal S Gandhi Charitable Trust	http://www.vsgandhitrust.org	

	Name	ngo url	
UniqueID			
UA/2016/0110542	Manav Sewa Samaj	http://manavsewasamaj.org	manavs
JK/2013/0058208	All India Security Council	http://www.aiscnation.org	cha
AP/2009/0002274	Gayatri Rural Educational Society	http://www.gresindia.org	

```
In [ ]:
```