Project Report 12

PLSA: Text Document Clustering

NAME: DHIVAKAR.R

COURSE: AI and ML

Question:

Perform topic modelling using the 20 Newsgroup dataset (the dataset is also available in sklearn datasets sub-module). Perform the required data cleaning steps using NLP and then model the topics

1. Using Latent Dirichlet Allocation (LDA).

2. Using Probabilistic Latent Semantic Analysis (PLSA)

Prerequisites

What things you need to install the software and how to install them:

Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-asan-internal-or-externalcommand/ . Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

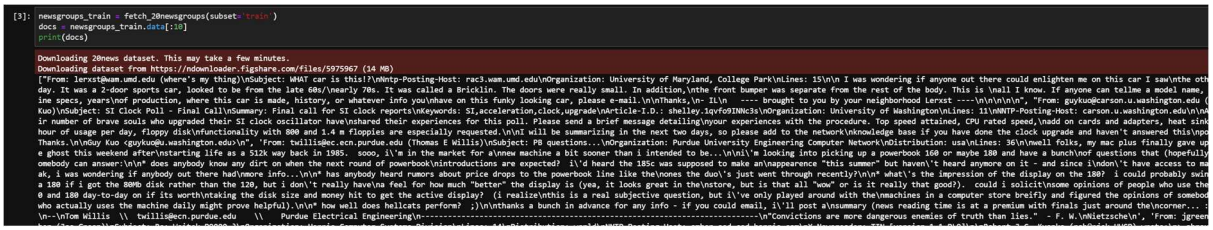
Second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6

Dataset Link: https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html

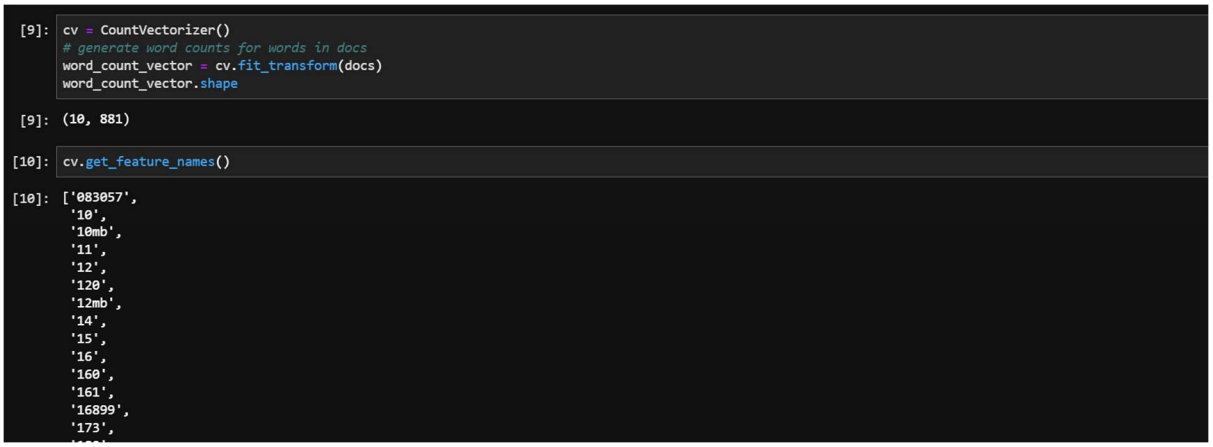
Load all required libraries and Dateset

```
[1]: ! pip install nltk

Requirement already satisfied: nltk in c:\users\dhiva\anaconda3\lib\site-packages (3.4.5)
Requirement already satisfied: six in c:\users\dhiva\anaconda3\lib\site-packages (from nltk) (1.15.0)

[2]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     from nltk.corpus import stopwords
     from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
     from sklearn.decomposition import NMF
     from sklearn.preprocessing import normalize
     from sklearn.datasets import fetch_20newsgroups
```

Downloading dataset



generate word counts for words in docs



get tfidf vector for first document

[15]:
```
import nltk
nltk.download("stopwords")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\dhiva\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

[15]:
```
True
```

[16]:
```
stopw = stopwords.words("english")
def stopwords_remove(x):
    terms = x.split(' ')
    terms = [w for w in terms if w not in stopw]
    sentence = ' '.join(terms)
    return sentence

data_text["Refined_headlines"] = data_text["headline_text"].apply(lambda x: stopwords_remove(x))
```

[17]:
```
data_text.head()
#stopw
```

[17]:

	headline_text	Refined_headlines
1	act fire witnesses must be aware of defamation	act fire witnesses must aware defamation
2	a g calls for infrastructure protection summit	g calls infrastructure protection summit
3	air nz staff in aust strike for pay rise	air nz staff aust strike pay rise
4	air nz strike to affect australian travellers	air nz strike affect australian travellers
5	ambitious olsson wins triple jump	ambitious olsson wins triple jump

[18]:
```
def word_count(x):
    terms = x.split()
    return len(terms)
data_text["word_count"] = data_text["Refined_headlines"].apply(lambda x: word_count(x))
```

[19]:
```
data_text.head()
```

[19]:

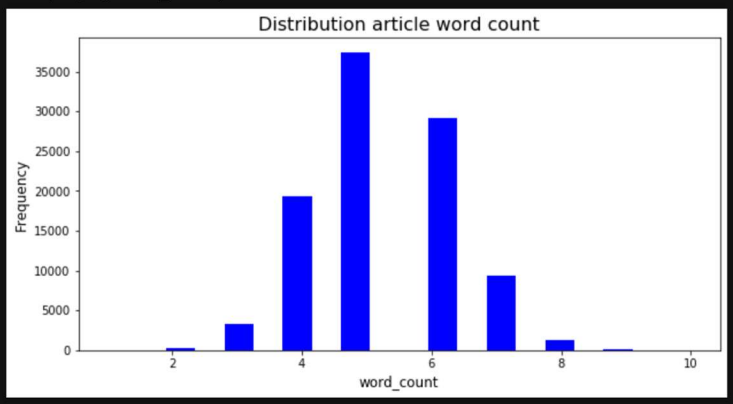
	headline_text	Refined_headlines	word_count
1	act fire witnesses must be aware of defamation	act fire witnesses must aware defamation	6
2	a g calls for infrastructure protection summit	g calls infrastructure protection summit	5
3	air nz staff in aust strike for pay rise	air nz staff aust strike pay rise	7
4	air nz strike to affect australian travellers	air nz strike affect australian travellers	6
5	ambitious olsson wins triple jump	ambitious olsson wins triple jump	5

[20]:
```
data_text["word_count"].describe()
```

[20]:
```
count    100000.000000
mean          5.251110
std           1.035618
min           1.000000
25%           5.000000
50%           5.000000
75%           6.000000
max          10.000000
Name: word_count, dtype: float64
```

[21]:
```
fig = plt.figure(figsize = (10, 5))
plt.hist(data_text["word_count"], bins = 20, color = "blue")
plt.title("Distribution article word count", fontsize = 16)
plt.ylabel("Frequency", fontsize = 12)
plt.xlabel("word_count", fontsize = 12)
```

[21]:
```
Text(0.5, 0, 'word_count')
```



[22]:
```
headline_sentences = [''.join(text) for text in data_text["Refined_headlines"]]

vectorizer = CountVectorizer(max_features = 5000)
x_counts = vectorizer.fit_transform(headline_sentences)

transformer = TfidfTransformer()
x_tfidf = transformer.fit_transform(x_counts)
x_tfidf_norm = normalize(x_tfidf, norm = 'l1', axis = 1)
```

[23]:
```
num_topics = 5

model = NMF(n_components = num_topics, init = "nndsvd")
model.fit(x_tfidf_norm)

def get_nmf_topics(model, n_top_words):
    feat_names = vectorizer.get_feature_names()

    word_dict = {}
    for i in range(num_topics):
        words_ids = model.components_[i].argsort()[:-n_top_words-1:-1]
        words = [feat_names[key] for key in words_ids]

        word_dict['Topic #' + '{:02d}'.format(i+1)] = words
    return pd.DataFrame(word_dict)
```

[24]:
```
get_nmf_topics(model, 30)
```

[24]:

	Topic #01	Topic #02	Topic #03	Topic #04	Topic #05
0	police	us	man	council	new
1	probe	iraq	charged	govt	zealand
2	investigate	killed	court	plan	laws
3	missing	troops	dies	water	president
4	search	iraqi	murder	urged	hospital
5	death	war	face	nsw	chief
6	shooting	says	car	says	set
7	fatal	baghdad	crash	call	record
8	car	soldiers	killed	boost	york
9	crash	soldier	hospital	wa	home
10	hunt	attack	jailed	qld	deal
11	seek	two	charges	funds	lead