Assignment 11

Write a Python code using NLP to Pre-Process the text data and convert Text-Numeric vectors.

I. Use Tokenization, Stopword removal, Stemming/Lemmatization , text preprocess logic using NLTK

II. Use SKLearn for converting Text-Numeric vectors using TF-IDF model

```
    consider novel.txt as text document for implementing question 1.
```

In [141]:
```
# Reading the text file and looking at the data first
filename = 'novel.txt'
file = open(filename, 'rt')
text = file.read()
print(text)
file.close()
```

```
/:;<=@
One morning, when Gregor Samsa woke from troubled dreams, he found
himself transformed in his bed into a horrible vermin.  He lay on
his armour-like back, and if he lifted his head a little he could
see his brown belly, slightly domed and divided by arches into stiff
sections.  The bedding was hardly able to cover it and seemed ready
to slide off any moment.  His many legs, pitifully thin compared
with the size of the rest of him, waved about helplessly as he
looked.

"What's happened to me?" he thought.  It wasn't a dream.  His room,
a proper human room although a little too small, lay peacefully
between its four familiar walls.  A collection of textile samples
lay spread out on the table - Samsa was a travelling salesman - and
above it there hung a picture that he had recently cut out of an
illustrated magazine and housed in a nice, gilded frame.  It showed
a lady fitted out with a fur hat and fur boa who sat upright,
raising a heavy fur muff that covered the whole of her lower arm
towards the viewer.
```

I. Use Tokenization, Stopword removal, Stemming/Lemmatization , text preprocess logic using NLTK

In [142]:
```
#import important libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os

#EDA and preprocessing
import re
import nltk.corpus
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from string import digits

#modeling
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import NMF
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics
import itertools
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split

import warnings
warnings.filterwarnings('ignore')
```

Reading the text file into data frame, so that we can perform preprocessing easily. Using fixed width file method to load text file into data frame

In [143]:
```
#Reading text file to data frame and giving column name as text

data = pd.read_fwf('novel.txt',names=['text'])
```

In [144]:
```
data.head(5)
```

Out[144]:

	text
0	/:;<=@
1	One morning, when Gregor Samsa woke from troub...
2	himself transformed in his bed into a horrible...
3	his armour-like back, and if he lifted his hea...
4	see his brown belly, slightly domed and divide...

In [145]:
```
data.tail(5)
```

Out[145]:

	text
2140	Most people start at our Web site which has th...
2141	This Web site includes information about Proje...
2142	including how to make donations to the Project...
2143	Archive Foundation, how to help produce our ne...
2144	subscribe to our email newsletter to hear abou...

In [146]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2145 entries, 0 to 2144
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    2145 non-null   object
dtypes: object(1)
memory usage: 16.9+ KB
```

In [147]:
```
data['text'].value_counts().sum()
```

Out[147]: 2145

Removing all punctuations

In [148]:
```
data['without_punct'] = data['text'].apply(lambda row: re.sub(r'[^\w\s]+', '', row))
```

In [149]:
```
data['without_punct']
```

Out[149]:
```
0
1       One morning when Gregor Samsa woke from troubl...
2       himself transformed in his bed into a horrible...
3       his armourlike back and if he lifted his head ...
4       see his brown belly slightly domed and divided...
                              ...
2140    Most people start at our Web site which has th...
2141    This Web site includes information about Proje...
2142    including how to make donations to the Project...
2143    Archive Foundation how to help produce our new...
2144    subscribe to our email newsletter to hear abou...
Name: without_punct, Length: 2145, dtype: object
```

Removing numbers

In [150]:
```
data['without_punct_num'] = data['without_punct'].apply(lambda row: re.sub(r'[0-9]+', '', row))
```

In [151]:
```
data['without_punct_num']
```

Out[151]:
```
0
1       One morning when Gregor Samsa woke from troubl...
2       himself transformed in his bed into a horrible...
3       his armourlike back and if he lifted his head ...
4       see his brown belly slightly domed and divided...
                              ...
2140    Most people start at our Web site which has th...
2141    This Web site includes information about Proje...
2142    including how to make donations to the Project...
2143    Archive Foundation how to help produce our new...
2144    subscribe to our email newsletter to hear abou...
Name: without_punct_num, Length: 2145, dtype: object
```

Removing stopwords

In [152]:
```
stop_words = stopwords.words('english')
data['without_stopwords'] = data['without_punct_num'].apply(lambda x: ' '.join([word for word in x.split
data['without_stopwords']
```

Out[152]:
```
0
1       One morning Gregor Samsa woke troubled dreams ...
2                  transformed bed horrible vermin He lay
3                armourlike back lifted head little could
4       see brown belly slightly domed divided arches ...
                              ...
2140      Most people start Web site main PG search facil
2141    This Web site includes information Project Gut...
2142    including make donations Project Gutenberg Lit...
2143           Archive Foundation help produce new eBooks
2144           subscribe email newsletter hear new eBooks
Name: without_stopwords, Length: 2145, dtype: object
```

Removing Extra Spaces between words

In [153]:
```
data['Final_Text'] = data['without_stopwords'].apply(lambda x: re.sub('\s+', ' ', x))
data['Final_Text']
```

Out[153]:
```
0
1       One morning Gregor Samsa woke troubled dreams ...
2                  transformed bed horrible vermin He lay
3                armourlike back lifted head little could
4       see brown belly slightly domed divided arches ...
                              ...
2140      Most people start Web site main PG search facil
2141    This Web site includes information Project Gut...
2142    including make donations Project Gutenberg Lit...
2143           Archive Foundation help produce new eBooks
2144           subscribe email newsletter hear new eBooks
Name: Final_Text, Length: 2145, dtype: object
```

Word Tokenizer

In [154]:
```
data['tokenized'] = data.apply(lambda row: nltk.word_tokenize(row['Final_Text']), axis=1)
data['tokenized']
```

Out[154]:
```
0                                                      []
1       [One, morning, Gregor, Samsa, woke, troubled, ...
2           [transformed, bed, horrible, vermin, He, lay]
3         [armourlike, back, lifted, head, little, could]
4       [see, brown, belly, slightly, domed, divided, ...
                              ...
2140    [Most, people, start, Web, site, main, PG, sea...
2141    [This, Web, site, includes, information, Proje...
2142    [including, make, donations, Project, Gutenber...
2143    [Archive, Foundation, help, produce, new, eBooks]
2144    [subscribe, email, newsletter, hear, new, eBooks]
Name: tokenized, Length: 2145, dtype: object
```

Stemming

In [155]:
```
from nltk.stem import PorterStemmer

ps = PorterStemmer()

def stemmatizer(text):
    stem = [ps.stem(word.lower()) for word in text]
    return stem

data['stemmatized'] = data['tokenized'].apply(lambda string: stemmatizer(string))

data['stemmatized']
```

Out[155]:
```
0                                                      []
1       [one, morn, gregor, samsa, woke, troubl, dream...
2              [transform, bed, horribl, vermin, he, lay]
3             [armourlik, back, lift, head, littl, could]
4       [see, brown, belli, slightli, dome, divid, arc...
                              ...
2140    [most, peopl, start, web, site, main, pg, sear...
2141    [thi, web, site, includ, inform, project, gute...
2142    [includ, make, donat, project, gutenberg, lite...
2143          [archiv, foundat, help, produc, new, ebook]
2144        [subscrib, email, newslett, hear, new, ebook]
Name: stemmatized, Length: 2145, dtype: object
```

Lemmatization

In [156]:
```
wordnet_lemmatizer = WordNetLemmatizer()

def lemmatizer(text):
    lem = [wordnet_lemmatizer.lemmatize(word.lower()) for word in text]
    return lem

data['lemmatized'] = data['tokenized'].apply(lambda string: lemmatizer(string))

data['lemmatized']
```

Out[156]:
```
0                                                      []
1       [one, morning, gregor, samsa, woke, troubled, ...
2           [transformed, bed, horrible, vermin, he, lay]
3         [armourlike, back, lifted, head, little, could]
4       [see, brown, belly, slightly, domed, divided, ...
                              ...
2140    [most, people, start, web, site, main, pg, sea...
2141    [this, web, site, includes, information, proje...
2142    [including, make, donation, project, gutenberg...
2143    [archive, foundation, help, produce, new, ebooks]
2144    [subscribe, email, newsletter, hear, new, ebooks]
Name: lemmatized, Length: 2145, dtype: object
```

II. Use SKLearn for converting Text-Numeric vectors using TF-IDF model

In [157]:
```
import sklearn
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.pipeline import Pipeline

corpus = data['Final_Text'].tolist()
corpus
```

```
 'One morning Gregor Samsa woke troubled dreams found',
 'transformed bed horrible vermin He lay',
 'armourlike back lifted head little could',
 'see brown belly slightly domed divided arches stiff',
 'sections The bedding hardly able cover seemed ready',
 'slide moment His many legs pitifully thin compared',
 'size rest waved helplessly',
 'looked',
 'Whats happened thought It wasnt dream His room',
 'proper human room although little small lay peacefully',
 'four familiar walls A collection textile samples',
 'lay spread table Samsa travelling salesman',
 'hung picture recently cut',
 'illustrated magazine housed nice gilded frame It showed',
 'lady fitted fur hat fur boa sat upright',
 'raising heavy fur muff covered whole lower arm',
 'towards viewer',
 'Gregor turned look window dull weather',
 'Drops rain could heard hitting pane made feel',
```

In [158]:
```
vectorizer = CountVectorizer(stop_words = 'english')

dtm = vectorizer.fit_transform(corpus)

pd.DataFrame(dtm.toarray(),index=corpus,columns=vectorizer.get_feature_names()).head(5)
```

Out[158]:

	abandoned	abandoning	abide	ability	able	abruptly	absolutely	accept	acceptance	accepted	...	yearning	years	yes	yest
	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
One morning Gregor Samsa woke troubled dreams found	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
transformed bed horrible vermin He lay	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
armourlike back lifted head little could	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
see brown belly slightly domed divided arches stiff	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

5 rows × 2818 columns

In [159]:
```
pipe = Pipeline([('count', CountVectorizer()),('tfid', TfidfTransformer())]).fit(corpus)

pipe['count'].transform(corpus).toarray()
```

Out[159]:
```
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 [160]:
```
pipe['tfid'].idf_
```

Out[160]:
```
array([7.57274863, 7.57274863, 7.97821374, ..., 6.72545077, 7.57274863,
       7.97821374])
```

In [161]:
```
pipe.transform(corpus).shape
```

Out[161]:
```
(2145, 3025)
```

In [162]:
```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(corpus)
X
```

Out[162]:
```
<2145x3025 sparse matrix of type '<class 'numpy.float64'>'
        with 12317 stored elements in Compressed Sparse Row format>
```

In [163]:
```
vectorizer.get_feature_names_out()
```

Out[163]:
```
array(['abandoned', 'abandoning', 'abide', ..., 'youre', 'youve', 'zip'],
      dtype=object)
```

In [164]:
```
print(X.shape)
```

```
(2145, 3025)
```