

# LDA Tuning Approach 3

Document = part between interludes

## Imports

```
In [1]: 1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4 from wordcloud import WordCloud, STOPWORDS
5 from gensim.utils import simple_preprocess, SaveLoad
6 #nltk.download('stopwords')
7 #nltk.download('wordnet')
8 from nltk.corpus import stopwords
9 from nltk.stem import WordNetLemmatizer, SnowballStemmer
10 from nltk.stem.porter import *
11 import time
12 from pickle import load, dump
13 import my_lda_utils
14 from my_lda_utils import my_lda_model
15 from gensim.models import CoherenceModel
16 import pyLDAvis
```

## Initializations

```
In [2]: 1 np.random.seed(2018)
2 stemmer = SnowballStemmer('english')
3 pyLDAvis.enable_notebook()
```

```
In [3]: 1 stop_words = stopwords.words('english')
2 stop_words.extend(STOPWORDS)
3 stop_words2 = stop_words
```

## Helpers

```
In [4]: 1 # Cleaning functions
2 def remove_stopwords(texts):
3     return [word for word in texts if word not in stop_words ]
4
5 def stem_and_lemmatize(texts):
6     return [stemmer.stem(WordNetLemmatizer().lemmatize(word, pos='v')) for word in texts] #try to lematize first: source https://towardsdatascience.com/tf-idf-for-document-ranking-from-s
7
8 #     return [word for word in simple_preprocess(str(doc))
9 #         if word not in stop_words] for doc in texts]
```

## Import Data

```
In [5]: 1 clean_data_folder = "../clean_data/"
2 with open("../clean_data/"+characters_with_gender.pkl", "rb") as f2:
3     characters = load(f2)
```

```
In [6]: 1 clean_data_folder_lda = "../clean_data_lda"
2 with open(clean_data_folder_lda+"/all_data.pkl", "rb") as f:
3     all_scripts = load(f)
4 data_lda = pd.merge(all_scripts, characters, on="character", how="inner")
```

```
In [7]: 1 all_scripts.reset_index(inplace=True, drop=True)
```

```
In [8]: 1 data = pd.merge(all_scripts, characters, on="character", how="inner")
```

```
In [9]: 1 data2 = data_lda[['text', 'doc_id']]
2 data3 = data2.groupby(['doc_id'])['text'].agg(sum).reset_index()
3 documents = data3.text.apply(stem_and_lemmatize)
```

## Tuning

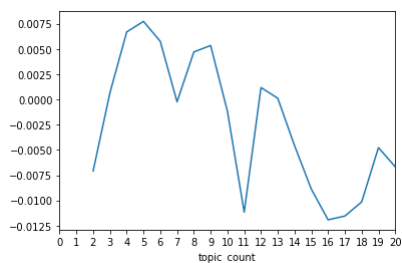
### Number of Topics

```
In [10]: 1 coherence_scores_nmpi = []
```

```
In [11]: 1 for i in range(2,21,1):
2     lda_model_x = my_lda_model(documents)
3     lda_model_x.train(num_topics = i, passes = 5, tfidf_threshold = 0.0)
4     coherence_model_lda = CoherenceModel(model=lda_model_x.lda_model, texts=lda_model_x.documents, dictionary=lda_model_x.dictionary, coherence='c_npmi')
5     coherence_lda = coherence_model_lda.get_coherence()
6     coherence_scores_nmpi.append([i, coherence_lda,lda_model_x])
7     print("coherence: ",[i, coherence_lda])
```

```
In [10]: 1 #df = pd.DataFrame(coherence_scores_nmpi)
2 #df.columns = ["topic_count", "coherence", "model"]
3 #df = df.set_index("topic_count")
4 df.coherence.plot(kind='line',
5                  xticks=[x for x in range(21)],
6                  xlim=[0,20])
7 df.coherence.idxmax()
```

Out[10]: 5



```
In [13]: 1 df.coherence[9].idxmax()
```

Out[13]: 12

In [14]: 1 #df.to\_pickle("c\_nmpi\_coherence\_approach3.pkl")

In [7]: 1 with open("c\_nmpi\_coherence\_approach3.pkl", "rb") as f:  
2 coherence\_scores\_topic\_counts = load(f)

In [8]: 1 coherence\_scores\_topic\_counts.head(2)

Out[8]:

	coherence	model
topic_count		
2	-0.007080	<my_lda_utils.my_lda_model object at 0x000001C...
3	0.000657	<my_lda_utils.my_lda_model object at 0x000001C...

In [9]: 1 df = coherence\_scores\_topic\_counts

5, 9, 12 maxima

In [ ]: 1 df.model[4].visual

In [ ]: 1 df.model[7].visual

In [ ]: 1 df.model[12].visual

pick topic count 8 for better comparisoin

TF-IDF - Theshold

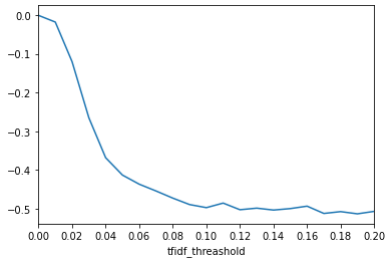
8 Topics

In [17]: 1 coherence\_scores\_tfidf = []

In [31]: 1 tfidf\_range = [round(x\*0.01, 2) for x in range(21)]  
2 for i in tfidf\_range:  
3 lda\_model\_x = my\_lda\_model(documents)  
4 lda\_model\_x.train(num\_topics = 8, passes = 5, tfidf\_threshold = i)  
5 coherence\_model\_lda = CoherenceModel(model=lda\_model\_x.lda\_model, texts=lda\_model\_x.documents, dictionary=lda\_model\_x.dictionary, coherence='c\_npmi')  
6 coherence\_lda = coherence\_model\_lda.get\_coherence()  
7 coherence\_scores\_tfidf.append([i, coherence\_lda,lda\_model\_x])  
8 print("coherence: ",[i, coherence\_lda])

In [40]: 1 df = pd.DataFrame(coherence\_scores\_tfidf)  
2 df.columns = ["tfidf\_threshold", "coherence", "model"]  
3 df = df.set\_index("tfidf\_threshold")  
4 df.coherence.plot(kind='line',  
5 xticks=[i for i in tfidf\_range if round(i\*100)%2 == 0], # get every second tick  
6 xlim=[0,0.2])  
7 df.coherence.idxmax()

Out[40]: 0.0



In [41]: 1 #df.to\_pickle("tfidf\_coherence-topics-8-passes-5\_approach3.pkl")  
2 #todo: compare to approach 2

In [99]: 1 with open("tfidf\_coherence-topics-8-passes-5\_approach3.pkl", "rb") as f:  
2 df = load(f)  
3 df.head(2)

Out[99]:

	coherence	model
tfidf_threashold		
0.00	-0.000705	<my_lda_utils.my_lda_model object at 0x0000028...
0.01	-0.017902	<my_lda_utils.my_lda_model object at 0x0000028...

In [108]: 1 df.model[0.11].visual

Out[108]:

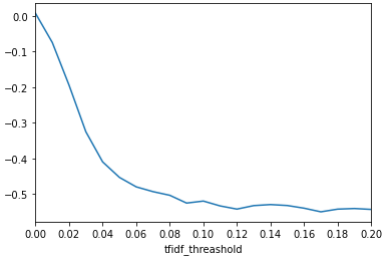
4 Topics

In [46]: 1 coherence\_scores\_tfidf = []

In [47]: 1 tfidf\_range = [round(x\*0.01, 2) for x in range(21)]  
2 for i in tfidf\_range:  
3 lda\_model\_x = my\_lda\_model(documents)  
4 lda\_model\_x.train(num\_topics = 4, passes = 5, tfidf\_threshold = i)  
5 coherence\_model\_lda = CoherenceModel(model=lda\_model\_x.lda\_model, texts=lda\_model\_x.documents, dictionary=lda\_model\_x.dictionary, coherence='c\_npmi')  
6 coherence\_lda = coherence\_model\_lda.get\_coherence()  
7 coherence\_scores\_tfidf.append([i, coherence\_lda,lda\_model\_x])  
8 print("coherence: ",[i, coherence\_lda])

```
In [48]: 1 df = pd.DataFrame(coherence_scores_tfidf)
2 df.columns = ["tfidf_threshold", "coherence", "model"]
3 df = df.set_index("tfidf_threshold")
4 df.coherence.plot(kind='line',
5                  xticks=[i for i in tfidf_range if round(i*100)%2 == 0], # get every second tick
6                  xlim=[0,0.2])
7 df.coherence.idxmax()
```

Out[48]: 0.0



```
In [49]: 1 #df.to_pickle("tfidf coherence-topics-4-passes-5 approach3.pkl")
```

```
In [8]: 1 with open("tfidf_coherence-topics-4-passes-5_approach3.pkl", "rb") as f:
2         df2 = load(f)
3         df2.head(2)
```

Out[8]:

	coherence	model
tfidf_threshold		
0.00	0.007221	<my_lda_utils.my_lda_model object at 0x000001E...
0.01	-0.074569	<my_lda_utils.my_lda_model object at 0x000001E...

```
In [49]: 1 df2.model[0.2].visual
```

Out[49]:

5 Topics

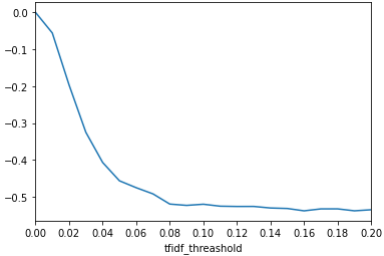
```
In [26]: 1 coherence_scores_tfidf = []
```

```
In [27]: 1 tfidf_range = [round(x*0.01, 2) for x in range(21)]
2 for i in tfidf_range:
3     lda_model_x = my_lda_model(documents)
4     lda_model_x.train(num_topics = 5, passes = 5, tfidf_threshold = i)
5     coherence_model_lda = CoherenceModel(model=lda_model_x.lda_model, texts=lda_model_x.documents, dictionary=lda_model_x.dictionary, coherence='c_npmi')
6     coherence_lda = coherence_model_lda.get_coherence()
7     coherence_scores_tfidf.append([i, coherence_lda,lda_model_x])
8     print("coherence: ",i, coherence_lda)
```

C:\Users\debor\anaconda3\lib\site-packages\pyLDavis\prepare.py:246: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.  
default\_term\_info = default\_term\_info.sort\_values(  
  
coherence: [0.0, -0.0005813933019606164]  
  
C:\Users\debor\anaconda3\lib\site-packages\pyLDavis\prepare.py:246: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.  
default\_term\_info = default\_term\_info.sort\_values(  
  
coherence: [0.01, -0.0562611808835745]  
  
C:\Users\debor\anaconda3\lib\site-packages\pyLDavis\prepare.py:246: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.  
default\_term\_info = default\_term\_info.sort\_values(  
  
coherence: [0.02, -0.19691455879759376]  
  
C:\Users\debor\anaconda3\lib\site-packages\pyLDavis\prepare.py:246: FutureWarning: In a future version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only.  
default\_term\_info = default\_term\_info.sort values(

```
In [28]: 1 df = pd.DataFrame(coherence_scores_tfidf)
2 df.columns = ["tfidf_threshold", "coherence", "model"]
3 df = df.set_index("tfidf_threshold")
4 df.coherence.plot(kind='line',
5                  xticks=[i for i in tfidf_range if round(i*100)%2 == 0], # get every second tick
6                  xlim=[0,0.2])
7 df.coherence.idxmax()
```

Out[28]: 0.0



```
In [29]: 1 #df.to_pickle("tfidf coherence-topics-5-passes-5 approach3.pkl")
```

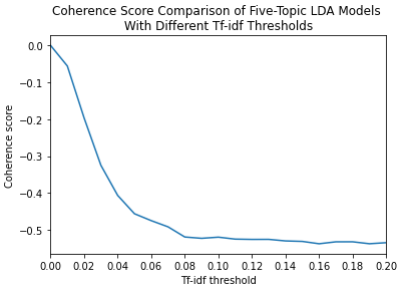
```
In [10]: 1 with open("tfidf_coherence-topics-5-passes-5_approach3.pkl", "rb") as f:
2         df2 = load(f)
3         df2.head(2)
```

Out[10]:

	coherence	model
tfidf_threshold		
0.00	-0.000581	<my_lda_utils.my_lda_model object at 0x0000025...
0.01	-0.056261	<my_lda_utils.my_lda_model object at 0x0000025...

```
In [22]: 1 df2.columns = ["Coherence", "Model"]
2         df2.index.names = ["Tf-idf threshold"]
3         df2.Coherence.plot(kind='line',
4                             xticks=[round(x*0.01, 2) for x in range(0, 21, 2)], # get every second tick
5                             xlim=[0,0.2],
6                             ylabel="Coherence score",
7                             title="Coherence Score Comparison of Five-Topic LDA Models \n With Different Tf-idf Thresholds "
8                             )
```

```
Out[22]: <AxesSubplot:title={'center':'Coherence Score Comparison of Five-Topic LDA Models \n With Different Tf-idf Thresholds '}, xlabel='Tf-idf threshold', ylabel='Coherence score'>
```



```
In [19]: 1
```

```
Out[19]: [0.0, 0.02, 0.04, 0.06, 0.08, 0.1, 0.12, 0.14, 0.16, 0.18, 0.2]
```

```
In [16]: 1 df2
```

Out[16]:

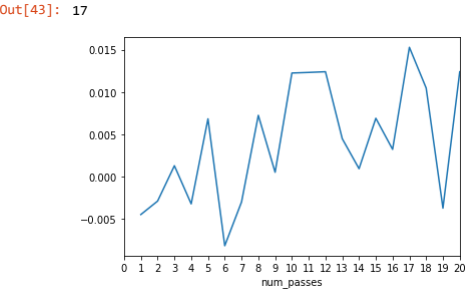
	Coherence	Model
Tf-idf threshold		
0.00	-0.000581	<my_lda_utils.my_lda_model object at 0x0000025...
0.01	-0.056261	<my_lda_utils.my_lda_model object at 0x0000025...
0.02	-0.196915	<my_lda_utils.my_lda_model object at 0x0000025...
0.03	-0.324870	<my_lda_utils.my_lda_model object at 0x0000025...
0.04	-0.406348	<my_lda_utils.my_lda_model object at 0x0000025...
0.05	-0.455895	<my_lda_utils.my_lda_model object at 0x0000025...
0.06	-0.474568	<my_lda_utils.my_lda_model object at 0x0000025...
0.07	-0.491303	<my_lda_utils.my_lda_model object at 0x0000025...
0.08	-0.518814	<my_lda_utils.my_lda_model object at 0x0000025...
0.09	-0.522374	<my_lda_utils.my_lda_model object at 0x0000025...
0.10	-0.519249	<my_lda_utils.my_lda_model object at 0x0000025...
0.11	-0.524537	<my_lda_utils.my_lda_model object at 0x0000025...
0.12	-0.525298	<my_lda_utils.my_lda_model object at 0x0000025...
0.13	-0.525205	<my_lda_utils.my_lda_model object at 0x0000025...
0.14	-0.529430	<my_lda_utils.my_lda_model object at 0x0000025...
0.15	-0.530739	<my_lda_utils.my_lda_model object at 0x0000025...
0.16	-0.537099	<my_lda_utils.my_lda_model object at 0x0000025...
0.17	-0.531854	<my_lda_utils.my_lda_model object at 0x0000025...
0.18	-0.531787	<my_lda_utils.my_lda_model object at 0x0000025...
0.19	-0.537047	<my_lda_utils.my_lda_model object at 0x0000025...
0.20	-0.534179	<my_lda_utils.my_lda_model object at 0x0000025...

Number of Passes

```
In [40]: 1 coherence_scores_no_passes = []
```

```
In [41]: 1 lds_train = df2.model[0.01].lda_model
2         for i in range(1,21,1):
3             lda_model_x = my_lda_model(documents)
4             lda_model_x.train(num_topics = 8, passes = i, tfidf_threshold = 0)#todo
5             coherence_model_lda = CoherenceModel(model=lda_model_x.lda_model, texts=lda_model_x.documents, dictionary=lda_model_x.dictionary, coherence='c_npmi')
6             coherence_lda = coherence_model_lda.get_coherence()
7             coherence_scores_no_passes.append([i, coherence_lda,lda_model_x])
8             print("coherence: ",[i, coherence_lda])
```

```
In [43]: 1 df = pd.DataFrame(coherence_scores_no_passes)
2 df.columns = ["num_passes", "coherence", "model"]
3 df = df.set_index("num_passes")
4 df.coherence.plot(kind='line',
5                   xticks=list(range(21)),
6                   xlim=[0,20])
7 df.coherence.idxmax()
```



```
In [44]: 1 #df.to_pickle("num_passes_coherence_approach3-topic8-tfidf-0.pkl")
```

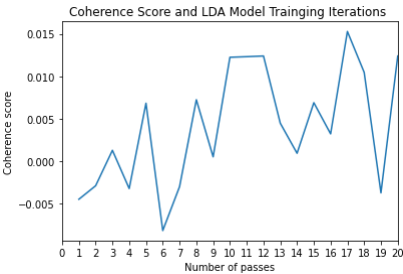
```
In [12]: 1 with open("num_passes_coherence_approach3-topic8-tfidf-0.pkl", "rb") as f:
2         df = load(f)
3         df.head(2)
```

Out[12]:

	coherence	model
num_passes		
1	-0.004455	<my_lda_utils.my_lda_model object at 0x000001B...
2	-0.002866	<my_lda_utils.my_lda_model object at 0x000001B...

```
In [14]: 1 # df = pd.DataFrame(coherence_scores_no_passes)
2 # df.columns = ["num_passes", "coherence", "model"]
3 # df = df.set_index("num_passes")
4 df.coherence.plot(kind='line',
5                   xticks=list(range(21)),
6                   xlim=[0,20],
7                   xlabel="Number of passes",
8                   ylabel="Coherence score",
9                   title="Coherence Score and LDA Model Trainging Iterations ")
```

Out[14]: <AxesSubplot:title={'center':'Coherence Score and LDA Model Trainging Iterations '}, xlabel='Number of passes', ylabel='Coherence score'>



```
In [ ]: 1
```