

# Two-day workshop

## Automated Content Analysis with Python

### Day 2

Damian Trilling

d.c.trilling@uva.nl  
@damian0604  
www.damiantrilling.net

Afdeling Communicatiewetenschap  
Universiteit van Amsterdam

2-2-2017

# Today

- ① Recap: Types of Automated Content Analysis
- ② Supervised Machine Learning
- ③ Unsupervised machine learning

## Recap: Types of Automated Content Analysis

# Explicit rules vs. implicit patterns

	Methodological approach		
	<i>Counting and Dictionary</i>	<i>Supervised Machine Learning</i>	<i>Unsupervised Machine Learning</i>
<b>Typical research interests and content features</b>	visibility analysis sentiment analysis subjectivity analysis	frames topics gender bias	frames topics
<b>Common statistical procedures</b>	string comparisons counting	support vector machines naive Bayes	principal component analysis cluster analysis latent dirichlet allocation semantic network analysis
<div> <div>deductive</div> <div>inductive</div> </div>			

Boumans, J.W., & Trilling, D. (2016). Taking stock of the toolkit: An overview of relevant automated content analysis approaches and techniques for digital journalism scholars. *Digital Journalism*, 4, 1. 8–23.

# Machine learning: supervised vs. unsupervised

## Unsupervised

PCA, LDA, Cluster Analysis

- No manually coded data
- We want to identify patterns or to make groups of most similar cases

Example: We have a dataset of Facebook-messages on an organizations' page. We use clustering to group them and later interpret these clusters (e.g., as complaints, questions, praise, ...)

## Supervised

Regression, Naïve Bayes, SVM

- We code a small dataset by hand and use it to “train” a machine
- The machine codes the rest

Example: We have 2,000 of these messages grouped into such categories by human coders. We then use this data to group all remaining messages as well.

## Supervised Machine Learning

# You have done it before!

## Regression

- 1 Based on your data, you estimate some regression equation  
$$y_i = \alpha + \beta_1 x_{i1} + \dots + \beta_p x_{ip} + \varepsilon_i$$
- 2 Even if you have some *new unseen data*, you can estimate your expected outcome  $\hat{y}$ !
- 3 Example: You estimated a regression equation where  $y$  is newspaper reading in days/week:  
$$y = -.8 + .4 \times \text{man} + .08 \times \text{age}$$
- 4 You could now calculate  $\hat{y}$  for a man of 20 years and a woman of 40 years – *even if no such person exists in your dataset*:  
$$\hat{y}_{\text{man}20} = -.8 + .4 \times 1 + .08 \times 20 = 1.2$$
$$\hat{y}_{\text{woman}40} = -.8 + .4 \times 0 + .08 \times 40 = 2.4$$

# This is Supervised Machine Learning!



... but...

- We will only use *half* (or another fraction) of our data to estimate the model, so that we can use the other half to check if our predictions match the manual coding (“labeled data”, “annotated data” in SML-lingo)
  - e.g., 2000 labeled cases, 1000 for training, 1000 for testing — if successful, run on 100,000 unlabeled cases
- We use many more independent variables (“features”)
- Typically, IVs are word frequencies (often weighted, e.g.  $\text{tf} \times \text{idf}$ ) ( $\Rightarrow$  BOW-representation)

# Applications

## In other fields

*A lot* of different applications

- from recognizing hand-written characters to recommendation systems

## In our field

It starts to get popular to measure latent variables

- frames
- topics

# SML to code frames and topics

## Some work by Burscher and colleagues

- Humans can code generic frames (human-interest, economic, ...)
- Humans can code topics from a pre-defined list
- **But it is very hard to formulate an explicit rule**  
(as in: code as 'Human Interest' if regular expression R is matched)

⇒ This is where you need supervised machine learning!

Burscher, B., Odijk, D., Vliegthart, R., De Rijke, M., & De Vreese, C. H. (2014). Teaching the computer to code frames in news: Comparing two supervised machine learning approaches to frame analysis. *Communication Methods and Measures*, 8(3), 190–206. doi:10.1080/19312458.2014.937527

Burscher, B., Vliegthart, R., & De Vreese, C. H. (2015). Using supervised machine learning to code policy issues: Can classifiers generalize across contexts? *Annals of the American Academy of Political and Social Science*, 659(1), 122–131.

TABLE 4  
Classification Accuracy of Frames in Sources Outside the Training Set

	<i>VK/NRC</i> <i>→ Tel</i>	<i>VK/TEL</i> <i>→ NRC</i>	<i>NRC/TEL</i> <i>→ VK</i>
Conflict	.69	.74	.75
Economic Cons.	.88	.86	.86
Human Interest	.69	.71	.67
Morality	.97	.90	.89

*Note.* VK = Volkskrant, NRC = NRC/Handelsblad, TEL = Telegraaf

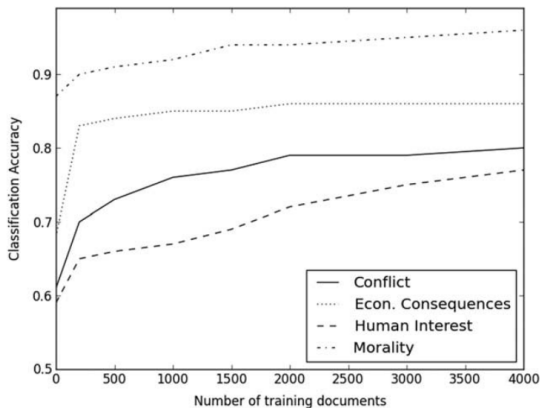


FIGURE 1 Relationship between classification accuracy and number of training documents.

FIGURE 1

Learning Curves for the Classification of News Articles and PQs

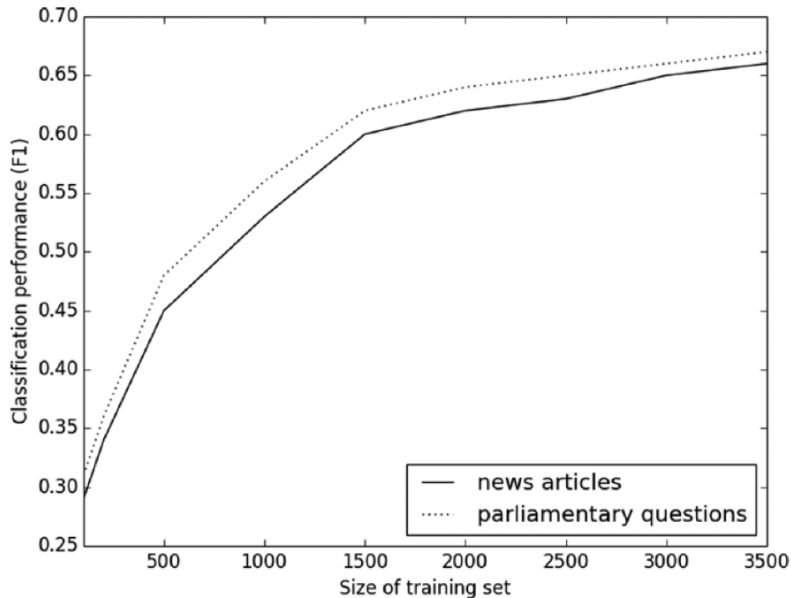
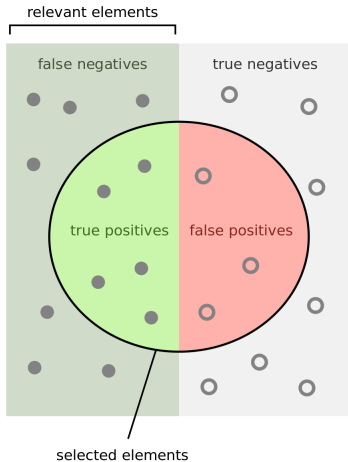


TABLE 1

## F1 Scores for SML-Based Issue Coding in News Articles and PQs

Issue	News Articles			PQs	
		All Words	Lead Only		All Words
Features	N	F1	F1	N	F1
Macroeconomics	413	.54	.63	172	.46
Civil rights and minority issues	327	.34	.28	192	.53
Health	444	.70	.71	520	.81
Agriculture	114	.72	.76	159	.66
Labor and employment	217	.43	.49	174	.58
Education	188	.79	.71	229	.78
Environment	152	.34	.44	237	.59
Energy	81	.35	.59	67	.66
Immigration and integration	150	.50	.57	239	.78
Transportation	416	.58	.67	306	.81
Law and crime	1198	.70	.69	685	.77
Social welfare	115	.33	.34	214	.54
Community development and housing	113	.45	.44	136	.72
Banking, finance, and commerce	622	.62	.67	188	.58
Defense	393	.59	.55	196	.71
Science, technology, and communication	426	.64	.59	57	.53
International affairs and foreign aid	1,106	.70	.64	352	.65
Government operations	1,301	.71	.72	276	.48
Other issue	3,322	.84	.80	360	.51
Total	11,089	.71	.68	4,759	.69


NOTE: The F1 score is equal to the harmonic mean of recall and precision. Recall is the fraction of relevant documents that are retrieved, and precision is the fraction of retrieved documents that are relevant.



## Some measures of accuracy

- Recall
- Precision
- $F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$
- AUC (Area under curve)  
[0, 1], 0.5 = random guessing

How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$


How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$


# What does this mean for our research?

It we have 2,000 documents with manually coded frames and topics. . .

- we can use them to train a SML classifier
- which can code an unlimited number of new documents
- with an acceptable accuracy

Some easier tasks even need only 500 training documents, see Hopkins, D. J., & King, G. (2010). A method of automated nonparametric content analysis for social science. *American Journal of Political Science*, 54(1), 229–247.



# An implementation

Let's say we have a list of tuples with movie reviews and their rating:

```
1 reviews=[("This is a great movie",1),("Bad movie",-1), ... ...]
```

And a second list with an identical structure:

```
1 test=[("Not that good",-1),("Nice film",1), ... ...]
```

Both are drawn from the same population, it is pure chance whether a specific review is on the one list or the other.

Based on an example from <http://blog.dataquest.io/blog/naive-bayes-movies/>

# Training a Naïve Bayes Classifier

```
1 from sklearn.naive_bayes import MultinomialNB
2 from sklearn.feature_extraction.text import CountVectorizer
3 from sklearn import metrics
4
5 # This is just an efficient way of computing word counts
6 vectorizer = CountVectorizer(stop_words='english')
7 train_features = vectorizer.fit_transform([r[0] for r in reviews])
8 test_features = vectorizer.transform([r[0] for r in test])
9
10 # Fit a naive bayes model to the training data.
11 nb = MultinomialNB()
12 nb.fit(train_features, [r[1] for r in reviews])
13
14 # Now we can use the model to predict classifications for our test
    features.
15 predictions = nb.predict(test_features)
16 actual=[r[1] for r in test]
17
18 # Compute the error.
19 fpr, tpr, thresholds = metrics.roc_curve(actual, predictions, pos_label
    =1)
20 print("Multinomial naive bayes AUC: {0}".format(metrics.auc(fpr, tpr)))
```

# And it works!

Using 50,000 IMDB movies that are classified as either negative or positive,

- I created a list with 25,000 training tuples and another one with 25,000 test tuples and
- trained a classifier
- that achieved an AUC of .82.

Dataset obtained from <http://ai.stanford.edu/~amaas/data/sentiment>, Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. *49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*

# Playing around with new data

```
1 newdata=vectorizer.transform(["What a crappy movie! It sucks!", "This is  
   awesome. I liked this movie a lot, fantastic actors","I would not  
   recomment it to anyone.", "Enjoyed it a lot"])  
2 predictions = nb.predict(newdata)  
3 print(predictions)
```

This returns, as you would expect and hope:

```
1 [-1  1 -1  1]
```

# But we can do even better

We can use different vectorizers and different classifiers.

# Different vectorizers

- CountVectorizer (=simple word counts)
- TfidfVectorizer (word counts (“term frequency”) weighted by number of documents in which the word occurs at all (“inverse document frequency”))
- additional options: stopwords, thresholds for minimum frequencies etc.

# Different classifiers

- Naïve Bayes
- Logistic Regression
- Support Vector Machine (SVM)
- ...

Let's look at the source code together and find out which setup performs best!



## **Unsupervised machine learning**

(something you already did in your Bachelor – no kidding.)

# Some terminology

## Supervised machine learning

You have a dataset with both predictor and outcome (dependent and independent variables) — a *labeled* dataset. Think of regression: You measured  $x_1$ ,  $x_2$ ,  $x_3$  and you want to predict  $y$ , which you also measured

## Unsupervised machine learning

You have no labels. (You did not measure  $y$ )

**Again, you already know some techniques to find out how  $x_1$ ,  $x_2, \dots x_i$  co-occur from other courses:**

- Principal Component Analysis (PCA)
- Cluster analysis
- ...

Enter **topic modeling with Latent Dirichlet Allocation (LDA)**

# LDA, what's that?

No mathematical details here, but the general idea

- There are  $k$  topics,  $T_1 \dots T_k$
- Each document  $D_i$  consists of a mixture of these topics, e.g. 80%  $T_1$ , 15%  $T_2$ , 0%  $T_3$ , ... 5%  $T_k$
- On the next level, each topic consists of a specific probability distribution of words
- Thus, based on the frequencies of words in  $D_i$ , one can infer its distribution of topics
- Note that LDA (like PCA) is a Bag-of-Words (BOW) approach

# Doing a LDA in Python

You can use gensim (Řehůřek & Sojka, 2010) for this.  
Furthermore, let us assume you have a list of lists of words (!)  
called texts:

```
1 articles=['The tax deficit is higher than expected. This said xxx ...',  
           'Germany won the World Cup. After a']  
2 texts=[art.split() for art in articles]
```

which looks like this:

```
1 [['The', 'tax', 'deficit', 'is', 'higher', 'than', 'expected.', 'This',  
   'said', 'xxx', '...'], ['Germany', 'won', 'the', 'World', 'Cup.', 'After', 'a']]
```

Řehůřek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pp. 45–50. Valletta, Malta: ELRA.

```
1 from gensim import corpora, models
2
3 NTOPICS = 100
4 LDAOUTPUTFILE="topicscores.tsv"
5
6 # Create a BOW representation of the texts
7 id2word = corpora.Dictionary(texts)
8 mm =[id2word.doc2bow(text) for text in texts]
9
10 # Train the LDA models.
11 lda = models.ldamodel.LdaModel(corpus=mm, id2word=id2word, num_topics=
    NTOPICS, alpha="auto")
12
13 # Print the topics.
14 for top in lda.print_topics(num_topics=NTOPICS, num_words=5):
15     print ("\n",top)
16
17 print ("\nFor further analysis, a dataset with the topic score for each
    document is saved to",LDAOUTPUTFILE)
18
19 scoresperdoc=lda.inference(mm)
20
21 with open(LDAOUTPUTFILE,"w",encoding="utf-8") as fo:
22     for row in scoresperdoc[0]:
23         fo.write("\t".join(["{:0.3f}".format(score) for score in row]))
24         fo.write("\n")
```

# Output: Topics (below) & topic scores (next slide)

```
1 0.069*fusie + 0.058*brussel + 0.045*europesecommissie + 0.036*europese +  
   0.023*overname  
2 0.109*bank + 0.066*britse + 0.041*regering + 0.035*financien + 0.033*  
   minister  
3 0.114*nederlandse + 0.106*nederland + 0.070*bedrijven + 0.042*rusland +  
   0.038*russische  
4 0.093*nederlandsespoorwegen + 0.074*den + 0.036*jaar + 0.029*onderzoek +  
   0.027*raad  
5 0.099*banen + 0.045*jaar + 0.045*productie + 0.036*ton + 0.029*aantal  
6 0.041*grote + 0.038*bedrijven + 0.027*ondernemers + 0.023*goed + 0.015*  
   jaar  
7 0.108*werknemers + 0.037*jongeren + 0.035*werkgevers + 0.029*jaar +  
   0.025*werk  
8 0.171*bank + 0.122* + 0.041*klanten + 0.035*verzekeraar + 0.028*euro  
9 0.162*banken + 0.055*bank + 0.039*centrale + 0.027*leningen + 0.024*  
   financiële  
10 0.052*post + 0.042*media + 0.038*nieuwe + 0.034*netwerk + 0.025*  
    personeel  
11 ...
```

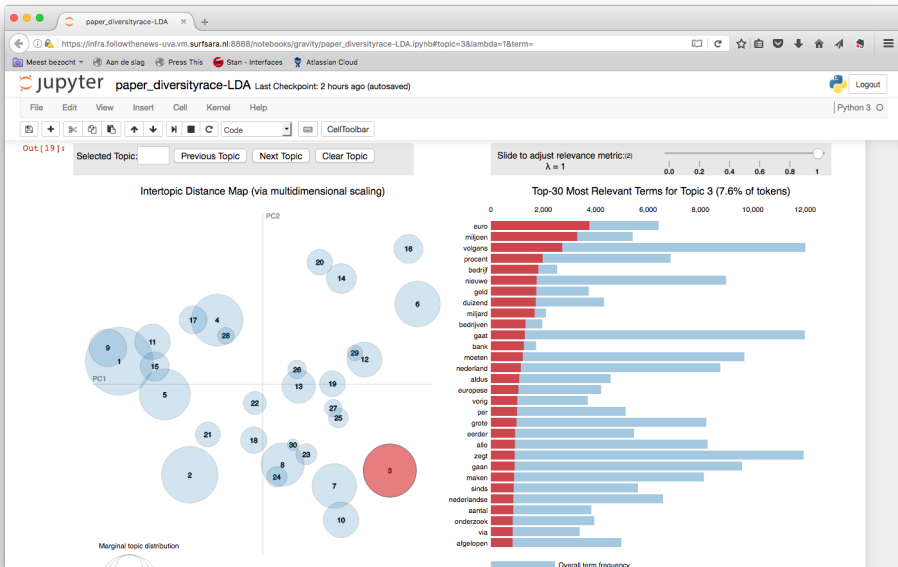
Data Editor (Browse) - topicscores.data													
topic4[2]		.019											
source2	firstwords	polarity	subjectivity	pubdate_day	pubdate_mo-h	pubdate_year	pubdate_da-k	topic1	topic2	topic3	topic4	topic5	
1	nrc handelsblad	palingsound schinke	-.0086207	.6069971	31	12	2011	zaterdag	.018	.019	3.587	.019	.019
2	nrc handelsblad	groep investeerders	-.1041667	.3129194	31	12	2011	zaterdag	.018	.019	.019	.019	.019
3	nrc handelsblad	abnamro debacles ij	.0082292	.4895443	31	12	2011	zaterdag	.018	27.71	.019	.019	.019
4	nrc handelsblad	abnamro financi' le	-.0179617	.5706419	31	12	2011	zaterdag	.018	15.1	.019	2.646	.019
5	nrc handelsblad	crisis verhouding k	.0758049	.5448864	31	12	2011	zaterdag	.018	.019	9.008	.019	.019
6	nrc handelsblad	snel vakantie vrije	-.016315	.5118008	31	12	2011	zaterdag	.018	.019	.019	.019	.019
7	nrc handelsblad	herinnering doos le	.18875	.6200333	31	12	2011	zaterdag	.018	.019	.019	.019	.019
8	nrc handelsblad	hackers publiceren	.1454545	.4545455	31	12	2011	zaterdag	.018	.019	.019	.019	.019
9	nrc handelsblad	waterballet nontevi	-.2333333	.4333333	31	12	2011	zaterdag	.018	.019	.019	.019	.019
10	nrc handelsblad	bouw dupe ambities	.0925417	.5939167	5	11	2010	vrijdag	.018	.019	.078	2.442	.019
11	nrc handelsblad	eindelijk wint nuh	.1755093	.48125	5	11	2010	vrijdag	.018	.019	8.302	.019	.019
12	nrc handelsblad	oud nieuws tv bbct	.02	.4322222	5	11	2010	vrijdag	.018	10.053	.019	.019	.019
13	nrc handelsblad	tag hyves krantenb	.0425203	.5420412	5	11	2010	vrijdag	.018	.019	.019	.019	.019
14	nrc handelsblad	getuigenis rechter	.0858929	.5770833	5	11	2010	vrijdag	.018	.019	.019	11.621	.019
15	nrc handelsblad	akzonobel philips g	.0220455	.4381818	5	11	2010	vrijdag	.018	.019	.019	.019	.019
16	nrc handelsblad	mondiaal kritiek be	-.038172	.3894624	5	11	2010	vrijdag	.018	19.957	.019	.019	.019
17	nrc handelsblad	export diamant fiat	.0628571	.4438895	5	11	2010	vrijdag	.018	4.745	.019	.019	.019
18	nrc handelsblad	canada bod potash r	.0252924	.4795322	5	11	2010	vrijdag	.018	26.741	.019	.019	.019
19	nrc handelsblad	zwakke bouwsector c	.0171	.4736333	14	3	2009	NA	.018	.019	.019	.019	4.806
20	nrc handelsblad	pensioenconflict wa	.028114	.4636842	14	3	2009	NA	.018	.019	.019	.019	.019
21	nrc handelsblad	rechter allin loon	.1318182	.3939394	14	3	2009	NA	.018	.019	.019	.019	.019
22	nrc handelsblad	bad bank remedie da	.0891026	.550641	14	3	2009	NA	.018	10.235	.019	.019	.019
23	nrc handelsblad	bescheiden salaris	-.075	.56	14	3	2009	NA	.018	.019	.019	.019	.019
24	nrc handelsblad	generalmotors autos	.0138889	.4388889	14	3	2009	NA	.018	.019	.019	.019	.019
25	nrc handelsblad	rusland rozen tuinb	.0314141	.5643051	14	3	2009	NA	.018	.019	24.595	.019	.019
26	nrc handelsblad	cynisae oplossing k	.0100833	.6511667	14	3	2009	NA	.018	.019	.019	.019	.019
27	nrc handelsblad	the good bed ugly l	.0265504	.5298449	13	3	2009	NA	.018	.019	.019	.019	.019
28	nrc handelsblad	kerk stroom nietswe	-.0087719	.6149123	13	3	2009	NA	.018	.019	.019	.019	.019
29	nrc handelsblad	kerk stroom goud ac	0	0	13	3	2009	NA	.018	.019	.019	.019	.019
30	nrc handelsblad	supersnelle koeknpe	0	0	13	3	2009	NA	.018	.019	.019	.019	.019
31	nrc handelsblad	dalailama chinese e	0	0	13	3	2009	NA	.018	.019	.019	.019	.019
32	nrc handelsblad	bezuinigen hulpgeld	.0894192	.4560606	13	3	2009	NA	.018	.019	.019	.019	.019
33	nrc handelsblad	vaders arbeidsethos	.0160985	.5575758	13	3	2009	NA	.018	.019	.019	.019	.019
34	nrc handelsblad	varkens lux winnaar	.040073	.6218254	4	10	2008	NA	.018	.019	.019	.019	.019
35	nrc handelsblad	liberale kinderopva	.1179095	.5297055	4	10	2008	NA	.018	.019	.019	.019	1.83
36	nrc handelsblad	banken verzinsels k	.068521	.6308389	4	10	2008	NA	8.232	.019	.019	.019	.019
37	nrc handelsblad	rabobanktopman bert	0	0	4	10	2008	NA	.018	.019	.019	.019	.019
38	nrc handelsblad	kinderopvang bril v	0	0	4	10	2008	NA	.018	.019	.019	.019	.019
39	nrc handelsblad	tassen gevoel verli	0	0	4	10	2008	NA	.018	.019	.019	.019	.019
40	nrc handelsblad	abnamro winklend p	.0876761	.62277	4	10	2008	NA	.018	.019	6.904	.019	5.511
41	nrc handelsblad	abnamro belgi' mole	.0439506	.4976852	4	10	2008	NA	.018	.019	.019	.019	.019
42	nrc handelsblad	abnamro handen deut	.1838401	.5264302	4	10	2008	NA	.018	.019	1.854	.019	.019
43	nrc handelsblad	abnamro fortis bank	.0842391	.494058	4	10	2008	NA	4.939	.019	14.39	.019	.019
44	nrc handelsblad	abnamro fortis spra	.0540715	.6290807	4	10	2008	NA	.018	.019	.019	.019	.019
45	nrc handelsblad	abnamro fortis jaar	.0297297	.4960135	4	10	2008	NA	.018	11.041	.019	.019	.019
46	nrc handelsblad	abnamro nederland s	.1006944	.6830555	4	10	2008	NA	.018	.019	.019	.019	.019
47	nrc handelsblad	abnamro belgi' mole	.0405952	.5804464	4	10	2008	NA	.018	.019	.019	.019	.019
48	nrc handelsblad	arbeidsmarkt vs sle	.0166667	.4	4	10	2008	NA	7.103	.019	.019	.019	12.682



# Explore topics interactively with pyLDAvis

If you install pyLDAvis, you can interactively explore the data by simply feeding the GenSim-model to pyLDAvis!

```
1 import pyLDAvis
2 import pyLDAvis.gensim
3 vis_data = pyLDAvis.gensim.prepare(lda,mm,id2word)
4 pyLDAvis.display(vis_data)
```



## Exercise