```
from google.colab import drive
drive.mount('/content/drive')
```

```
    Mounted at /content/drive
```

```
!pip install sentence-transformers
```

```
    Collecting sentence-transformers
      Downloading https://files.pythonhosted.org/packages/f5/5a/6e41e8383913dd2ba923cdcd
         |████████████████████████████████| 71kB 5.5MB/s
    Collecting transformers<3.6.0,>=3.1.0
      Downloading https://files.pythonhosted.org/packages/3a/83/e74092e7f24a08d751aa59b3
         |████████████████████████████████| 1.3MB 12.8MB/s
    Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from
    Requirement already satisfied: torch>=1.6.0 in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from
    Requirement already satisfied: scikit-learn in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from
    Requirement already satisfied: nltk in /usr/local/lib/python3.6/dist-packages (from
    Collecting sentencepiece==0.1.91
      Downloading https://files.pythonhosted.org/packages/d4/a4/d0a884c4300004a78cca907a
         |████████████████████████████████| 1.1MB 35.3MB/s
    Requirement already satisfied: requests in /usr/local/lib/python3.6/dist-packages (f
    Requirement already satisfied: dataclasses; python_version < "3.7" in /usr/local/lib
    Collecting sacremoses
      Downloading https://files.pythonhosted.org/packages/7d/34/09d19aff26edcc8eb2a01bed
         |████████████████████████████████| 890kB 50.9MB/s
    Collecting tokenizers==0.9.3
      Downloading https://files.pythonhosted.org/packages/4c/34/b39eb9994bc3c999270b69c9
         |████████████████████████████████| 2.9MB 49.4MB/s
    Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: packaging in /usr/local/lib/python3.6/dist-packages (
    Requirement already satisfied: protobuf in /usr/local/lib/python3.6/dist-packages (f
    Requirement already satisfied: filelock in /usr/local/lib/python3.6/dist-packages (f
    Requirement already satisfied: typing-extensions in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: future in /usr/local/lib/python3.6/dist-packages (fro
    Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from n
    Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-pa
    Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-p
    Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-package
    Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local
    Requirement already satisfied: click in /usr/local/lib/python3.6/dist-packages (from
    Requirement already satisfied: pyparsing>=2.0.2 in /usr/local/lib/python3.6/dist-pac
    Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages
    Building wheels for collected packages: sentence-transformers, sacremoses
      Building wheel for sentence-transformers (setup.py) ... done
      Created wheel for sentence-transformers: filename=sentence_transformers-0.3.9-cp36
      Stored in directory: /root/.cache/pip/wheels/fc/89/43/f2f5bc00b03ef9724b0f6254a97e
      Building wheel for sacremoses (setup.py) ... done
      Created wheel for sacremoses: filename=sacremoses-0.0.43-cp36-none-any.whl size=89
      Stored in directory: /root/.cache/pip/wheels/29/3c/fd/7ce5c3f0666dab31a50123635e6f
    Successfully built sentence-transformers sacremoses
    Installing collected packages: sentencepiece, sacremoses, tokenizers, transformers,
    Successfully installed sacremoses-0.0.43 sentence-transformers-0.3.9 sentencepiece-0
```

```
# import all the necessary libraries
# import all the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
from sklearn.cluster import DBSCAN
import string
import unicodedata
# from sklearn.feature_extraction.text import TfidfVectorizer
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
import spacy
from sentence_transformers import SentenceTransformer
from sklearn.manifold import TSNE
```

```
# loading the dataset
train=pd.read_csv("/content/drive/My Drive/3rd Sem/Code v0.2/excel_data/summarydata-xlnet-
```

```
train.head()
```

	News_ID	GPT-2	XLNET
0	1	Abu Dhabi [UAE], October 7 (ANI): USA pacer Al...	Abu Dhabi [UAE], October 7 (ANI): USA pacer Al...
1	2	Abu Dhabi [UAE], October 6 (ANI): England and ...	Abu Dhabi [UAE], October 6 (ANI): England and ...
2	3	Sydney [Australia], October 7 (ANI): Arjun Nai...	Sydney [Australia], October 7 (ANI): Arjun Nai...

```
train.dropna(inplace=True)
```

```
train.isnull().sum()
```

```
    News_ID    0
    GPT-2      0
    XLNET      0
    dtype: int64
```

```
#convert each question to a list of string
data = pd.Series(train["XLNET"].tolist()).astype(str)
```

```
data.head()
```

```
    0    Abu Dhabi [UAE], October 7 (ANI): USA pacer Al...
    1    Abu Dhabi [UAE], October 6 (ANI): England and ...
    2    Sydney [Australia], October 7 (ANI): Arjun Nai...
    3    Sydney [Australia], October 7 (ANI): Sydney Th...
    4    Abu Dhabi [UAE], October 6 (ANI): Mumbai India...
    dtype: object
```

```
data1 = data[:100]
```

```
sentences_list = data
```

Text Preprocessing

```
nlp = spacy.load('en_core_web_sm')
# stop_list = ['best','different',"won\'t", "couldn\'t", "mustn\'t", "didn\'t", "dtype obj
# for word in stop_list:
#     spacy.lang.en.stop_words.STOP_WORDS.add(word)
#     nlp.vocab[word].is_stop = True


def normalize(data):
    """Run all the functions for preprocessing in a pipeline"""
    clean_data = re.sub(re.compile('<.*?>'), '', data)
    cleaned_list = [ unicodedata.normalize('NFKD', word.text).encode('ascii', 'ignore').de
    cleaned_list = " ".join(cleaned_list)
    cleaned_list = [word.text.rstrip('0123456789').lower() for word in nlp(cleaned_list) i
    return cleaned_list


# Preprocess the text data
normalized_data = []
for i, batch in data.groupby(np.arange(len(data)) // 10):
    for batch_data in batch:
        normalized_data.append(normalize(batch_data))

    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
```

```
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
```

```
# Data after prerocessing
print(normalized_data[0])
len(normalized_data)
```

```
['abu', 'dhabi', 'uae', 'october', 'ani', 'usa', 'pacer', 'ali', 'khan', 'ruled', 'i
9048
```

```
# function to form sentences from token
sentence = []
sentences = []


def token_2_sentence(normalized_data):
    """Join the tokens in each list with space to form a sentence"""
    for i in normalized_data:
      sentence = " ".join(i)
      sentences.append(sentence)
      sentence = []
    return sentences

sentences_list = token_2_sentence(normalized_data)


sentences_list[:10]
```

```
sentences_list[:10]
```

```
['abu dhabi uae october ani usa pacer ali khan ruled indian premier league ipl injur
 'abu dhabi uae october ani england rajasthan royals rounder ben stokes reckons kart
 'sydney australia october ani arjun nair signed big bash league bbl season sydney t
 'sydney australia october ani sydney thunder completed squad women big bash league
 'abu dhabi uae october ani mumbai indians brigade continued impress edition indian
 'abu dhabi uae october ani reminiscing catch dismiss rajasthan royals mahipal lomro
 'adelaide australia october ani west indies captain stafanie taylor rejoin adelaide
 'abu dhabi uae october ani rajasthan royals skipper steve smith fined maintaining r
 'abu dhabi uae october ani registering win rajasthan royals mumbai indians bowling
 'new delhi india october ani india head coach ravi shastri rounder yuvraj singh pra
```

```
import csv
with open('./normalized.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow("Normalized")
```

```
for item in sentences_list:
  with open('./normalized.csv', 'a', newline='') as file:
      writer = csv.writer(file)
      writer.writerow([item])
```

```
# sentences_list = pd.read_csv("./normalized.csv")
# sentences_list = sentences_list.values.tolist()
# sentences_list[0]
```

```
# sentences_list[0][0]
```

Clustering with Embedding

```
model = SentenceTransformer('distilbert-base-nli-mean-tokens')
```

```
100%|██████████| 245M/245M [00:16<00:00, 14.9MB/s]
```

```
def data_gen(data):
    for sen in data:
      yield sen
```

```
a = data_gen(sentences_list)
```

```
encoding_arr = list()
current = 1
for item in a:
  embeddings = model.encode(item)
  encoding_arr.append(embeddings)
  print("Current:",current)
  current += 1
```

```
Streaming output truncated to the last 5000 lines.
Current: 4049
Current: 4050
Current: 4051
Current: 4052
Current: 4053
Current: 4054
Current: 4055
Current: 4056
Current: 4057
Current: 4058
Current: 4059
Current: 4060
Current: 4061
Current: 4062
Current: 4063
Current: 4064
Current: 4065
Current: 4066
Current: 4067
Current: 4068
Current: 4069
Current: 4070
Current: 4071
Current: 4072
Current: 4073
Current: 4074
Current: 4075
Current: 4076
Current: 4077
Current: 4078
Current: 4079
Current: 4080
Current: 4081
Current: 4082
Current: 4083
Current: 4084
Current: 4085
Current: 4086
Current: 4087
Current: 4088
Current: 4089
Current: 4090
Current: 4091
Current: 4092
Current: 4093
Current: 4094
Current: 4095
Current: 4096
Current: 4097
Current: 4098
Current: 4099
Current: 4100
Current: 4101
Current: 4102
Current: 4103
Current: 4104
Current: 4105
Current: 4106
```

```
encoded_arr = np.array(encoding_arr)
encoded_arr_gpt2 = encoded_arr
encoded_arr_gpt2.shape
```

```
(9048, 768)
```

```
# from gensim.models.doc2vec import Doc2Vec, TaggedDocument
```

```
def tagged_document(normalized_data):
    tagged_corpus = []
    tagged_corpus = [TaggedDocument(words = d, tags=[str(i)]) for i,d in enumerate(normali
    return tagged_corpus
```

```
tagged_corpus =  tagged_document(normalized_data)
```

```
tagged_corpus
```

```
[TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'usa', 'pacer', '
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'england', 'rajas
 TaggedDocument(words=['sydney', 'australia', 'october', 'ani', 'arjun', 'nair',
 TaggedDocument(words=['sydney', 'australia', 'october', 'ani', 'sydney', 'thunde
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'mumbai', 'indian
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'reminiscing', 'c
 TaggedDocument(words=['adelaide', 'australia', 'october', 'ani', 'west', 'indies
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'rajasthan', 'roy
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'registering', 'w
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'india', 'head'
 TaggedDocument(words=['paris', 'france', 'october', 'ani', 'argentina', 'diego',
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'mumbai', 'indian
 TaggedDocument(words=['london', 'uk', 'october', 'ani', 'expressing', 'sadness',
 TaggedDocument(words=['mumbai', 'maharashtra', 'india', 'october', 'ani', 'west'
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'onslaught', 'sur
 TaggedDocument(words=['bhubaneswar', 'odisha', 'india', 'october', 'ani', 'odish
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'kolkata', 'knigh
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'playing', 'match
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'facing', 'defeat
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'mumbai', 'indian
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'stumbling', 'def
 TaggedDocument(words=['paris', 'france', 'october', 'ani', 'rafael', 'nadal', 't
 TaggedDocument(words=['dubai', 'uae', 'october', 'ani', 'delhi', 'capitals', 'si
 TaggedDocument(words=['los', 'angeles', 'october', 'ani', 'facing', 'loss', 'gam
 TaggedDocument(words=['baidurjo', 'bhosedubai', 'uae', 'october', 'ani', 'wins',
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'kiren', 'rijij
 TaggedDocument(words=['brisbane', 'australia', 'october', 'ani', 'women', 'team'
 TaggedDocument(words=['mumbai', 'maharashtra', 'india', 'october', 'ani', 'onlin
 TaggedDocument(words=['uk', 'october', 'ani', 'celtic', 'football', 'club', 'wed
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'india', 'royal
 TaggedDocument(words=['dhaka', 'bangladesh', 'october', 'ani', 'bangladesh', 'cr
 TaggedDocument(words=['mumbai', 'maharashtra', 'india', 'october', 'ani', 'nba',
 TaggedDocument(words=['bristol', 'uk', 'october', 'ani', 'gloucestershire', 'cri
 TaggedDocument(words=['bengaluru', 'karnataka', 'india', 'october', 'ani', 'men'
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'scalping', 'wick
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'skipper', 'vir
 TaggedDocument(words=['uk', 'october', 'ani', 'england', 'county', 'cricket', 'c
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'india', 'men',
 TaggedDocument(words=['liverpool', 'uk', 'october', 'ani', 'liverpool', 'midfiel
 TaggedDocument(words=['dubai', 'uae', 'october', 'ani', 'delhi', 'capitals', 'sp
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'batsman', 'cri
 TaggedDocument(words=['melbourne', 'australia', 'october', 'ani', 'batsman', 'de
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'kolkata', 'knigh
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'rajasthan', 'r
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'testing', 'cor
 TaggedDocument(words=['rome', 'italy', 'october', 'ani', 'expressing', 'elation'
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'registering', 'w
 TaggedDocument(words=['uk', 'october', 'ani', 'stuart', 'armstrong', 'tested', '
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'stumbling', 'los
 TaggedDocument(words=['london', 'uk', 'october', 'ani', 'arsenal', 'kieran', 'ti
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'playing', 'match
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'started', 'injur
 TaggedDocument(words=['paris', 'france', 'october', 'ani', 'world', 'number', 'n
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'kolkata', 'knigh
 TaggedDocument(words=['al', 'khor', 'qatar', 'october', 'ani', 'fifa', 'presiden
 TaggedDocument(words=['abu', 'dhabi', 'uae', 'october', 'ani', 'suffering', 'def
 TaggedDocument(words=['new', 'delhi', 'india', 'october', 'ani', 'boxer', 'mary'
```

```
def build_model(tagged_corpus,max_epochs,vec_size, alpha):
    model = Doc2Vec(size=vec_size, alpha=alpha,min_alpha=0.001, min_count=1,dm =1)
    model.build_vocab(tagged_corpus)

    for epoch in range(max_epochs):
        model.train(tagged_corpus,total_examples=model.corpus_count, epochs=model.iter)
        # decrease the learning rate
        model.alpha -= 0.002
        # fix the learning rate, no decay
        model.min_alpha = model.alpha


    model.save("d2v.model")
    print("Model Saved")
    model_name = "d2v.model"
    return model_name



# from gensim.models.doc2vec import Doc2Vec

def load_model(model_name, data):
    corpus_vector = []
    model= Doc2Vec.load(model_name)
    for doc in data:
        corpus_vector.append(model.infer_vector(doc.split()))
    return corpus_vector


max_epochs = 100
vec_size = 100
alpha = 0.001
model_name = build_model(tagged_corpus,max_epochs,vec_size, alpha)
```

```
/usr/local/lib/python3.6/dist-packages/gensim/models/doc2vec.py:570: UserWarning: Th
  warnings.warn("The parameter `size` is deprecated, will be removed in 4.0.0, use `
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:7: DeprecationWarning:
  import sys
Model Saved
```

```
corpus_vector = load_model("d2v.model",data)
```

```
corpus_vector = np.array(corpus_vector)
```

```
corpus_vector.shape
```

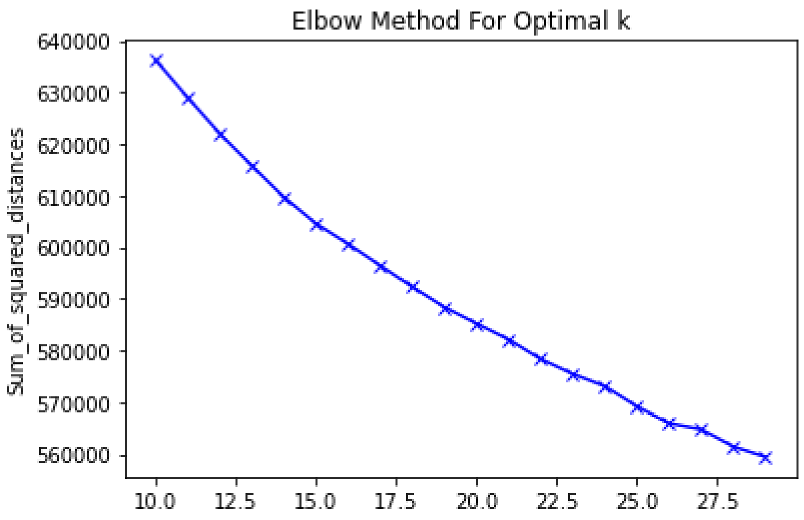
```
(9048, 100)
```

```
#KMeans (WITHOUT Dimensionality Reduction)

from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

def kmeans(corpus_vector):
    """Function to form dbscan clusters and display them"""
#     eps = 0.005# how close points should be to each other to be considered a part of a c
#     min_samples = 3# the minimum number of points to form a dense region
#     dbscan = DBSCAN( eps=eps, min_samples=min_samples,metric = "cosine" )
#     dbscan_model = dbscan.fit(corpus_vector)

    # pca = PCA(n_components=2)
    # result = pca.fit_transform(corpus_vector)
    # print(result.shape)

    Sum_of_squared_distances = []
    K = range(10,30)
    for k in K:
      km = KMeans(n_clusters=k, max_iter=200, n_init=10)
      km = km.fit(corpus_vector)
      Sum_of_squared_distances.append(km.inertia_)
      print(Sum_of_squared_distances[-1])
    plt.plot(K, Sum_of_squared_distances, 'bx-')
    plt.xlabel('k')
    plt.ylabel('Sum_of_squared_distances')
    plt.title('Elbow Method For Optimal k')
    plt.show()
```

```
#K-Means on BERT Embedding
```

```
kmeans(encoded_arr_gpt2)
```

```
636226.0324123233
628973.8118242959
622017.4900697161
615828.3449469479
609725.3463181213
604594.2566440391
600650.9279740779
596480.4479859769
592284.740296833
588372.4161972597
585212.4296981939
582143.8975672007
578342.2957697631
575470.4618724076
573107.9021732396
569223.0575977203
566010.5284941453
564815.6141669714
561504.6678504735
559502.8424815659
```

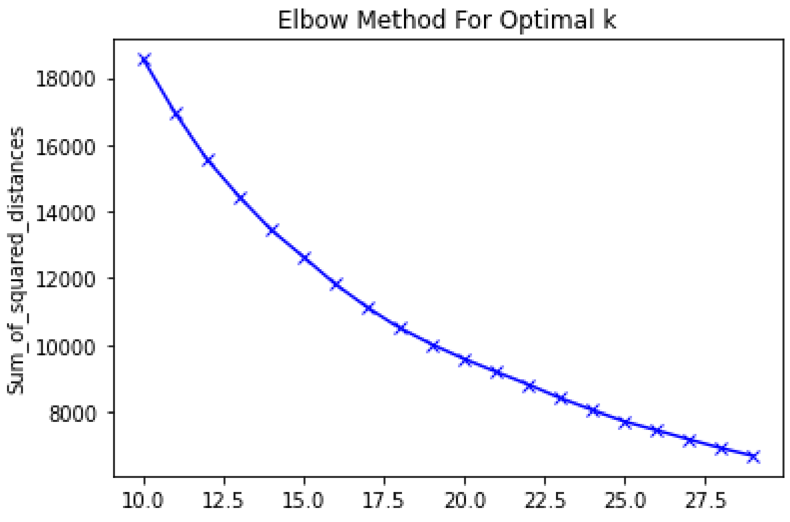


```
#K-Means on Doc2Vec Embedding


kmeans(corpus_vector)
```

```
     595.2052971046071
     498.3961963259021
     424.85619221293695
     363.90301615043563
     314.131308092469
     273.30019552791697
     241.0352172027823
     212.70686324965433
     184.70589215744454
     161.50985955189276
     141.5167325296013
     127.22148430703739
     115.70601297996417
     106.87262962339061
     97.41163393995076
     90.07300019001077
     82.86541267159937
     77.4268748899201
```

```
#KMeans (WITH Dimensionality Reduction PCA)

from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

def kmeans_pca(corpus_vector):
    """Function to form dbscan clusters and display them"""
#     eps = 0.005# how close points should be to each other to be considered a part of a c
#     min_samples = 3# the minimum number of points to form a dense region
#     dbscan = DBSCAN( eps=eps, min_samples=min_samples,metric = "cosine" )
#     dbscan_model = dbscan.fit(corpus_vector)

    pca = PCA(n_components=2)
    result = pca.fit_transform(corpus_vector)
    print(result.shape)

    Sum_of_squared_distances = []
    K = range(10,30)
    for k in K:
      km = KMeans(n_clusters=k, max_iter=200, n_init=10)
      km = km.fit(result)
      Sum_of_squared_distances.append(km.inertia_)
      print(k,":",Sum_of_squared_distances[-1])
    plt.plot(K, Sum_of_squared_distances, 'bx-')
    plt.xlabel('k')
    plt.ylabel('Sum_of_squared_distances')
    plt.title('Elbow Method For Optimal k')
    plt.show()




##K-Means on BERT Embedding + PCA
kmeans_pca(encoded_arr_gpt2)
```

```
(9048, 2)
10 : 18560.953165198403
11 : 16941.98028871723
12 : 15560.145491031366
13 : 14447.949595274118
14 : 13448.976200781664
15 : 12648.336964584621
16 : 11823.031852330876
17 : 11127.175079239358
18 : 10508.065434639233
19 : 10016.186003388093
20 : 9584.227048580493
21 : 9194.963335656228
22 : 8815.940349506627
23 : 8408.708616932508
24 : 8054.366870214579
25 : 7722.770213917418
26 : 7456.347378347164
27 : 7181.7400829122325
28 : 6922.813370109593
29 : 6689.579809549076
```

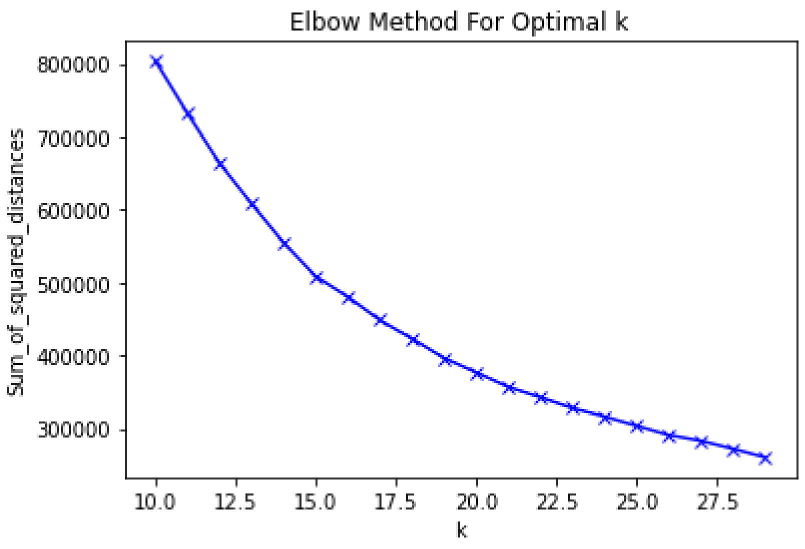


```
##K-Means on Doc2Vec Embedding + PCA
kmeans_pca(corpus_vector)
```

```
     (9048, 2)
     10 : 589.8270535416361
     11 : 492.68293239377005
     12 : 417.731480173745
     13 : 356.9957108851029
     14 : 307.4881037861577
     15 : 267.8127340112212
     16 : 236.3501422643724
     17 : 205.12716869565094
     18 : 177.8361509148462
     19 : 154.310813229925
     20 : 134.00469789229328
     21 : 120.58393720231933
     22 : 109.90203646650781
     23 : 99.45126063720592
     24 : 90.11679269751588
     25 : 82.86848935640349
     26 : 77.12506397161907
```

```
#KMeans (WITH Dimensionality Reduction T-SNE)

from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

def kmeans_tsne(corpus_vector):
    """Function to form dbscan clusters and display them"""
#     eps = 0.005# how close points should be to each other to be considered a part of a c
#     min_samples = 3# the minimum number of points to form a dense region
#     dbscan = DBSCAN( eps=eps, min_samples=min_samples,metric = "cosine" )
#     dbscan_model = dbscan.fit(corpus_vector)

   # Initialize t-SNE
    tsne = TSNE(n_components = 2, init = 'random', random_state = 10, perplexity = 100)
    # Use only 400 rows to shorten processing time
    result = tsne.fit_transform(corpus_vector)
    print(result.shape)

    Sum_of_squared_distances = []
    K = range(10,30)
    for k in K:
      km = KMeans(n_clusters=k, max_iter=200, n_init=10)
      km = km.fit(result)
      Sum_of_squared_distances.append(km.inertia_)
      print(k,":",Sum_of_squared_distances[-1])
    plt.plot(K, Sum_of_squared_distances, 'bx-')
    plt.xlabel('k')
    plt.ylabel('Sum_of_squared_distances')
    plt.title('Elbow Method For Optimal k')
    plt.show()



##K-Means on BERT Embedding + t-SNE
kmeans_tsne(encoded_arr_gpt2)
```

```
(9048, 2)
10 : 803553.9036114095
11 : 731874.6509677971
12 : 664052.4382794974
13 : 608375.1049985159
14 : 555030.3500704686
15 : 508757.1024201889
16 : 480702.5714086454
17 : 448770.7673411322
18 : 422944.3448997744
19 : 396396.9741545371
20 : 376731.21501901274
21 : 356827.59344433196
22 : 342695.1671310902
23 : 327992.25940867746
24 : 316552.04825420957
25 : 304099.1789397448
26 : 291423.7453893733
27 : 283406.0512863829
28 : 272492.583235071
29 : 260790.82284315466
```



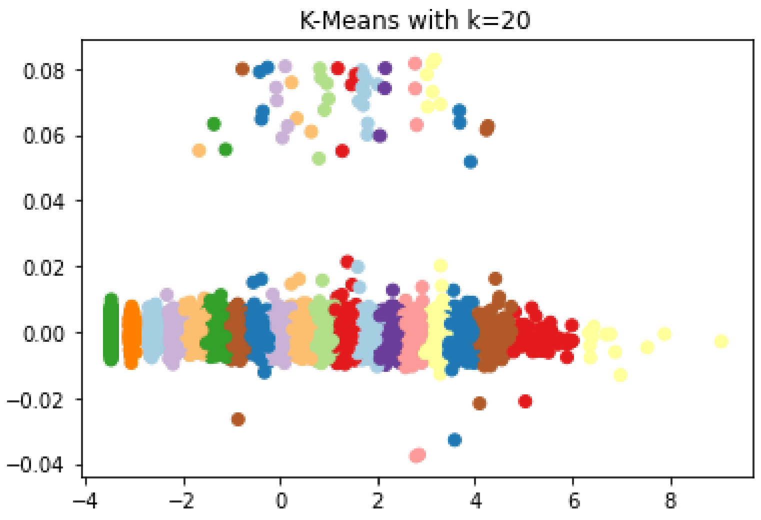
```
##K-Means on Doc2Vec Embedding + t-SNE
kmeans_tsne(corpus_vector)
```

```
    (9048, 2)
    10 : 1310723.232258192
    11 : 1127150.2634369275
    12 : 983638.7330644688
    13 : 864367.1744404761
    14 : 767983.2299486648
    15 : 678309.9839990888
    16 : 600015.4224532119
    17 : 533924.7962152096
    18 : 469093.9064445958
    19 : 417351.54656462226
    20 : 374410.48544527567
    21 : 340791.01442558883
    22 : 312374.88933397114
    23 : 286989.1207015667
    24 : 268047.0972165676
    25 : 248336.53097968714
    26 : 229237.0255422142
    27 : 216141.1147729199
```

```
def plot_kmeans_pca(true_k, corpus_vector):
  pca = PCA(n_components=2)
  result_pca = pca.fit_transform(corpus_vector)
  print(result_pca.shape)

  model = KMeans(n_clusters=true_k, init='k-means++', max_iter=200, n_init=10)
  model.fit(result_pca)
  print("SSD:",model.inertia_)
  labels=model.labels_
  print("Labels:",labels)
  y_pred = model.fit_predict(result_pca)
  plt.scatter(result_pca[:,0], result_pca[:,1],c=y_pred, cmap='Paired')
  plt.title("K-Means with k="+str(true_k))
  return labels
```

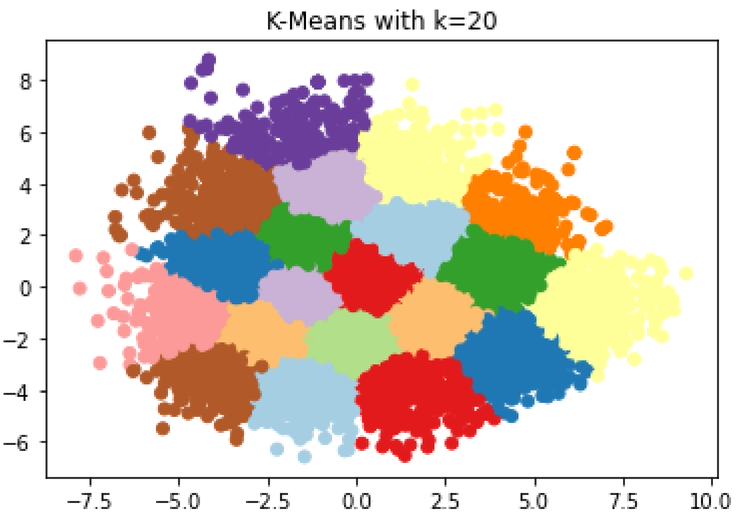
```
 labels_kmeans_pca = plot_kmeans_pca(20, corpus_vector)
```

```
    (9048, 2)
    SSD: 133.8402222337754
    Labels: [ 3 16  2 ... 19  4  9]
```



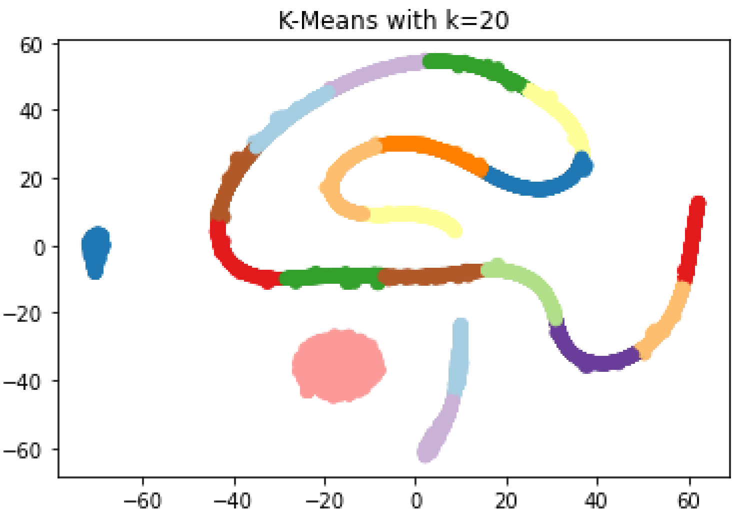
```
plot_kmeans_pca(20, encoded_arr_gpt2)
```

```
(9048, 2)
SSD: 9582.360340157447
Labels: [ 1 18  9 ... 17 16 10]
array([ 1, 18,  9, ..., 17, 16, 10], dtype=int32)
```



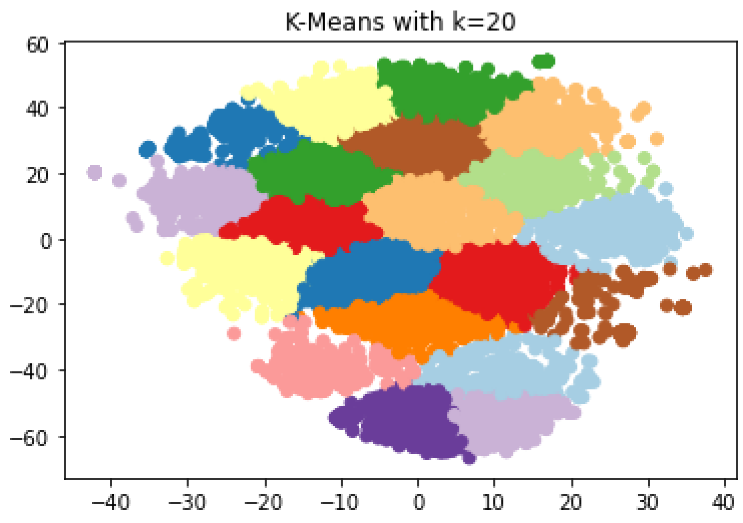
```
def plot_kmeans_tsne(true_k, corpus_vector):
  tsne = TSNE(n_components = 2, init = 'random', random_state = 10, perplexity = 100)
  # Use only 400 rows to shorten processing time
  result_tsne = tsne.fit_transform(corpus_vector)
  print(result_tsne.shape)

  model = KMeans(n_clusters=true_k, init='k-means++', max_iter=200, n_init=10)
  model.fit(result_tsne)
  print("SSD:",model.inertia_)
  labels=model.labels_
  print(labels)
  y_pred = model.fit_predict(result_tsne)
  plt.scatter(result_tsne[:,0], result_tsne[:,1],c=y_pred, cmap='Paired')
  plt.title("K-Means with k="+str(true_k))


plot_kmeans_tsne(20, corpus_vector)
```

```
(9048, 2)
SSD: 375280.9844821786
[ 2 15 10 ...  3 18  9]
```



```
plot_kmeans_tsne(20, encoded_arr_gpt2)
```

```
    (9048, 2)
    SSD: 375852.4768484771
    [ 3  3 15 ...  8  5  6]
```



```
labels1 = labels_kmeans_pca.tolist()
```

```
total = 0
for each in range(-1, 21):
  print(each, ":", labels1.count(each))
  total += labels1.count(each)

print(total)
```

```
    -1 : 0
    0 : 1050
    1 : 647
    2 : 546
    3 : 378
    4 : 537
    5 : 192
    6 : 428
    7 : 427
    8 : 354
    9 : 653
    10 : 563
    11 : 12
    12 : 526
    13 : 324
    14 : 509
    15 : 281
    16 : 400
    17 : 643
    18 : 54
    19 : 524
    20 : 0
    9048
```

```
list3 = [i for i, e in enumerate(labels1) if e == 2]
```

```
combined_sent = ""
for each in list3:
  print(sentences_list[each])
  print()
  combined_sent += sentences_list[each]
```

```
sydney australia october ani arjun nair signed big bash league bbl season sydney

abu dhabi uae october ani reminiscing catch dismiss rajasthan royals mahipal lomr

new delhi india october ani players teams participating egoal social responsibili

new delhi india october ani sports authority india approved month coaching camp c

new delhi india october ani spin wizard brad hogg praised kolkata knight riders s

dubai uae october ani sunrisers hyderabad srh skipper david warner praised rashid

students opportunity participate topics match analysis strength conditioning sess

dubai uae october ani kings xi punjab bowling coach charl langeveldt impressed sp

birmingham uk october ani aston villa women team member tested coronavirus club a

abu dhabi uae october ani delhi capitals suffered blow wicketkeeper batsman risha

new delhi india october ani union minister youth sports kiren rijiju thursday att

tasmania australia october ani england dawid malan joined hobart hurricanes seaso

patna bihar india september ani lok janshakti party chief chirag paswan candidate

new delhi india october ani meeting bharatiya janata party bjp leadership union h

panaji goa india october ani farmers country welcomed laws passed parliament oppo

vijayawada andhra pradesh india october ani union finance minister nirmala sithar

patna bihar india october ani ljp deciding contest polls nda bihar bjp leader sus

chennai tamil nadu india october ani edappadi k palaniswami eps named candidate i

new delhi india october ani condemning treatment west bengal police bjp workers n

patna bihar india october ani rjd leader tej pratap yadav sunday received party s

new delhi india october ani run west bengal assembly polls congress revamped stat

bhopal madhya pradesh india october ani madhya pradesh chief minister kamal nath

dehradun uttarakhand india october ani bharatiya janata party bjp ward councillor

hyderabad telangana india october ani congress party slammed ruling telangana ras

bhopal madhya pradesh india october ani senior congress leader digvijaya singh th

thoothukudi tamil nadu india october ani tamil nadu police filed fir people aiadm

washington october ani look trailer news world starring actor tom hanks released
washington october ani pixar soul skipping theatres debut disney time christmas a

washington october ani considering poster boy privilege rock band founder singer
```

```
wordlist = combined_sent.split()
wordfreq = {}
for w in wordlist:
  if w not in wordfreq:
    wordfreq[w] = 0
  wordfreq[w] += 1
```

```
sorted_words = dict(sorted(wordfreq.items(), key=lambda item: item[1],reverse=True))
print(sorted_words)
```

```
{'said': 372, 'india': 168, 'october': 141, 'oct': 114, 'delhi': 109, 'government':
```

```
# kmeans_pca(encoded_arr)
```

```
# kmeans(encoded_arr)
```

```
from sklearn.decomposition import PCA

def dbscan(corpus_vector, eps= 0.005, min_samples = 3):
    """Function to form dbscan clusters and display them"""
#     eps = 0.005# how close points should be to each other to be considered a part of a c
#     min_samples = 3# the minimum number of points to form a dense region
#     dbscan = DBSCAN( eps=eps, min_samples=min_samples,metric = "cosine" )
#     dbscan_model = dbscan.fit(corpus_vector)

    pca = PCA(n_components=2)
    result = pca.fit_transform(corpus_vector)
    print(result.shape)
    db = DBSCAN(eps=eps, min_samples=min_samples)
    dbscan_model = db.fit(result)
    #Forming the clusters

    core_samples_mask = np.zeros_like(dbscan_model.labels_, dtype=bool)
    core_samples_mask[dbscan_model.core_sample_indices_] = True
    labels1 = dbscan_model.labels_
    n_clusters_ = len(set(labels1)) - (1 if -1 in labels1 else 0) # Number of clusters in
    print(labels1)
    print(len(labels1))
    print(n_clusters_) # number of clusters

    clusters1 = {} # a dictionary for different cluster
    for c, i in enumerate(labels1):
        if i == -1:
            continue
        elif i in clusters1:
            clusters1[i].append( data[c] )
        else:
            clusters1[i] = [data[c]]

    for c in clusters1: # print the different clusters
        # print("Cluster No."+" "+str(c)+" "+str(clusters1[c]))
        # print()
        pass

    return labels1, clusters1


labels1, clusters1 = dbscan(corpus_vector,0.005,3)
```

```
(9048, 2)
[ 0  1  2 ... 24 26  6]
9048
168
```

```
labels1, clusters1 = dbscan(corpus_vector,0.01,3)
```

```
(9048, 2)
[0 1 0 ... 0 0 0]
9048
35
```

```
labels1, clusters1 = dbscan(encoded_arr_gpt2,0.01,3)
```

```
(9048, 2)
[-1 -1 -1 ... -1 -1 -1]
9048
13
```

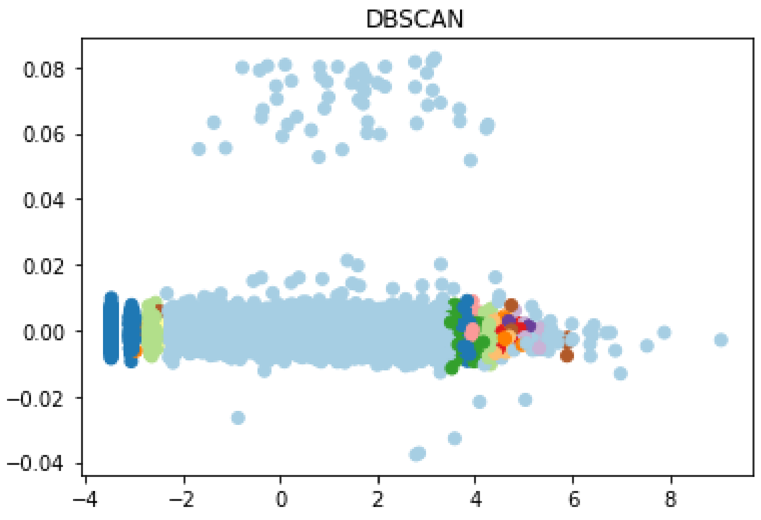
```
labels1, clusters1 = dbscan(encoded_arr_gpt2,0.04,3)
```

```
(9048, 2)
[ -1  -1 293 ...  -1  -1 367]
9048
368
```

```
from sklearn.decomposition import PCA

def plot_dbscan(X , eps, min_samples):
    """Function to plot clusters"""
    pca = PCA(n_components=2)
    result = pca.fit_transform(X)
    print(result.shape)
    db = DBSCAN(eps=eps, min_samples=min_samples)
    db.fit(result)
    y_pred = db.fit_predict(result)
    plt.scatter(result[:,0], result[:,1],c=y_pred, cmap='Paired')
    plt.title("DBSCAN")
```

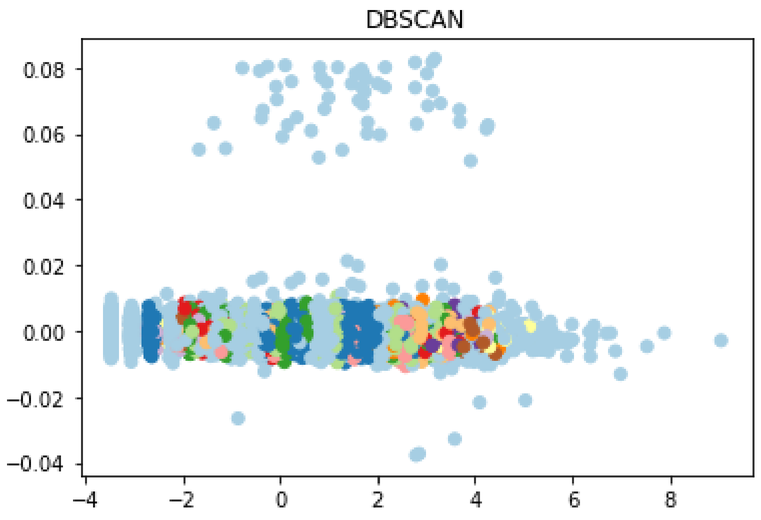
```
plot_dbscan(corpus_vector,0.01,3)
```

```
(9048, 2)
```



```
plot_dbscan(corpus_vector,0.005,3)
```

```
(9048, 2)
```

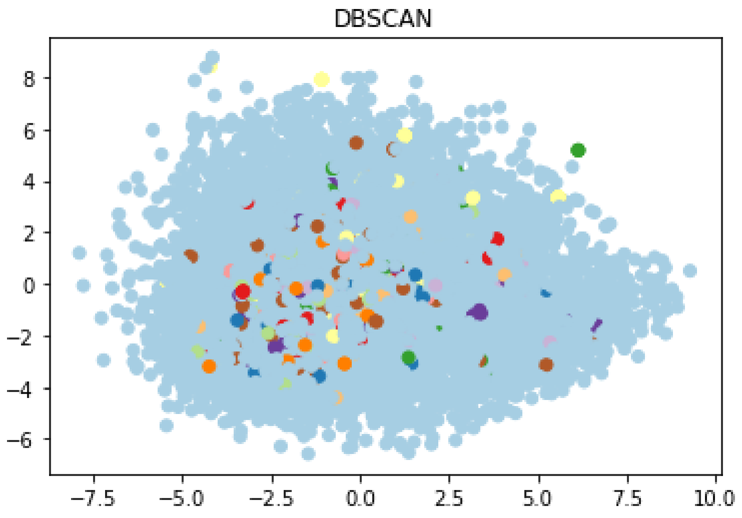


```
plot_dbscan(encoded_arr_gpt2,0.01,3)
```

```
    (9048, 2)
```

```
plot_dbscan(encoded_arr_gpt2,0.04,3)
```

```
    (9048, 2)
```



```
labels1 = labels1.tolist()
```

```
total = 0
for each in range(-1, 150):
  print(each, ":", labels1.count(each))
  total += labels1.count(each)

print(total)
```

```
    -1 : 177
    0 : 301
    1 : 33
    2 : 1188
    3 : 5252
    4 : 420
    5 : 472
    6 : 48
    7 : 371
    8 : 281
    9 : 5
    10 : 73
    11 : 44
    12 : 96
    13 : 3
    14 : 6
    15 : 36
    16 : 9
    17 : 10
    18 : 14
    19 : 5
    20 : 12
    21 : 3
    22 : 4
    23 : 15
    24 : 5
    25 : 8
```

```
26 : 23
27 : 5
28 : 4
29 : 11
30 : 18
31 : 7
32 : 7
33 : 7
34 : 7
35 : 5
36 : 12
37 : 3
38 : 4
39 : 4
40 : 8
41 : 4
42 : 3
43 : 3
44 : 3
45 : 3
46 : 0
47 : 0
48 : 0
49 : 0
50 : 0
51 : 0
52 : 0
53 : 0
54 : 0
55 : 0
56 : 0
57 : 0
```

```
[i for i, e in enumerate(labels1) if e == 3]
```

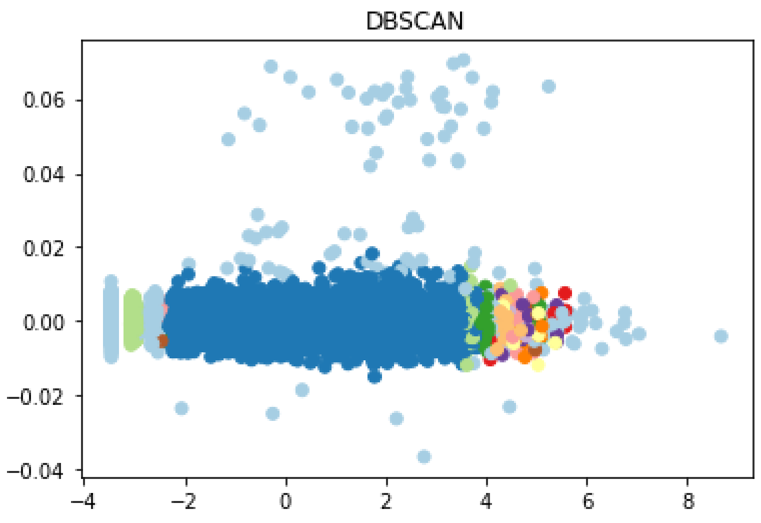
```
[3,
 5,
 7,
 10,
 11,
 13,
 14,
 15,
 23,
 24,
 26,
 27,
 29,
 30,
 31,
 32,
 34,
 36,
 37,
 39,
 40,
 41,
 42,
 45,
 48,
 50,
      52,
      54,
      61,
      63,
      67,
      72,
      73,
      74,
      75,
      76,
      78,
      80,
      84,
      85,
      87,
      88,
      92,
      93,
      97,
      99,
      102,
      103,
      104,
      111,
      112,
      114,
      116,
      117,
      119,
      120,
      127,
      128,
      129,
```

```
print(sentences_list[20])
print()
print(sentences_list[1468])
print()
print(sentences_list[1523])
```

```
abu dhabi uae october ani stumbling defeat hands mumbai indians rajasthan royals wic

new york usa october ani newsvoir support expansion testing contact tracing india ro

new delhi india october ani newsvoir arjun anand author art photographer launched bo
```

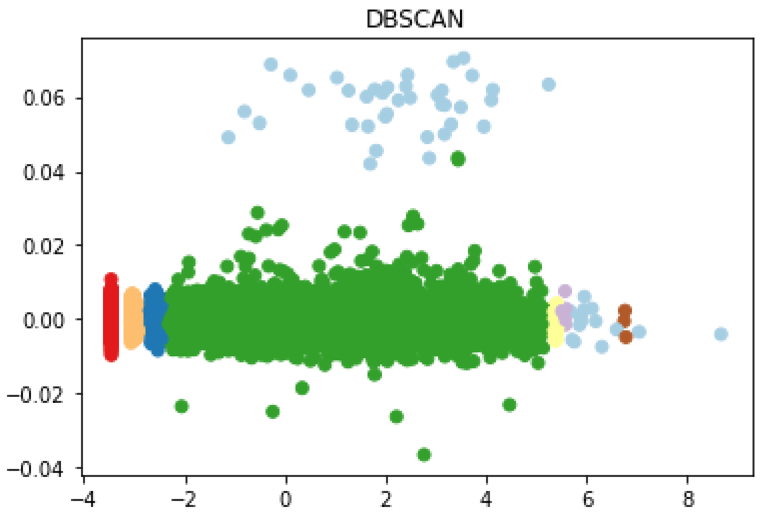
```
from sklearn.decomposition import PCA

def plot_dbscan(X , eps, min_samples):
    """Function to plot clusters"""
    pca = PCA(n_components=2)
    result = pca.fit_transform(X)
    print(result.shape)
    db = DBSCAN(eps=eps, min_samples=min_samples)
    db.fit(result)
    y_pred = db.fit_predict(result)
    plt.scatter(result[:,0], result[:,1],c=y_pred, cmap='Paired')
    plt.title("DBSCAN")

plot_dbscan(corpus_vector,0.01,3)
```

```
(9032, 2)
```



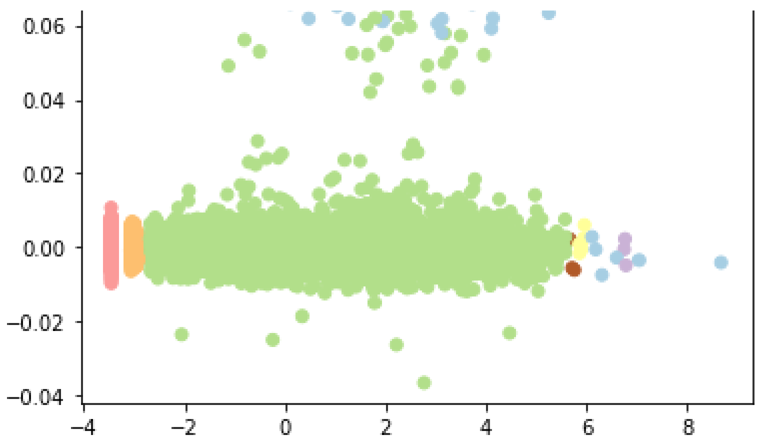
```
plot_dbscan(corpus_vector,0.03, 3)
```

```
(9032, 2)
```



```
plot_dbscan(corpus_vector,0.05, 3)
```

```
(9032, 2)
```



0.06
0.04
0.02
0.00
−0.02
−0.04
−4
−2
0
2
4
6
8