Topic-Modeling

June 15, 2021

1 Topic Modeling

1.0.1 01 Default Topic Modeling

[65]:
```
corpus = ['bread bread bread bread bread bread bread bread bread bread',
          'milk milk milk milk milk milk milk milk milk milk',
          'pet pet pet pet pet pet pet pet pet pet',
          'bread bread bread bread bread bread bread bread bread bread milk milk␣
 ↪milk milk milk milk milk milk milk milk']
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
vec = CountVectorizer(lowercase=True)

# https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.
 ↪text.CountVectorizer.html
matrixX = vec.fit_transform(corpus)  ## counter Vector izer

features = vec.get_feature_names()


lda = LatentDirichletAllocation(n_components=3)
lda.fit(matrixX)

print("Compponents : \n",lda.components_)

for tid,topic in enumerate(lda.components_):
    print("topic ID :",tid)
    print("words IDS : ",topic.argsort()[::-1])
    print("word : ",[features[i] for i in topic.argsort()[::-1]])
```

```
Compponents :
 [[ 0.33371676  0.33371671 10.33313136]
 [ 0.36228994 20.30299748  0.33343434]
 [20.3039933   0.3632858   0.3334343 ]]
topic ID : 0
words IDS :  [2 0 1]
word :  ['pet', 'bread', 'milk']
topic ID : 1
```

```
words IDS :  [1 0 2]
word :  ['milk', 'bread', 'pet']
topic ID : 2
words IDS :  [0 1 2]
word :  ['bread', 'milk', 'pet']
```

1.0.2 02 Topic Modeling using UCI Datasets

[52]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation
df = open("Datasets/Datasets.csv").read()
docs = df.split("\n")

tfidf = TfidfVectorizer()
matrixX =tfidf.fit_transform(docs)

features = tfidf.get_feature_names()

lda = LatentDirichletAllocation(n_components=5)
lda.fit(matrixX)


print("Components : \n",lda.components_)

for tid,topic in enumerate(lda.components_):
    print("Topic : ",tid)
    print("WordID ",topic.argsort()[::-1])
    print("word :",[features[i] for i in topic.argsort()[:-10:-1]])
```

```
Components :
 [[0.72276081 0.20444066 0.20002678 … 0.20006314 0.20005232 0.20003568]
 [0.20045891 0.85039319 0.51969443 … 0.21255988 0.30470302 0.62261786]
 [0.20001369 0.20011697 0.20003001 … 0.4703575  0.2000589  0.20004064]
 [0.20001054 0.4619186  0.20002114 … 0.20005534 0.20004294 0.20002802]
 [0.20000823 0.76662028 0.20001733 … 0.3795579  0.48602655 0.20002212]]
Topic :  0
WordID  [2546 2306  540 …  559 1106 1234]
word : ['you', 'thank', 'back', 'love', 'to', 'come', 'place', 'beautiful',
'will']
Topic :  1
WordID  [2310  470 2472 … 1758 1106  517]
word : ['the', 'and', 'we', 'to', 'you', 'is', 'was', 'of', 'hotel']
Topic :  2
WordID  [2546 2306 1255 … 1106    0 1234]
word : ['you', 'thank', 'great', 'love', '155', '279', '163', '19', '238']
Topic :  3
```

```
WordID  [2310 2472 2308 … 2290  517    0]
word : ['the', 'we', 'thanks', 'and', 'to', 'so', 'of', 'this', 'in']
Topic :  4
WordID  [2310  470 1174 …  559 1106    0]
word : ['the', 'and', 'for', 'we', 'to', 'you', 'is', 'in', 'of']
```

1.0.3 03 LDA with HyperParameters

[63]:
```
corpus = ['bread bread bread bread bread bread bread bread bread bread',
          'milk milk milk milk milk milk milk milk milk milk',
          'pet pet pet pet pet pet pet pet pet pet',
          'bread bread bread bread bread bread bread bread bread bread milk milk␣
 ↪milk milk milk milk milk milk milk milk']
from sklearn.feature_extraction.text import CountVectorizer

vec  = CountVectorizer()

matrix_X = vec.fit_transform(corpus)
features= vec.get_feature_names()

from sklearn.decomposition import LatentDirichletAllocation
lda = LatentDirichletAllocation(n_components=3 , topic_word_prior=0.1 ,␣
 ↪doc_topic_prior=0.1)
lda.fit(matrixX)


for tid,topic in enumerate(lda.components_):
    print("Topic : ",tid)
    print("WordID ",topic.argsort()[::-1])
    print("word :",[features[i] for i in topic.argsort()[:-10:-1]])
```

```
Topic :  0
WordID  [0 1 2]
word : ['bread', 'milk', 'pet']
Topic :  1
WordID  [2 1 0]
word : ['pet', 'milk', 'bread']
Topic :  2
WordID  [1 0 2]
word : ['milk', 'bread', 'pet']
```

1.0.4 04 Online LDA

[10]:
```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation
df = open("Datasets/Datasets.csv").read()
```

```
docs = df.split("\n")

tfidf = TfidfVectorizer()
matrixX =tfidf.fit_transform(docs)

features = tfidf.get_feature_names()

lda = LatentDirichletAllocation(n_components=2 , max_iter=200,learning_offset=4.
 ↪0,learning_method='online')
lda.fit(matrixX)


print("Components : \n",lda.components_)

for tid,topic in enumerate(lda.components_):
    print("Topic : ",tid)
    print("WordID ",topic.argsort()[::-1])
    print("word :",[features[i] for i in topic.argsort()[:-10:-1]])
```

```
Components :
 [[0.9254465  1.72779267 0.75691672 … 0.87805524 0.81466339 0.83221083]
 [0.52098404 0.51792382 0.51025146 … 0.51268105 0.51419621 0.52846798]]
Topic :  0
WordID  [2310  470 2472 …   33   62 1438]
word : ['the', 'and', 'we', 'to', 'you', 'for', 'of', 'place', 'is']
Topic :  1
WordID  [ 331  564  873 …  992 1567 2169]
word : ['394', 'beautiful', 'dear', 'maria', 'joana', 'hello', 'dazzled',
'review', 'wonder']
```

1.0.5 05 perplexity

[12]:
```
corpus = open("Datasets/Datasets.csv").read()
docs = corpus.split('\n')

from sklearn.feature_extraction.text import CountVectorizer
vec = CountVectorizer()
matrix_X = vec.fit_transform(docs)

from sklearn.decomposition import LatentDirichletAllocation
lda1 = LatentDirichletAllocation(n_components = 3)
lda2 = LatentDirichletAllocation(n_components = 2)

lda1.fit(matrix_X[:500])
lda2.fit(matrix_X[:500])

print('lda1: ', lda1.perplexity(matrix_X[500:]))
```

```
print('lda2: ', lda2.perplexity(matrix_X[500:]))
```

```
lda1:  2644.8666986269363
lda2:  2094.9233425456696
```

```
[ ]:
```