Analysis of Raw Text

Now that we know how to scrape text data donline. We need to analyze the data. In this tutorial, I'll try to go over the different steps of the text analysis.

Preliminary: Reading the data

In [1]:
```
import matplotlib
%matplotlib inline
import pandas as pd
```

```
/usr/local/lib/python2.7/dist-packages/matplotlib/font_manager.py:273: UserWarning: Matplotlib is building the font cache using fc-list. This may take a moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take a moment.')
```

In [2]:
```
data = pd.read_csv("queens_data.csv") #read data
```

In [3]:
```
for ix, name in enumerate(data.columns):
    print ix,name
```

```
0 Unnamed: 0
1 cool
2 date
3 eliteStatus
4 friends
5 funny
6 location
7 name
8 phone
9 ratings
10 review
11 reviewHistory
12 useful
13 userlink
14 username
```

In [4]:
```
data = data.drop(data.columns[[0]],axis =1) # drop first column not important!
data['date'] = pd.to_datetime(data['date'])
data.head()
```

Out[4]:

	cool	date	eliteStatus	friends	funny	location	name	phone	ratings	review	reviewHistory	useful	userlink	userr
0	1.0	2014-12-20	NaN	216	2.0	Long Island City, NY	Riverhead Inn	17183927664	3.0 star rating	Serves its purpose if you want to go some plac...	276	2.0	/user_details?userid=cT0m9dUAA4tg4q2iACszkw	Louis
1	2.0	2012-10-10	NaN	978	9.0	New York, NY	Riverhead Inn	17183927664	1.0 star rating	Uh, let's just say that Bill and Ted's adventu...	1789	5.0	/user_details?userid=ZD84600Tw1WqeKpVuNiKzg	Vivian
2	NaN	2015-11-27	NaN	0	NaN	Flushing, NY	Corato I Pizza Restaurant	17184976177	5.0 star rating	Michael says this is the best Pizzeria in Ridg...	1	NaN	/user_details?userid=w-iGjSjpp_ektS25EUCHVA	Micha
3	NaN	2015-11-27	NaN	0	1.0	Bronx, NY	Corato I Pizza Restaurant	17184976177	2.0 star rating	There should be a stricter rule when the serve...	6	1.0	/user_details?userid=RXML51WRJKho3EOyH8SGLw	Cherr
4	1.0	2015-11-24	NaN	0	NaN	Brooklyn, NY	Corato I Pizza Restaurant	17184976177	4.0 star rating	This is actually really good pizza! I was so s...	2	1.0	/user_details?userid=PyAP927Y3XazPJQNuVc8cA	Nikki

In [5]:
```
review = data.review[3] ##Look at the text data
review
```

Out[5]:
```
'There should be a stricter rule when the servers give you your food. I watched them make pizzas hands all on the dough then take dirty money then went back to the same
h to complete the pizza without washing their hands. Another worker had so much dirt under his nails the only thing missing was flowers. The only good thing is the food
ood just a little dirty lol'
```

Basic Steps

In this section, we will use the review above to practice the tokenization, stemming, and stop word removal steps.

Tokenization

The tokenization step is normally performed using **regular expression**. However, regular expression do have an entry cost and can be frustrating for beginners. Thus this famous quote by Jamie Zawinski(a famous hacker):

"Some people, when confronted with a problem, think "I know, I'll use regular expressions." Now they have two problems."

Thus, I normally recommend beginners use the nltk package in python when they are just starting. Let us first start with tokenzing sentences.

In [6]:
```
from nltk.tokenize import sent_tokenize
sentences = sent_tokenize(review)
sentences
```

Out[6]:
```
['There should be a stricter rule when the servers give you your food.',
 'I watched them make pizzas hands all on the dough then take dirty money then went back to the same dough to complete the pizza without washing their hands.',
 'Another worker had so much dirt under his nails the only thing missing was flowers.',
 'The only good thing is the food is good just a little dirty lol']
```

What about at the word level? nltk also has a package to perform it for you

In [7]:
```
from nltk.tokenize import word_tokenize
tokens = word_tokenize(review)
tokens
```

```
Out[7]: ['There',
         'should',
         'be',
         'a',
         'stricter',
         'rule',
         'when',
         'the',
         'servers',
         'give',
         'you',
         'your',
         'food',
         '.',
         'I',
         'watched',
         'them',
         'make',
         'pizzas',
         'hands',
         'all',
         'on',
         'the',
         'dough',
         'then',
         'take',
         'dirty',
         'money',
         'then',
         'went',
         'back',
         'to',
         'the',
         'same',
         'dough',
         'to',
         'complete',
         'the',
         'pizza',
         'without',
         'washing',
         'their',
         'hands',
         '.',
         'Another',
         'worker',
         'had',
         'so',
         'much',
         'dirt',
         'under',
         'his',
         'nails',
         'the',
         'only',
         'thing',
         'missing',
         'was',
```

```
'flowers',
'.',
'The',
'only',
'good',
'thing',
'is',
'the',
'food',
'is',
'good',
'just',
'a',
'little',
'dirty',
'lol']
```

This is really simple and effective. But, as we can see from the output there quite a few punctuation signs and words like "to", "it" in the output. There are 2 ways out of this issue one

```
1- Add them in the  stop words list (will go over this later).
2- Regular expression
```

Let's solve this problem using regular expression. To understand what is regular expression, I will refer you to the wikipedia page on the subject: https://en.wikipedia.org/wiki/Regular_expression (https://en.wikipedia.org/wiki/Regular_expression), this website is also useful for practicing http://regexr.com/ (http://regexr.com/)

A regular expression is just a sequence of characters use for pattern matching. To use them in python, we can just call the re package.

In [8]:
```
import re
re.findall(r'[a-zA-Z]+', review) # to explained meaning in class!
```

```
Out[8]: ['There',
         'should',
         'be',
         'a',
         'stricter',
         'rule',
         'when',
         'the',
         'servers',
         'give',
         'you',
         'your',
         'food',
         'I',
         'watched',
         'them',
         'make',
         'pizzas',
         'hands',
         'all',
         'on',
         'the',
         'dough',
         'then',
         'take',
         'dirty',
         'money',
         'then',
         'went',
         'back',
         'to',
         'the',
         'same',
         'dough',
         'to',
         'complete',
         'the',
         'pizza',
         'without',
         'washing',
         'their',
         'hands',
         'Another',
         'worker',
         'had',
         'so',
         'much',
         'dirt',
         'under',
         'his',
         'nails',
         'the',
         'only',
         'thing',
         'missing',
         'was',
         'flowers',
         'The',
```

```
'only',
'good',
'thing',
'is',
'the',
'food',
'is',
'good',
'just',
'a',
'little',
'dirty',
'lol']
```

Now all the punctuation signs are gone but the "a", "it" are not really informative. Thus, the need to get rid of them. As a general rule, we always dropped all *words that are less than 3 characters*. Furthermore, a closer look at the output and we see that the word "isn't" is tokenized into "isn" and "t" which is also problematic. To take care of these issues, we will use the following regex

In [9]:
```
token = re.findall(r"[a-zA-Z']{3,}", review)
token
```

```
Out[9]: ['There',
         'should',
         'stricter',
         'rule',
         'when',
         'the',
         'servers',
         'give',
         'you',
         'your',
         'food',
         'watched',
         'them',
         'make',
         'pizzas',
         'hands',
         'all',
         'the',
         'dough',
         'then',
         'take',
         'dirty',
         'money',
         'then',
         'went',
         'back',
         'the',
         'same',
         'dough',
         'complete',
         'the',
         'pizza',
         'without',
         'washing',
         'their',
         'hands',
         'Another',
         'worker',
         'had',
         'much',
         'dirt',
         'under',
         'his',
         'nails',
         'the',
         'only',
         'thing',
         'missing',
         'was',
         'flowers',
         'The',
         'only',
         'good',
         'thing',
         'the',
         'food',
         'good',
         'just',
```

```
 'little',
 'dirty',
 'lol']
```

This is a much better output; but, is by no means perfect. For example, if you want to seperate hashtag from other text than this would not be suitable. Furthermore, if the text is in a foreign language than this will also fail.

One should notice that words like **"The"** are still in the token list; Even though it does not add any value to our analysis. Thus it should be removed.

Stop Words Removal

Let us take a quick look at the top words of the model

In [10]:
```
from collections import Counter
count = Counter()
for text in data.review:
    token_list = Counter(re.findall(r"[a-zA-Z]{3,}",text.lower()))
    for token_ in token_list:
        count[token_] +=1

count.most_common(10)
```

Out[10]:
```
[('and', 53),
 ('the', 53),
 ('pizza', 45),
 ('for', 31),
 ('they', 29),
 ('but', 29),
 ('was', 28),
 ('that', 26),
 ('good', 25),
 ('with', 25)]
```

As we can see quite a few of the words contains no meaningfull information. Thus we need to remove them.

Stop words are just a list of words that we would like to filter out. Luckily, nltk provides us with a list of stop words that we use.

In [11]:
```
from nltk.corpus import stopwords
stop = stopwords.words('english')
stop
```

```
Out[11]: [u'i',
          u'me',
          u'my',
          u'myself',
          u'we',
          u'our',
          u'ours',
          u'ourselves',
          u'you',
          u'your',
          u'yours',
          u'yourself',
          u'yourselves',
          u'he',
          u'him',
          u'his',
          u'himself',
          u'she',
          u'her',
          u'hers',
          u'herself',
          u'it',
          u'its',
          u'itself',
          u'they',
          u'them',
          u'their',
          u'theirs',
          u'themselves',
          u'what',
          u'which',
          u'who',
          u'whom',
          u'this',
          u'that',
          u'these',
          u'those',
          u'am',
          u'is',
          u'are',
          u'was',
          u'were',
          u'be',
          u'been',
          u'being',
          u'have',
          u'has',
          u'had',
          u'having',
          u'do',
          u'does',
          u'did',
          u'doing',
          u'a',
          u'an',
          u'the',
          u'and',
          u'but',
```

```
                u'if',
                u'or',
                u'because',
                u'as',
                u'until',
                u'while',
                u'of',
                u'at',
                u'by',
                u'for',
                u'with',
                u'about',
                u'against',
                u'between',
                u'into',
                u'through',
                u'during',
                u'before',
                u'after',
                u'above',
                u'below',
                u'to',
                u'from',
                u'up',
                u'down',
                u'in',
                u'out',
                u'on',
                u'off',
                u'over',
                u'under',
                u'again',
                u'further',
                u'then',
                u'once',
                u'here',
                u'there',
                u'when',
                u'where',
                u'why',
                u'how',
                u'all',
                u'any',
                u'both',
                u'each',
                u'few',
                u'more',
                u'most',
                u'other',
                u'some',
                u'such',
                u'no',
                u'nor',
                u'not',
                u'only',
                u'own',
                u'same',
                u'so',
                u'than',
```

```
 u'too',
 u'very',
 u's',
 u't',
 u'can',
 u'will',
 u'just',
 u'don',
 u'should',
 u'now']
```

Although, it is a pretty decent list, one should notice that this list is generic and is by no mean complete. There some situation where we need to increment that list. One way to increment it is by looking at the most frequent words list and removing the one that do not add any value to our objectives. Apply this list of stop words to our review and we get:

In [12]:
```
review_stop = [element for element in token if element not in stop]
" ".join(review_stop)
```

Out[12]:
```
'There stricter rule servers give food watched make pizzas hands dough take dirty money went back dough complete pizza without washing hands Another worker much dirt nai
hing missing flowers The good thing food good little dirty lol'
```

As we can see now The word **the** is now gone but "isn't" is still there.

Stemming

As we know already, stemming is the act of reducing a word to its root. worked, work, works => work They exist multiple different stemming algorithm; but, the most popular one are Porter's and Lancaster. It should be said that Porter stemmer is more well known in marketing.

In [13]:
```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
" ".join([stemmer.stem(tok) for tok in review_stop])
```

Out[13]:
```
u'There stricter rule server give food watch make pizza hand dough take dirti money went back dough complet pizza without wash hand Anoth worker much dirt nail thing mis
ower The good thing food good littl dirti lol'
```

In [14]:
```
from nltk.stem import LancasterStemmer
stemmer = LancasterStemmer()
" ".join([stemmer.stem(tok) for tok in review_stop])
```

Out[14]:
```
'ther stricter rul serv giv food watch mak pizza hand dough tak dirty money went back dough complet pizz without wash hand anoth work much dirt nail thing miss flow the
thing food good littl dirty lol'
```

A comparison in the reults of the output of the 2 different stemmers and one can see that the Landcaster stemmer is more aggresive in stemming words. It is worth noting that when your goal is prediction, one should try different type of stemming.

Corpus Level Operation

Now that we have seen the basic in 1 text passage, we will now focus on doing this on an entire document of text. This is where python will come handy. More precisely, the sklearn package in python contain a function that will perform all the above for you in 1 line of command.

In [15]:
```
from sklearn.feature_extraction.text import CountVectorizer
vec = CountVectorizer(decode_error = 'ignore', token_pattern = "[a-zA-Z]{3,}",lowercase= True,stop_words = stopwords.words("english"), max_features =10)
```

In [16]:
```
sparse_matrix = vec.fit_transform(data.review)
sparse_matrix.toarray()
```

```
Out[16]: array([[0, 0, 0, 1, 1, 0, 0, 0, 1, 1],
                [0, 0, 0, 0, 0, 0, 0, 0, 1, 1],
                [0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
                [0, 0, 2, 2, 0, 0, 0, 1, 0, 0],
                [0, 0, 0, 1, 0, 2, 1, 2, 0, 1],
                [1, 0, 0, 0, 0, 0, 1, 2, 1, 0],
                [0, 0, 1, 0, 1, 1, 1, 2, 0, 1],
                [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
                [0, 4, 2, 0, 0, 4, 0, 1, 0, 0],
                [0, 0, 0, 0, 1, 1, 1, 0, 0, 0],
                [0, 0, 0, 3, 0, 0, 0, 1, 0, 0],
                [1, 0, 1, 1, 1, 0, 0, 5, 2, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 1, 1, 0, 0, 0, 3, 0, 2],
                [1, 0, 0, 0, 0, 0, 0, 2, 0, 0],
                [1, 1, 3, 2, 2, 2, 0, 0, 0, 1],
                [0, 0, 0, 0, 0, 1, 1, 1, 0, 0],
                [0, 2, 2, 0, 2, 1, 1, 1, 0, 0],
                [0, 0, 1, 1, 0, 0, 1, 3, 0, 1],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 1, 0, 1, 0, 0, 1, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
                [1, 2, 0, 2, 0, 1, 2, 1, 0, 1],
                [0, 1, 0, 0, 0, 0, 1, 0, 0, 0],
                [0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
                [0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 3, 0, 1, 0, 2, 0, 1],
                [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
                [0, 0, 2, 1, 0, 2, 0, 1, 0, 0],
                [0, 1, 0, 0, 1, 1, 1, 1, 0, 0],
                [0, 1, 0, 1, 0, 1, 0, 3, 0, 0],
                [1, 0, 1, 0, 3, 0, 0, 2, 0, 2],
                [1, 0, 0, 0, 0, 0, 0, 1, 0, 1],
                [0, 0, 0, 2, 1, 0, 0, 1, 0, 0],
                [0, 0, 0, 2, 0, 0, 2, 0, 0, 1],
                [1, 0, 0, 1, 0, 0, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
                [0, 0, 0, 1, 1, 0, 0, 1, 0, 0],
                [0, 0, 0, 0, 2, 1, 0, 4, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [1, 0, 0, 1, 1, 0, 0, 2, 0, 0],
                [0, 0, 2, 1, 0, 1, 0, 1, 3, 2],
                [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 5, 0, 0],
                [0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
                [0, 0, 0, 3, 0, 0, 0, 2, 3, 2],
                [0, 0, 1, 0, 1, 0, 0, 4, 1, 0],
                [1, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0],
                [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                [0, 1, 0, 1, 0, 0, 0, 1, 0, 0],
                [0, 0, 0, 0, 0, 0, 1, 0, 2, 0],
                [0, 0, 1, 0, 0, 0, 0, 1, 2, 0],
                [2, 0, 0, 2, 0, 0, 0, 0, 0, 3],
                [1, 0, 1, 0, 0, 0, 0, 1, 1, 0],
                [0, 0, 0, 0, 1, 0, 1, 3, 1, 0],
                [0, 1, 0, 0, 0, 0, 0, 1, 1, 0],
                [4, 2, 0, 1, 7, 2, 0, 1, 1, 2],
```

```
       [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 1, 0, 1, 0, 0],
       [0, 0, 0, 1, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
       [0, 0, 0, 2, 1, 2, 1, 4, 0, 0],
       [0, 0, 0, 2, 0, 0, 0, 2, 1, 0]])
```

let us format this a little bit more, so that we understand what we are observing.

In [17]:
```
Term_document = pd.DataFrame(sparse_matrix.toarray(), columns = vec.get_feature_names())
Term_document
```

Out[17]:

	also	delivery	food	good	like	order	ordered	pizza	place	really
0	0	0	0	1	1	0	0	0	1	1
1	0	0	0	0	0	0	0	0	1	1
2	0	0	0	0	0	0	0	0	1	0
3	0	0	2	2	0	0	0	1	0	0
4	0	0	0	1	0	2	1	2	0	1
5	1	0	0	0	0	0	1	2	1	0
6	0	0	1	0	1	1	1	2	0	1
7	0	0	0	0	0	0	0	1	0	0
8	0	4	2	0	0	4	0	1	0	0
9	0	0	0	0	1	1	1	0	0	0
10	0	0	0	3	0	0	0	1	0	0
11	1	0	1	1	1	0	0	5	2	0
12	0	0	0	0	0	0	0	0	0	0
13	0	0	1	1	0	0	0	3	0	2
14	1	0	0	0	0	0	0	2	0	0
15	1	1	3	2	2	2	0	0	0	1
16	0	0	0	0	0	1	1	1	0	0
17	0	2	2	0	2	1	1	1	0	0
18	0	0	1	1	0	0	1	3	0	1
19	0	0	0	0	0	0	0	0	0	0
20	0	0	1	0	1	0	0	1	0	0
21	0	0	0	0	0	0	0	1	0	0
22	1	2	0	2	0	1	2	1	0	1
23	0	1	0	0	0	0	1	0	0	0
24	0	1	0	0	0	1	0	0	0	0
25	0	0	1	0	0	0	0	0	0	0
26	0	0	0	3	0	1	0	2	0	1
27	0	0	0	0	0	0	0	1	0	0
28	0	0	2	1	0	2	0	1	0	0
29	0	1	0	0	1	1	1	1	0	0
...	...	...	...	...	...	...	...	...	...	...
34	0	0	0	2	0	0	2	0	0	1
35	1	0	0	1	0	0	0	0	0	0

	also	delivery	food	good	like	order	ordered	pizza	place	really
36	0	0	0	0	0	0	0	1	0	0
37	0	0	0	1	1	0	0	1	0	0
38	0	0	0	0	2	1	0	4	0	0
39	0	0	0	0	0	0	0	0	0	0
40	1	0	0	1	1	0	0	2	0	0
41	0	0	2	1	0	1	0	1	3	2
42	0	1	0	0	0	0	0	0	0	0
43	0	1	0	0	0	1	0	0	0	0
44	0	0	0	0	0	0	0	5	0	0
45	0	0	0	0	0	0	1	0	0	0
46	0	0	0	3	0	0	0	2	3	2
47	0	0	1	0	1	0	0	4	1	0
48	1	0	0	0	0	0	2	2	0	0
49	0	0	0	0	0	0	0	0	0	0
50	0	1	0	1	0	0	0	1	0	0
51	0	0	0	0	0	0	1	0	2	0
52	0	0	1	0	0	0	0	1	2	0
53	2	0	0	2	0	0	0	0	0	3
54	1	0	1	0	0	0	0	1	1	0
55	0	0	0	0	1	0	1	3	1	0
56	0	1	0	0	0	0	0	1	1	0
57	4	2	0	1	7	2	0	1	1	2
58	0	0	0	0	0	0	0	1	0	0
59	0	0	0	0	0	1	0	1	0	0
60	0	0	0	1	0	0	0	1	0	0
61	0	0	0	0	0	0	0	1	0	0
62	0	0	0	2	1	2	1	4	0	0
63	0	0	0	2	0	0	0	2	1	0

64 rows × 10 columns

What about TF-Idf?

Sklearn also have a TFidf vectorizer, that automatically compute the tfidf scores for you.

In [18]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
vec = TfidfVectorizer(decode_error = 'ignore', token_pattern = "[a-zA-Z]{3,}",lowercase= True,stop_words = stopwords.words("english"), max_features =10)
sparse_matrix = vec.fit_transform(data.review)
pd.DataFrame(sparse_matrix.toarray(), columns = vec.get_feature_names())
```

Out[18]:

	also	delivery	food	good	like	order	ordered	pizza	place	really
0	0.000000	0.000000	0.000000	0.427018	0.508961	0.000000	0.000000	0.000000	0.535207	0.521697
1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.716087	0.698012
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000
3	0.000000	0.000000	0.755374	0.618286	0.000000	0.000000	0.000000	0.217101	0.000000	0.000000
4	0.000000	0.000000	0.000000	0.299770	0.000000	0.681624	0.366235	0.421036	0.000000	0.366235
5	0.507861	0.000000	0.000000	0.000000	0.000000	0.000000	0.468969	0.539143	0.481113	0.000000
6	0.000000	0.000000	0.403587	0.000000	0.393733	0.375571	0.403587	0.463977	0.000000	0.403587
7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
8	0.000000	0.712613	0.329020	0.000000	0.000000	0.612361	0.000000	0.094563	0.000000	0.000000
9	0.000000	0.000000	0.000000	0.000000	0.581185	0.554375	0.595729	0.000000	0.000000	0.000000
10	0.000000	0.000000	0.000000	0.973678	0.000000	0.000000	0.000000	0.227927	0.000000	0.000000
11	0.268521	0.000000	0.247958	0.202957	0.241904	0.000000	0.000000	0.712651	0.508757	0.000000
12	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
13	0.000000	0.000000	0.340134	0.278406	0.000000	0.000000	0.000000	0.586544	0.000000	0.680268
14	0.685675	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.727908	0.000000	0.000000
15	0.229342	0.229342	0.635337	0.346690	0.413217	0.394156	0.000000	0.000000	0.000000	0.211779
16	0.000000	0.000000	0.000000	0.000000	0.000000	0.627913	0.674753	0.387859	0.000000	0.000000
17	0.000000	0.565008	0.521739	0.000000	0.509002	0.242761	0.260870	0.149952	0.000000	0.000000
18	0.000000	0.000000	0.387967	0.317558	0.000000	0.000000	0.387967	0.669030	0.000000	0.387967
19	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
20	0.000000	0.000000	0.661950	0.000000	0.645788	0.000000	0.000000	0.380500	0.000000	0.000000
21	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
22	0.282067	0.564134	0.000000	0.426392	0.000000	0.242385	0.520933	0.149720	0.000000	0.260466
23	0.000000	0.734678	0.000000	0.000000	0.000000	0.000000	0.678416	0.000000	0.000000	0.000000
24	0.000000	0.758441	0.000000	0.000000	0.000000	0.651742	0.000000	0.000000	0.000000	0.000000
25	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
26	0.000000	0.000000	0.000000	0.808808	0.000000	0.306515	0.000000	0.378666	0.000000	0.329379
27	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
28	0.000000	0.000000	0.687439	0.281340	0.000000	0.639718	0.000000	0.197576	0.000000	0.000000
29	0.000000	0.520971	0.000000	0.000000	0.469330	0.447680	0.481075	0.276530	0.000000	0.000000
...	...	...	...	...	...	...	...	...	...	...
34	0.000000	0.000000	0.000000	0.590718	0.000000	0.000000	0.721693	0.000000	0.000000	0.360847
35	0.797760	0.000000	0.000000	0.602975	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

	also	delivery	food	good	like	order	ordered	pizza	place	really
36	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
37	0.000000	0.000000	0.000000	0.585829	0.698246	0.000000	0.000000	0.411408	0.000000	0.000000
38	0.000000	0.000000	0.000000	0.000000	0.618262	0.294871	0.000000	0.728562	0.000000	0.000000
39	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
40	0.533772	0.000000	0.000000	0.403444	0.480862	0.000000	0.000000	0.566650	0.000000	0.000000
41	0.000000	0.000000	0.454798	0.186130	0.000000	0.211613	0.000000	0.130713	0.699862	0.454798
42	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
43	0.000000	0.758441	0.000000	0.000000	0.000000	0.651742	0.000000	0.000000	0.000000	0.000000
44	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
45	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000
46	0.000000	0.000000	0.000000	0.538111	0.000000	0.000000	0.000000	0.251931	0.674446	0.438281
47	0.000000	0.000000	0.347296	0.000000	0.338817	0.000000	0.000000	0.798527	0.356290	0.000000
48	0.424944	0.000000	0.000000	0.000000	0.000000	0.000000	0.784803	0.451118	0.000000	0.000000
49	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50	0.000000	0.734613	0.000000	0.555246	0.000000	0.000000	0.000000	0.389930	0.000000	0.000000
51	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.438115	0.000000	0.898919	0.000000
52	0.000000	0.000000	0.424849	0.000000	0.000000	0.000000	0.000000	0.244211	0.871702	0.000000
53	0.535298	0.000000	0.000000	0.404597	0.000000	0.000000	0.000000	0.000000	0.000000	0.741457
54	0.574306	0.000000	0.530326	0.000000	0.000000	0.000000	0.000000	0.304840	0.544058	0.000000
55	0.000000	0.000000	0.000000	0.000000	0.399015	0.000000	0.409000	0.705301	0.419591	0.000000
56	0.000000	0.677412	0.000000	0.000000	0.000000	0.000000	0.000000	0.359569	0.641734	0.000000
57	0.485492	0.242746	0.000000	0.091738	0.765393	0.208596	0.000000	0.064424	0.114980	0.224156
58	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
59	0.000000	0.000000	0.000000	0.000000	0.000000	0.850779	0.000000	0.525523	0.000000	0.000000
60	0.000000	0.000000	0.000000	0.818360	0.000000	0.000000	0.000000	0.574706	0.000000	0.000000
61	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000
62	0.000000	0.000000	0.000000	0.447501	0.266687	0.508769	0.273361	0.628529	0.000000	0.000000
63	0.000000	0.000000	0.000000	0.728182	0.000000	0.000000	0.000000	0.511378	0.456337	0.000000

64 rows × 10 columns

Latent Dirichlet Allocation

As we can see from the matrix above there are a lot of 0 and if we were to use all the vocabulary the matrix above would be a lot bigger. This is where, LDA can help. LDA was built for the purpose of finding hidden themes in large corpora. It is great for classifying text in to categories and reducing the dimension of the matrix.

There are a lot of packages online that can be used to perform LDA. My 2 prefered packages in python are "gensim" and "sklearn". Gensim is built purely for textual analysis, it contains different variation of LDA and other great text analysis software such as LSI etc. On the other hand sklearn is a machine learning package, if your goal is to use output of LDA to predict an outcome then this is the best package for the task.

For this demonstration, I will use the sklearn packages.

I am splitting the data into a train test set. As we need an test set to find the optimal number of topics.

In [19]:
```
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.cross_validation import train_test_split
vec = CountVectorizer( token_pattern= "[a-zA-Z']{3,}",decode_error ='ignore', lowercase=True, stop_words = stopwords.words('english'),max_features = 100)
sparse_matrice = vec.fit_transform(data.review)
X_train, X_test = train_test_split(sparse_matrice, test_size=0.20, random_state=42) ## 80 -20 split
```

Now that we are done with all the preliminaries, let us run the LDA with the code below. The function takes the following argument:

- n_topics : the number of topics you want to define.
- topic_word_prior : which is nomrally set at 0.1 to induce sparse and reproducible solution(see Griffith Steyvers 2004)
- doc_topic_prior : which is normally set to 50.0/n_topics but that is assuming that you are starting with 50 topics, so for this demo I set it to 1
- Learning_method : "online" is big data sets and "Batch" is for small data set.
- n_jobs : is the number of cores you want to use, I suggest using 1, unless your data is extremely big.
- max_iter : is the number of iteration that you want! Griffith Stevers showed that using gibbs sampling that it converges with less 1000 iterations. To be safe, I used 1500.
- random_State : is a parameter for reproducibility.

In [20]:
```
lda = LatentDirichletAllocation(n_topics = 10, max_iter=1500, topic_word_prior= 0.01, doc_topic_prior=1.0,learning_method='batch',n_jobs= 1,random_state=275)

topic_components = lda.fit_transform(X_train)
```

To see the top words of the various topics, you can use this function to observe the top words of the topics.

In [21]:
```
def print_top_words(model, feature_names, n_top_words):
    for topic_idx, topic in enumerate(model.components_):
        print("Topic #%d:" % topic_idx)
        print(" ".join([feature_names[i]
                        for i in topic.argsort()[:-n_top_words - 1:-1]]))
    print()
```

In [22]:
```
print_top_words(lda,vec.get_feature_names(), 5) #print the top 5 words!
```

```
Topic #0:
pizza i've eat coratos one
Topic #1:
rosa's would pizza even corato
Topic #2:
ordered bad cheese delicious slices
Topic #3:
chicken best around pizzeria got
Topic #4:
food always service never i'm
Topic #5:
good really pizza place food
Topic #6:
like away better time blocks
Topic #7:
slice fresh don't also it's
Topic #8:
order delivery get customer little
Topic #9:
pizza card worth corato's area
()
```

As we can see this works but the results are not that great. This is due to the fact that the data set is kind of small and I probably did not set the correct number of topics.

Finding the optimal number of topics

In the nlp litterature, the criterion for the optimal number of topics is the perplexity from a **hold out set**. The perplexity is a goodness of fit measure widely use in this type of field. The rule of thumb is the smaller the better. To see the perplexity of the our small model, you just need to enter the following code.

In [23]:
```
lda.perplexity(X_test)
```

Out[23]:
```
11871.586580352408
```

Note: that there is a paper that shows that using the optimal number of topics does not always produce the best topic for understanding. So I would take the optimal number of topics with a caveat.

Either ways for your homework you might want to run LDA for different number of topics and look at the perplexity and the word grouping you have to decide.

Fancier output

Althoough, what we showed so far is really not pretty you can run use "pyLDAvis" to see the topics. The pyLDAvis package require inputs from a gensim trained model. Although this is not crucial for the assignment, it is something that you should be aware of.

In [24]:
```
import pyLDAvis
import gensim
dictionary = gensim.corpora.Dictionary.load('yelp_token.dict')
corpus = gensim.corpora.MmCorpus('yelp_data.mm')
lda = gensim.models.ldamodel.LdaModel.load('model 50 component.lda')
```

```
/usr/local/lib/python2.7/dist-packages/numpy/lib/utils.py:99: DeprecationWarning: `scipy.sparse.sparsetools` is deprecated!
scipy.sparse.sparsetools is a private module for scipy.sparse, and should not be used.
  warnings.warn(depdoc, DeprecationWarning)
```

In [25]:
```
import pyLDAvis.gensim
pyLDAvis.enable_notebook()
pyLDAvis.gensim.prepare(lda, corpus, dictionary)
```

```
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
```

```
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
```

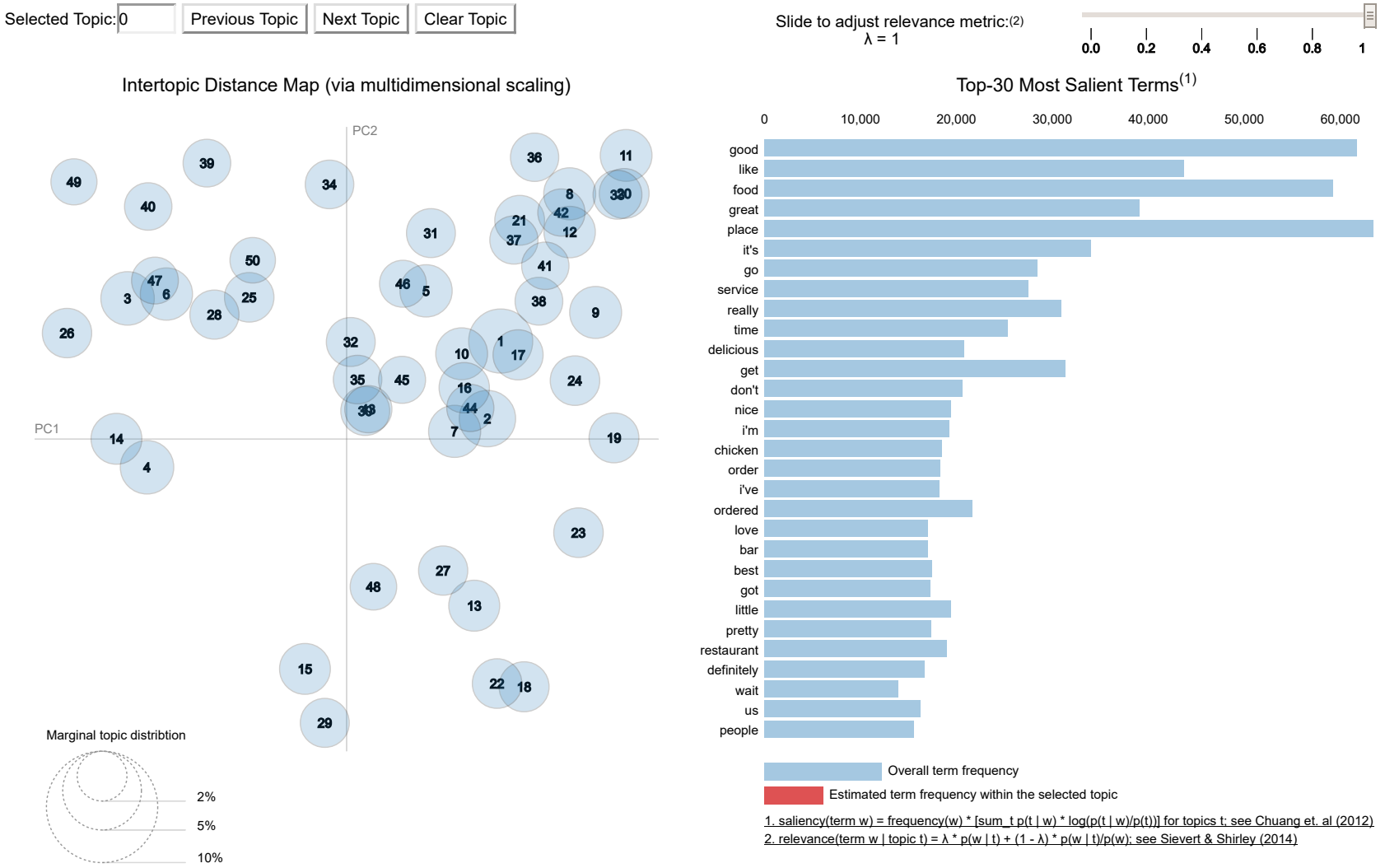
```
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
```

```
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
```

```
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
```

```
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
/usr/lib/python2.7/dist-packages/numexpr/necompiler.py:742: DeprecationWarning: using `oa_ndim == 0` when `op_axes` is NULL is deprecated. Use `oa_ndim == -1` or the Mul
New iterator for NumPy <1.8 compatibility
  return compiled_ex(*arguments, **kwargs)
```

Out[25]:


Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0
0.2
0.4
0.6
0.8
1
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
Marginal topic distribtion
2%
5%
10%
Top-30 Most Salient Terms(1)
0
10,000
20,000
30,000
40,000
50,000
60,000
good
like
food
great
place
it's
go
service
really
time
delicious
get
don't
nice
i'm
chicken
order
i've
ordered
love
bar
best
got
little
pretty
restaurant
definitely
wait
us
people
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

In [ ]: