Problem #2A: The goal is to identify the best number of clusters that responses to the following question organize into using NLP methods:

"What one action can faculty take to improve your educational experience at UW?"

No assumptions are made about how many clusters (groups) these responses will fall into. The goal of this portion of the NLP project is to identify the optimal number of clusters to support future coding of these responses.

This will be accomplished by representing the students' responses into three categories:: part A: topic, part B: sentiment analysis, and part B: semantic similarity.

This code solves part A: Topic Modelling

In [68]:
```
from numpy import array
from keras.preprocessing.text import one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers.core import Activation, Dropout, Dense
from keras.layers import Flatten, LSTM
from keras.layers import GlobalMaxPooling1D
from keras.models import Model
from keras.layers.embeddings import Embedding
from sklearn.model_selection import train_test_split
from keras.preprocessing.text import Tokenizer
from keras.layers import Input
from keras.layers.merge import Concatenate

import pandas as pd
import numpy as np
import re

import matplotlib.pyplot as plt
```

In [100…
```
df = pd.read_csv('/Users/nehakardam/Documents/UWclasses /EE517 NLP/Project/Facul
```

In [101…
```
df.shape
```

Out[101… (1624, 60)

In [102…
```
df.head()
```

Out[102…

	Join Code	RemoteTrad	Subject Code	Class	Quarter	Year	Section	A1_Status
0	48.0	2.0	EE233_SP20_AC_48	EE233_Spring2020	Spring	2020.0	AC	2
1	49.0	2.0	EE233_SP20_AA_49	EE233_Spring2020	Spring	2020.0	AA	4
2	63.0	2.0	EE235_SP20_AD_63	EE235_Spring2020	Spring	2020.0	AD	2

	Join Code	RemoteTrad	Subject Code	Class	Quarter	Year	Section	A1_Status
3	11.0	2.0	EE331_SP20_AA_11	EE331_Spring2020	Spring	2020.0	AA	3
4	3.0	2.0	EE233_SP20_AB_3	EE233_Spring2020	Spring	2020.0	AB	2

5 rows × 60 columns

In [103…
```
df["SA1"][350]
```

Out[103…
```
'0e breakout rooms in lectures to allow students to still have ineraction with o
ne another.'
```

In [104…
```
from sklearn.feature_extraction.text import CountVectorizer

count_vect = CountVectorizer(max_df=0.8, min_df=2, stop_words='english')
doc_term_matrix = count_vect.fit_transform(df['SA1'].values.astype('U'))
```

In [105…
```
doc_term_matrix
```

Out[105…
```
<1624x1138 sparse matrix of type '<class 'numpy.int64'>'
        with 12089 stored elements in Compressed Sparse Row format>
```

Method 1: LDA for Topic Modeling

LDA is used to create topics along with the probability distribution for each word in our vocabulary for each topic.

In [106…
```
from sklearn.decomposition import LatentDirichletAllocation

LDA = LatentDirichletAllocation(n_components=5, random_state=42)
LDA.fit(doc_term_matrix)
```

Out[106…
```
LatentDirichletAllocation(n_components=5, random_state=42)
```

In [107…
```
first_topic = LDA.components_[0]
```

In [108…
```
top_topic_words = first_topic.argsort()[-10:]
```

In [109…
```
for i in top_topic_words:
    print(count_vect.get_feature_names()[i])
```

```
classes
assignments
online
questions
work
exam
professors
time
class
students
```

In [110…
```
for i,topic in enumerate(LDA.components_):
    print(f'Top 10 words for topic #{i}:')
```

```
    print([count_vect.get_feature_names()[i] for i in topic.argsort()[-10:]])
    print('\n')
```

```
Top 10 words for topic #0:
['classes', 'assignments', 'online', 'questions', 'work', 'exam', 'professors',
'time', 'class', 'students']


Top 10 words for topic #1:
['giving', 'practice', 'real', 'solve', 'assignments', 'example', 'problems', 'p
rofessors', 'exams', 'nan']


Top 10 words for topic #2:
['online', 'provide', 'recordings', 'professor', 'post', 'lectures', 'class', 's
lides', 'notes', 'lecture']


Top 10 words for topic #3:
['help', 'having', 'problems', 'homework', 'online', 'class', 'time', 'question
s', 'office', 'hours']


Top 10 words for topic #4:
['students', 'make', 'provide', 'problems', 'material', 'practice', 'professor
s', 'examples', 'class', 'lectures']
```

In [111…
```
topic_values = LDA.transform(doc_term_matrix)
topic_values.shape
```

Out[111…
```
(1624, 5)
```

In [112…
```
df['Topic'] = topic_values.argmax(axis=1)
```

In [113…
```
df.head()
```

Out[113…

	Join Code	RemoteTrad	Subject Code	Class	Quarter	Year	Section	A1_Status
0	48.0	2.0	EE233_SP20_AC_48	EE233_Spring2020	Spring	2020.0	AC	2
1	49.0	2.0	EE233_SP20_AA_49	EE233_Spring2020	Spring	2020.0	AA	4
2	63.0	2.0	EE235_SP20_AD_63	EE235_Spring2020	Spring	2020.0	AD	2
3	11.0	2.0	EE331_SP20_AA_11	EE331_Spring2020	Spring	2020.0	AA	3
4	3.0	2.0	EE233_SP20_AB_3	EE233_Spring2020	Spring	2020.0	AB	2

5 rows × 61 columns

In [114…
```
df.to_csv('/Users/nehakardam/Documents/UWclasses /EE517 NLP/Project/FS_Topic_LDA
```

Method 2: Non-Negative Matrix Factorization (NMF)

Non-negative matrix factorization is also a supervised learning technique which performs clustering as well as dimensionality reduction. It can be used in combination with TF-IDF scheme to perform topic modeling. In this section, we will see how Python can be used to perform non-negative matrix factorization for topic modeling.

In [115…
```
df = pd.read_csv('/Users/nehakardam/Documents/UWclasses /EE517 NLP/Project/Facul
df.head()
```

Out[115…

	Join Code	RemoteTrad	Subject Code	Class	Quarter	Year	Section	A1_Status
0	48.0	2.0	EE233_SP20_AC_48	EE233_Spring2020	Spring	2020.0	AC	2
1	49.0	2.0	EE233_SP20_AA_49	EE233_Spring2020	Spring	2020.0	AA	4
2	63.0	2.0	EE235_SP20_AD_63	EE235_Spring2020	Spring	2020.0	AD	2
3	11.0	2.0	EE331_SP20_AA_11	EE331_Spring2020	Spring	2020.0	AA	3
4	3.0	2.0	EE233_SP20_AB_3	EE233_Spring2020	Spring	2020.0	AB	2

5 rows × 60 columns

In [116…
```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vect = TfidfVectorizer(max_df=0.8, min_df=2, stop_words='english')
doc_term_matrix1 = tfidf_vect.fit_transform(df['SA1'].values.astype('U'))
```

In [123…
```
from sklearn.decomposition import NMF

nmf = NMF(n_components=5, init='random', random_state=0)
nmf.fit(doc_term_matrix1 )
```

Out[123… `NMF(init='random', n_components=5, random_state=0)`

In [124…
```
import random

for i in range(10):
    random_id = random.randint(0,len(tfidf_vect.get_feature_names()))
    print(tfidf_vect.get_feature_names()[random_id])
```

```
web
early
reflect
understands
220
loud
single
discuss
accountable
similar
```

In [125…
```
first_topic = nmf.components_[0]
top_topic_words = first_topic.argsort()[-10:]
```

In [126…
```
for i in top_topic_words:
    print(tfidf_vect.get_feature_names()[i])
```

```
helps
learn
sessions
cover
provide
exam
understand
review
material
nan
```

In [127…
```
for i,topic in enumerate(nmf.components_):
    print(f'Top 10 words for topic #{i}:')
    print([tfidf_vect.get_feature_names()[i] for i in topic.argsort()[-10:]])
    print('\n')
```

```
Top 10 words for topic #0:
['helps', 'learn', 'sessions', 'cover', 'provide', 'exam', 'understand', 'revie
w', 'material', 'nan']


Top 10 words for topic #1:
['material', 'exam', 'extra', 'tests', 'homework', 'examples', 'exams', 'provid
e', 'problems', 'practice']


Top 10 words for topic #2:
['review', 'clear', 'online', 'canvas', 'provide', 'recordings', 'slides', 'pos
t', 'notes', 'lecture']


Top 10 words for topic #3:
['sessions', 'online', 'lots', 'extra', 'holding', 'help', 'hold', 'available',
'office', 'hours']


Top 10 words for topic #4:
['online', 'ask', 'helpful', 'make', 'professors', 'time', 'questions', 'lecture
s', 'students', 'class']
```

In [128…
```
topic_values1 = nmf.transform(doc_term_matrix1)
df['Topic1'] = topic_values1.argmax(axis=1)
df.head()
```

Out[128…

	Join Code	RemoteTrad	Subject Code	Class	Quarter	Year	Section	A1_Status
0	48.0	2.0	EE233_SP20_AC_48	EE233_Spring2020	Spring	2020.0	AC	2
1	49.0	2.0	EE233_SP20_AA_49	EE233_Spring2020	Spring	2020.0	AA	4
2	63.0	2.0	EE235_SP20_AD_63	EE235_Spring2020	Spring	2020.0	AD	2
3	11.0	2.0	EE331_SP20_AA_11	EE331_Spring2020	Spring	2020.0	AA	3
4	3.0	2.0	EE233_SP20_AB_3	EE233_Spring2020	Spring	2020.0	AB	2

5 rows × 61 columns

In [129…
```
df.to_csv('/Users/nehakardam/Documents/UWclasses /EE517 NLP/Project/FS_Topic_NMF
```

Reference:https://stackabuse.com/python-for-nlp-topic-modeling/

In [ ]: