In [1]:

```
import os
import pandas as pd
import numpy as np
import bs4 as bs
import urllib.request
import re
import spacy
import re, string, unicodedata
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
```

```
2023-07-21 02:12:53.199692: I tensorflow/core/platform/cpu_feature_gua
rd.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural
Network Library (oneDNN) to use the following CPU instructions in perf
ormance-critical operations:  SSE4.1 SSE4.2 AVX AVX2 FMA
To enable them in other operations, rebuild TensorFlow with the approp
riate compiler flags.
```

In [2]:

```
lem=WordNetLemmatizer()
```

In [3]:

```
os.chdir(r'/Users/myyntiimac/Desktop/Combine xml file')
```

In [4]:

```
from glob import glob

path=r'/Users/myyntiimac/Desktop/Combine xml file'
all_files = glob(os.path.join(path, "*.xml"))
```

In [5]:

```
import xml.etree.ElementTree as ET
```

In [7]:

```
dfs = []
for filename in all_files:
    tree = ET.parse(filename)
    root = tree.getroot()
    root=ET.tostring(root, encoding='utf8').decode('utf8')
    dfs.append(root)


parsed_article = bs.BeautifulSoup(dfs[0],'xml')

paragraphs = parsed_article.find_all('p')




article_text_full = ""

for p in paragraphs:
    article_text_full += p.text
    print(p.text)




def data_prepracessing(each_file):

    parsed_article = bs.BeautifulSoup(each_file,'xml')

    paragraphs = parsed_article.find_all('para')



    article_text_full = ""

    for p in paragraphs:
        article_text_full += p.text
        print(p.text)

    return article_text_full
```

In [8]:

```
data=[data_prepracessing(each_file) for each_file in dfs]
```

```
Difficulties in social situations (in suspected social anxiety disorde
r)
Irrational and out-of-proportion fear or avoidance of particular objec
ts or situations (in suspected specific phobia)
Intense anxiety reactions with exposure to specific situations (in sus
pected agoraphobia)
Anxiety is the most common feature in phobic disorders. Manifestations
include the following:
Elevated heart rate
Elevated blood pressure
Tremor
Palpitations
Diarrhea
Sweating
Dyspnea
Paresthesias
Dizziness
Because anxiety manifests with a number of physical symptoms, any pati
ent who presents with a de novo complaint of physical symptoms suggest
ive of an anxiety disorder should undergo a physical examination to he
```

In [9]:

```
def remove_stop_word(file):
    # Load the spaCy English language model
    nlp = spacy.load("en_core_web_sm")

    # Define punctuation characters
    punctuations = string.punctuation

    # Load stopwords from NLTK library
    from nltk.corpus import stopwords
    stopwords = stopwords.words('english')

    # Additional symbols to remove
    SYMBOLS = " ".join(string.punctuation).split(" ") + ["-", "...", "”", "”"]
    stopwords = nltk.corpus.stopwords.words('english') + SYMBOLS

    # Apply spaCy NLP pipeline to the file text, disabling parser and named enti
    doc = nlp(file, disable=['parser', 'ner'])

    # Lemmatize tokens and remove pronouns ("-PRON-")
    tokens = [tok.lemma_.lower().strip() for tok in doc if tok.lemma_ != '-PRON-

    # Remove stopwords and punctuation from the tokens
    tokens = [tok for tok in tokens if tok not in stopwords and tok not in punct

    # Lemmatize tokens again using NLTK WordNetLemmatizer
    s = [lem.lemmatize(word) for word in tokens]
    tokens = ' '.join(s)

    # Remove square brackets and extra whitespaces
    article_text = re.sub(r'\[[0-9]*\]', ' ', tokens)
    article_text = re.sub(r'\s+', ' ', article_text)

    # Remove non-alphabetic characters
    formatted_article_text = re.sub('[^a-zA-Z]', ' ', article_text)
    formatted_article_text = re.sub(r'\s+', ' ', formatted_article_text)

    # Remove words with 3 characters or less
    formatted_article_text = re.sub(r'\W*\b\w{1,3}\b', "", formatted_article_tex

    return formatted_article_text

```

In [10]:

```
clean_data=[remove_stop_word(file) for file in data]
clean_data
```

```
erapy social anxiety disorder include following paroxetine sertraline
approved venlafaxine approved escitalopram fluoxetine fluvoxaminephene
lzinemoclobemide approve united states tricyclic antidepressant tcas b
eta blocker propranolol selected anticonvulsant gabapentin pregabalin
valproic acid topiramate tiagabine control study demonstrate efficacy
psychopharmacologic intervention specific phobia need administration s
hort benzodiazepine useful temporary anxiety relief specific situation
agent consider agoraphobia include following ssri escitalopram citalop
ram fluoxetine fluvoxamine paroxetine sertraline venlafaxine reboxetin
esome clomipramine imipramine some benzodiazepine alprazolam lorazepam
diazepam clonazepam mirtazapinemoclobemidepsychotherapeutic interventi
on helpful treat phobic disorder include following social anxiety diso
rder social phobia self exposure monotherapy computer base exposure tr
aining clinician lead exposure combination therapy self exposure cogni
tive behavioral therapy self help manual specific phobia base approach
include gradual desensitization relaxation breathing control technique
exposure therapy agoraphobia combination exposure therapy relaxation b
reathe retrainingsee treatment medication detail phobia define irratio
nal fear produce conscious avoidance fear subject activity situation a
ffected person usually recognize reaction excessive collectively phobi
```

In [ ]:

```
#vectorizing the clean data
```

In [11]:

```
from sklearn.feature_extraction.text import  CountVectorizer
from sklearn.preprocessing import normalize
import pandas
```

In [12]:

```
vectorizer = CountVectorizer(stop_words=stopwords.words('english'),ngram_range=(
vectorizer
```

Out[12]:

```
CountVectorizer(ngram_range=(2, 2),
                stop_words=['i', 'me', 'my', 'myself', 'we', 'our', 'o
urs',
                            'ourselves', 'you', "you're", "you've", "y
ou'll",
                            "you'd", 'your', 'yours', 'yourself', 'you
rselves',
                            'he', 'him', 'his', 'himself', 'she', "sh
e's",
                            'her', 'hers', 'herself', 'it', "it's", 'i
ts',
                            'itself', ...])
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [17]:

```
X=vectorizer.transform(clean_data)
X
```

Out[17]:

```
<30x66449 sparse matrix of type '<class 'numpy.int64'>'
        with 72731 stored elements in Compressed Sparse Row format>
```

In [18]:

```
# Get the feature names using get_feature_names_out
feature_names = vectorizer.get_feature_names_out()
```

In [19]:

```
# Convert the sparse matrix to a DataFrame using the feature names
data_final = pd.DataFrame(X.toarray(), columns=feature_names)
data_final
```

Out[19]:

	aase smith	aasly aharon	abandon lead	abbass premaxillary	abbott barrett	abbott melhem	abbreviation head	abdelrahman abbass	abdoman chest
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	1	0	0	1	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	1	0	0
10	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	0	0
14	0	0	0	0	1	0	0	0	0
15	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0
20	0	0	1	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	1	0	0	0
28	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0

30 rows × 66449 columns

In [20]:

```
from sklearn.feature_extraction.text import TfidfTransformer

tran=TfidfTransformer().fit(data_final.values)

X=tran.transform(X).toarray()

X = normalize(X)
```

In [21]:

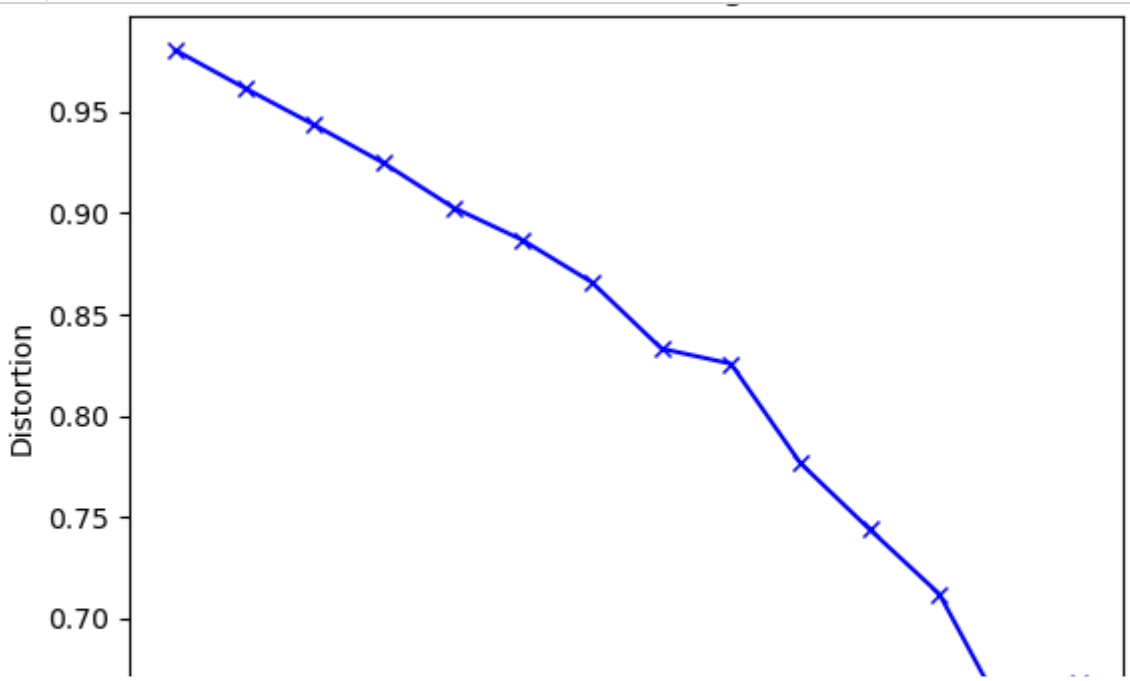
```
X
```

Out[21]:

```
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.]])
```

In [22]:

```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt


distortions = []
inertias = []
mapping1 = {}
mapping2 = {}
K = range(1,15)

for k in K:
    #Building and fitting the model
    kmeanModel = KMeans(n_clusters=k).fit(X)
    kmeanModel.fit(X)

    distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_,
                      'euclidean'),axis=1)) / X.shape[0])
    inertias.append(kmeanModel.inertia_)

    mapping1[k] = sum(np.min(cdist(X, kmeanModel.cluster_centers_,
                 'euclidean'),axis=1)) / X.shape[0]
    mapping2[k] = kmeanModel.inertia_

for key,val in mapping1.items():
    print(str(key)+' : '+str(val))

plt.plot(K, distortions, 'bx-')
plt.xlabel('Values of K')
plt.ylabel('Distortion')
plt.title('The Elbow Method using Distortion')
plt.show()

```



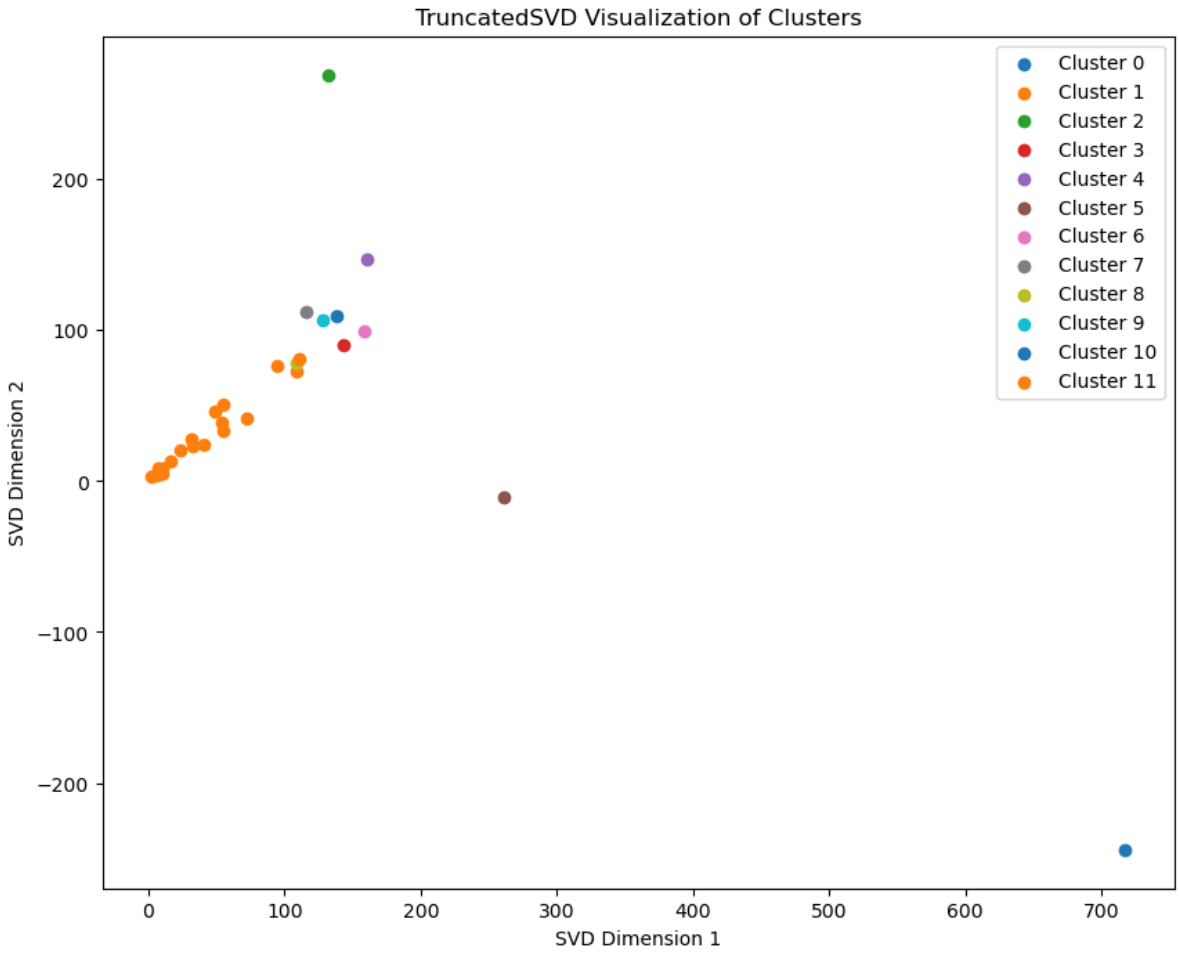
In [34]:

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.cluster import KMeans
import pandas as pd

# Assuming you have a variable 'text_data' that contains the text data as a list

# Initialize the CountVectorizer
vectorizer = CountVectorizer()

# Fit and transform the text data 'text_data'
X_vectorized = vectorizer.fit_transform(clean_data)

# Define the number of clusters ('true_k') and create the KMeans model
true_k = 12
model = KMeans(n_clusters=true_k, init='k-means++', max_iter=100, n_init=1)

# Fit the KMeans model to the vectorized data
model.fit(X_vectorized)

# Create a DataFrame to store the vectorized data and cluster labels
cluster_result_data = pd.DataFrame(clean_data, columns=['text'])
cluster_result_data['group'] = model.predict(X_vectorized)

# Get the feature names using get_feature_names_out
terms = vectorizer.get_feature_names_out()

# Get the cluster centers and print the top 50 terms for each cluster
order_centroids = model.cluster_centers_.argsort()[:, ::-1]

for i in range(true_k):
    print('Cluster %d:' % i)
    for ind in order_centroids[i, :50]:
        print(' %s' % terms[ind])

```

```
 failure
 wall
 develop
 surgical
 early
 infarct
 anterior
 severe
 increase
 usually
 muscle
 intervention
 treat
 cardiogenic
 first
 mitral
 vfwr
Cluster 11:
 heat
 patient
```

In [38]:

```
import matplotlib.pyplot as plt
from sklearn.decomposition import TruncatedSVD

# Dimensionality reduction using TruncatedSVD
svd = TruncatedSVD(n_components=2, random_state=42)
X_svd = svd.fit_transform(X_vectorized)

# Get the cluster labels from the fitted KMeans model
cluster_labels = model.labels_

# Plot the clusters using a scatter plot
plt.figure(figsize=(10, 8))
for i in range(true_k):
    plt.scatter(X_svd[cluster_labels == i, 0], X_svd[cluster_labels == i, 1], la

plt.title('TruncatedSVD Visualization of Clusters')
plt.xlabel('SVD Dimension 1')
plt.ylabel('SVD Dimension 2')
plt.legend()
plt.show()

```



In [ ]:

```

```