Demonstration of the topic coherence pipeline in Gensim

Introduction

We will be using the `u_mass` and `c_v` coherence for two different LDA models: a "good" and a "bad" LDA model. The good LDA model will be trained over 50 iteration. Hence in theory, the good LDA model will be able come up with better or more human-understandable topics. Therefore the coherence measure output be more (better) than that for the bad LDA model. This is because, simply, the good LDA model usually comes up with better topics that are more human interpre

In [1]:
```
import numpy as np
import logging
import pyLDAvis.gensim
import json
import warnings
warnings.filterwarnings('ignore')  # To ignore all warnings that arise here to enhance clarity

from gensim.models.coherencemodel import CoherenceModel
from gensim.models.ldamodel import LdaModel
from gensim.corpora.dictionary import Dictionary
from numpy import array
```

Set up logging

In [2]:
```
logger = logging.getLogger()
logger.setLevel(logging.DEBUG)
logging.debug("test")
```

Set up corpus

As stated in table 2 from this (http://www.cs.bham.ac.uk/~pxt/IDA/lsa_ind.pdf) paper, this corpus essentially has two classes of documents. First five are about hu the other four are about graphs. We will be setting up two LDA models. One with 50 iterations of training and the other with just 1. Hence the one with 50 iteratio able to capture this underlying pattern of the corpus better than the "bad" LDA model. Therefore, in theory, our topic coherence for the good LDA model should b bad LDA model.

In [3]:
```
texts = [['human', 'interface', 'computer'],
         ['survey', 'user', 'computer', 'system', 'response', 'time'],
         ['eps', 'user', 'interface', 'system'],
         ['system', 'human', 'system', 'eps'],
         ['user', 'response', 'time'],
         ['trees'],
         ['graph', 'trees'],
         ['graph', 'minors', 'trees'],
         ['graph', 'minors', 'survey']]
```

In [4]:
```
dictionary = Dictionary(texts)
corpus = [dictionary.doc2bow(text) for text in texts]
```

Set up two topic models

We'll be setting up two different LDA Topic models. A good one and bad one. To build a "good" topic model, we'll simply train it using more iterations than the bac coherence should in theory be better for the good model than the bad one since it would be producing more "human-interpretable" topics.

In [5]:
```
goodLdaModel = LdaModel(corpus=corpus, id2word=dictionary, iterations=50, num_topics=2)
badLdaModel = LdaModel(corpus=corpus, id2word=dictionary, iterations=1, num_topics=2)
```

Using U_Mass Coherence

In [14]:
```
goodcm = CoherenceModel(model=goodLdaModel, corpus=corpus, dictionary=dictionary, coherence='u_mass')
```

In [15]:
```
badcm = CoherenceModel(model=badLdaModel, corpus=corpus, dictionary=dictionary, coherence='u_mass')
```

View the pipeline parameters for one coherence model

Following are the pipeline parameters for `u_mass` coherence. By pipeline parameters, we mean the functions being used to calculate segmentation, probability measure and aggregation as shown in figure 1 in this (http://svn.aksw.org/papers/2015/WSDM_Topic_Evaluation/public.pdf) paper.

In [16]:
```
print goodcm
```

```
CoherenceModel(segmentation=<function s_one_pre at 0x7f663ae82f50>, probability estimation=<function p_boolean_document at 0x7f66
```

Interpreting the topics

As we will see below using LDA visualization, the better model comes up with two topics composed of the following words:

1. goodLdaModel:
   - **Topic 1**: More weightage assigned to words such as "system", "user", "eps", "interface" etc which captures the first set of documents.
   - **Topic 2**: More weightage assigned to words such as "graph", "trees", "survey" which captures the topic in the second set of documents.
2. badLdaModel:
   - **Topic 1**: More weightage assigned to words such as "system", "user", "trees", "graph" which doesn't make the topic clear enough.
   - **Topic 2**: More weightage assigned to words such as "system", "trees", "graph", "user" which is similar to the first topic. Hence both topics are not humar

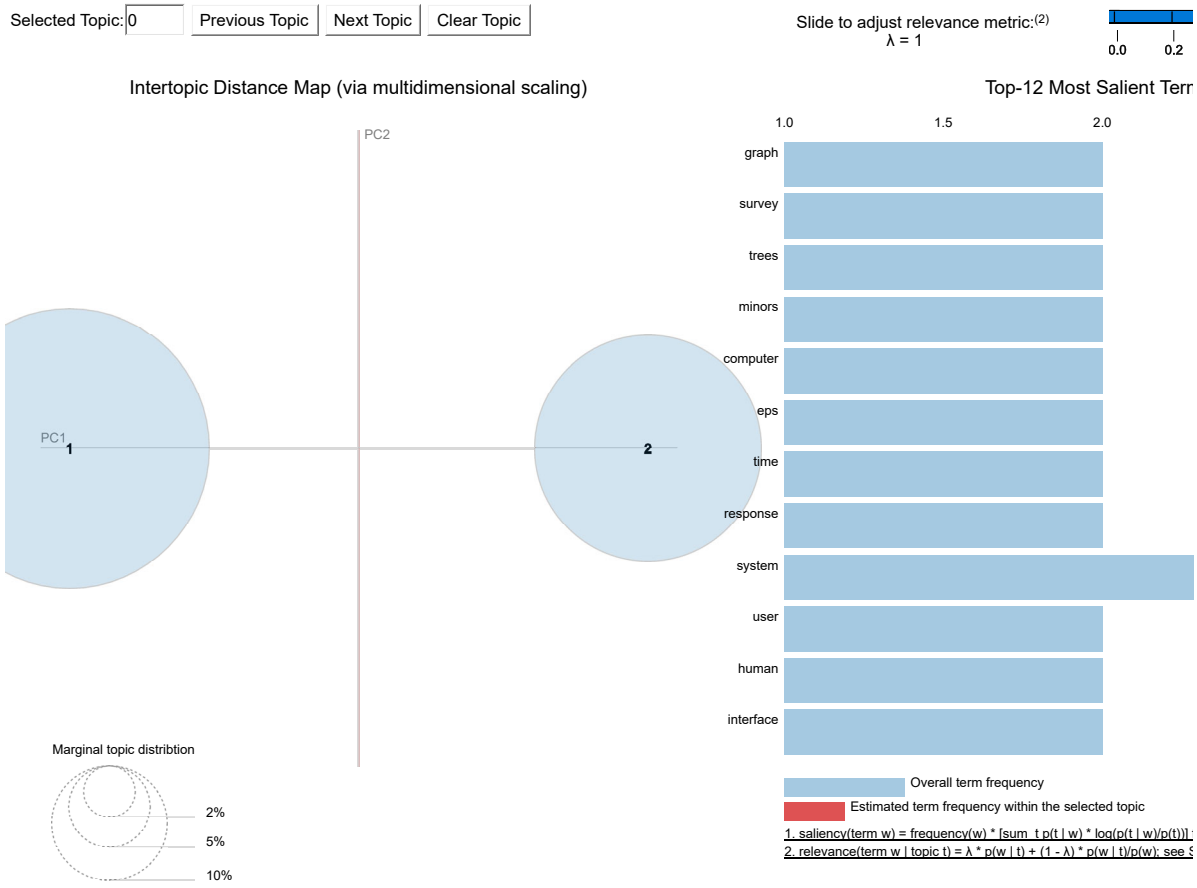
Therefore, the topic coherence for the goodLdaModel should be greater for this than the badLdaModel since the topics it comes up with are more human-interpr `u_mass` and `c_v` topic coherence measures.

Visualize topic models

In [17]:
```
pyLDAvis.enable_notebook()
```

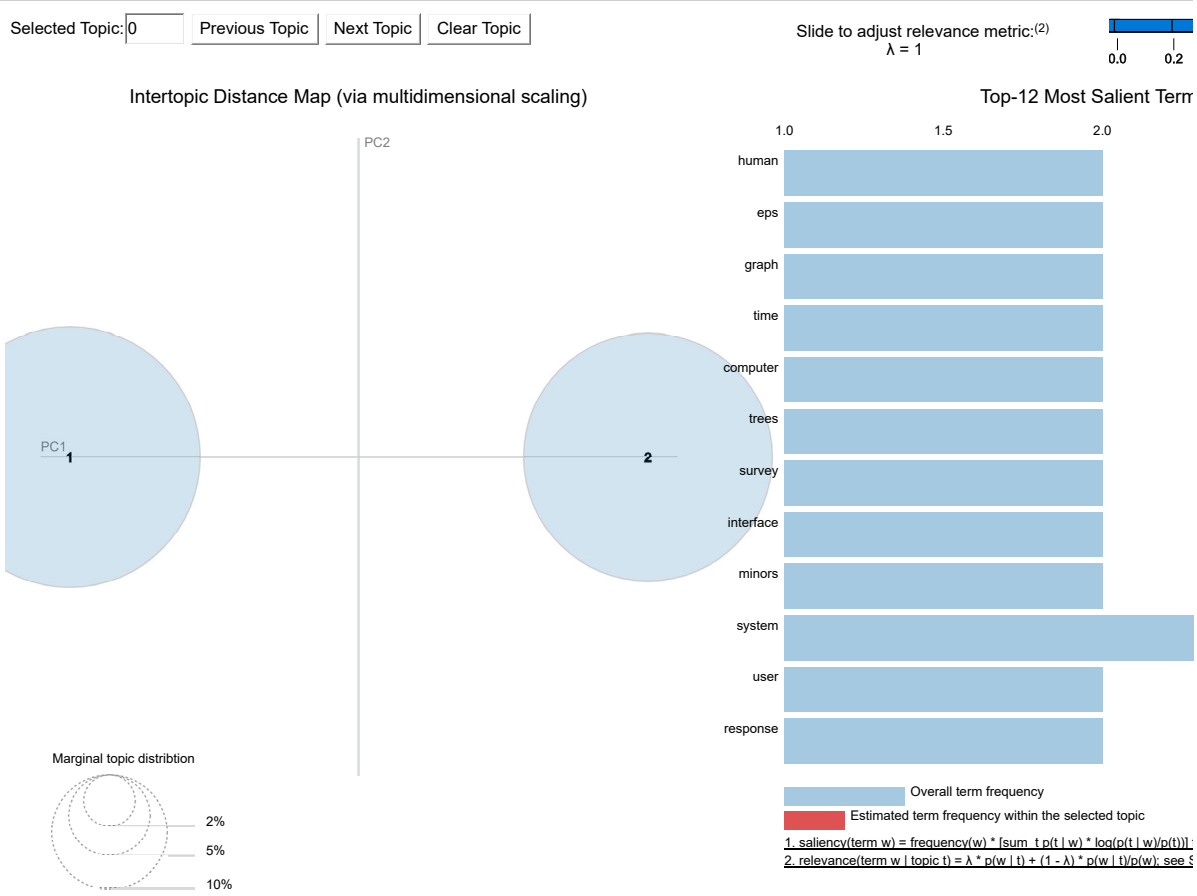
In [18]:
```
pyLDAvis.gensim.prepare(goodLdaModel, corpus, dictionary)
```

Out[18]:



In [19]:
```
pyLDAvis.gensim.prepare(badLdaModel, corpus, dictionary)
```

Out[19]:



In [20]:
```
print goodcm.get_coherence()
```

```
-14.0842451581
```

In [21]:
```
print badcm.get_coherence()
```

```
-14.4434307511
```

Using C_V coherence

In [25]:
```
goodcm = CoherenceModel(model=goodLdaModel, texts=texts, dictionary=dictionary, coherence='c_v')
```

In [26]:
```
badcm = CoherenceModel(model=badLdaModel, texts=texts, dictionary=dictionary, coherence='c_v')
```

Pipeline parameters for C_V coherence

In [27]:
```
print goodcm
```

```
CoherenceModel(segmentation=<function s_one_set at 0x7f663ae8a050>, probability estimation=<function p_boolean_sliding_window at
```

Print coherence values

In [28]:
```
print goodcm.get_coherence()
```

```
0.552164532134
```

In [29]:
```
print badcm.get_coherence()
```

```
0.5269189184
```

Conclusion

Hence as we can see, the `u_mass` and `c_v` coherence for the good LDA model is much more (better) than that for the bad LDA model. This is because, simpl
comes up with better topics that are more human interpretable. The badLdaModel however fails to decipher between these two topics and comes up with topics
The `u_mass` and `c_v` topic coherences capture this wonderfully by giving the interpretability of these topics a number as we can see above. Hence this coher
compare difference topic models based on their human-interpretability.