Code ▾

Exploring the potential and suitability of using natural language processing (NLP) to systematize analysis of SenseMaker narratives

Anna Hanchar, PhD

The Data Atelier
(mailto:#)anna.h@thedataatelier.com (mailto:anna.h@thedataatelier.com)

26 October 2017

Background reading

A general introduction to natural language processing (NLP) in development context:

- Our recent paper “Data Innovation for International Development: An overview of natural language processing for qualitative data anlaysis” in proceedings of the Frontiers and Advances of Data Science IEEE Conference. Available from ArXiv https://arxiv.org/abs/1709.05563 (https://arxiv.org/abs/1709.05563)

Good, introductory overview papers about NLP and its application:

- Lucas et al., “Computer-Assisted Text Analysis for Comparative Politics”, Political Analysis, 2015, 23: 254-277. Available here: http://christopherlucas.org/files/PDFs/text_comp_politics.pdf (http://christopherlucas.org/files/PDFs/text_comp_politics.pdf)

- Grimmer and Stewart, “Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts”, Political Analysis, 2013. Available here: http://web.stanford.edu/~jgrimmer/tad2.pdf (http://web.stanford.edu/~jgrimmer/tad2.pdf)

Introductory tutorial on “quanteda” package: https://cran.r-project.org/web/packages/quanteda/vignettes/quickstart.html (https://cran.r-project.org/web/packages/quanteda/vignettes/quickstart.html)

Introduction to R Markdown and R Notebooks environment (used for transparent and reproducible research): http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com) and http://rmarkdown.rstudio.com/r_notebooks.html (http://rmarkdown.rstudio.com/r_notebooks.html)

An excellent introduction to R and data science can be found here:

- Garrett Grolemund and Hadley Wickham (2016) *R for Data Science*, O’Reilly Media. Note: Online version is available from the authors’ page here (http://r4ds.had.co.nz/index.html).

Step 1: Loading packages and data

Load packages:

Hide

```
library(readtext)
library(quanteda)
library(dplyr)
library(stringr)
library(ggplot2)
library(haven)
library(readxl)
library(magrittr)
library(stm)
library(readr)
```

Load data (from original CSV file), specifying which column contains stories. Function “readtext” from the eponymous package simplifies the loading of text data (see https://github.com/kbenoit/readtext (https://github.com/kbenoit/readtext)).

Hide

```
moldova_foi <- readtext("foimoldova2015_Standard.csv",text_field = "Your experience
")
```

Step 2: Preparing data

Step 2.1: Creating ‘corpus’

Create a “corpus” object for analysis (see https://en.wikipedia.org/wiki/Text_corpus (https://en.wikipedia.org/wiki/Text_corpus) for general introduction to the concept):

Hide

```
moldova_corpus <- corpus(moldova_foi)
```

Step 2.2: Pre-processing data

Transform the corpus into separate words (tokens) and perform basic pre-processing:

Hide

```
tok <- tokens(moldova_corpus, what = "word",
              remove_punct = TRUE,
              remove_symbols = TRUE,
              remove_numbers = TRUE,
              remove_twitter = TRUE,
              remove_url = TRUE,
              remove_hyphens = TRUE,
              verbose = TRUE)
```

```
Starting tokenization...
...tokenizing 1 of 1 blocks
...removing URLs
...serializing tokens 3723 unique types
...total elapsed:  0.0940000000000509 seconds.
Finished tokenizing and cleaning 519 texts.
```

This allows to remove any digits and punctuation, that may be part of tokens through mistakes in text conversion and input; to remove any tokens containing less than three characters long (picks up some other mistakes and typos); to convert everything into lower case. A “regular expression” (regex) function is used here (see https://en.wikipedia.org/wiki/Regular_expression (https://en.wikipedia.org/wiki/Regular_expression))

Hide

```
tok.m <- tokens_select(tok, c("[\\d-]", "^.{1,2}$", "[[:punct:]]"),
                       selection = "remove",
                    valuetype="regex", verbose = TRUE)
```

```
removed 160 features
```

Hide

```
tok.r <- tokens_tolower(tok.m)
```

To remove “stop words” (not carrying functional meaning) “smart” list is used (see http://docs.quanteda.io/reference/stopwords.html (http://docs.quanteda.io/reference/stopwords.html))

Hide

```
toks2 <- tokens_remove(tok.r, stopwords("SMART"), padding = TRUE,
                       verbose = TRUE)
```

```
removed 291 features
```

If an automatic translation systems (e.g. Google Translate) is used, not all words can be translated (e.g. proper names, places). Such non-Enlish tokens can be removed using regex functions.

Hide

```
toks2 <- tokens_select(toks2, "[^ -~]", selection = "remove",
                       valuetype = "regex", case_insensitive = TRUE, padding = TRUE
,
                       verbose = TRUE)
```

```
removed 60 features
```

Step 2.3: Creating document feature matrix (DFM)

A document feature matrix (aka document term matrix) or DFM is a fundamental input into natural language processing (see https://en.wikipedia.org/wiki/Document-term_matrix (https://en.wikipedia.org/wiki/Document-term_matrix)). We construct a DFM from tokens.

Stemming tokens, removing “padding” (white space where tokens e.g. non-English words were removed previously but space added for analysis of phrases).

Hide

```
dfm <- dfm(toks2, stem=TRUE, verbose = TRUE, remove = "")
```

```
Creating a dfm from a tokens input...
   ... lowercasing
   ... found 519 documents, 2,966 features
   ... removed 1 feature
   ... stemming features (English)
, trimmed 660 feature variants
   ... created a 519 x 2,306 sparse dfm
   ... complete.
Elapsed time: 0.139 seconds.
```

Trimming the DFM: dropping tokens appearing less than two times, mainly to catch typos and text conversion mistakes. The logic is that if a token is used only once in all narratives, that could be a feature that does not distinuish well between documents. Alternatively that can be a spelling mistake or typo.

Hide

```
dfm.trim <- dfm_trim(dfm, min_count = 2)
```

Top 50 tokens that appear most frequently in our DFM. This can be a diagnosis if there are some erroneous features appearing. For example, if non-English words still appear despite corresponding pre-processing. Changing “decreasing = FALSE” results in listing least frequent features in DFM.

Hide

```
topfeatures(dfm.trim, n = 50, decreasing = TRUE)
```

```
   peopl    worri  countri   situat     work  moldova   recent    polit   villag 
live     year 
     110       87       80       70       70       51       49       47       44 
41       41 
encourag    money      day     time children  increas    young  problem   famili 
chang   salari 
      39       38       37       36       34       32       29       28       28 
28       28 
 protest     made   govern   action    organ    state    water     good   school 
road     event 
      27       27       26       26       26       26       26       25       25 
25       25 
 concern   person   street     citi      pay    price  citizen  student    place 
minist     hous 
      25       25       24       23       23       23       23       22       22 
21       21 
    home     make  pension     life   social     feel 
      21       21       21       21       20       20 
```

Total number of tokens in DFM, shows the size of the DFM for analysis.

Hide

```
nfeature(dfm.trim)
```

```
[1] 1012
```

Step 3: Data Exploration

3.1 Frequency analysis

Hide

```
freq <- textstat_frequency(dfm.trim)
freq
```

feature	frequency	rank	docfreq
<chr>	<dbl>	<int>	<dbl>
peopl	110	1	88
worri	87	2	81
countri	80	3	62
situat	70	4	61
work	70	5	54

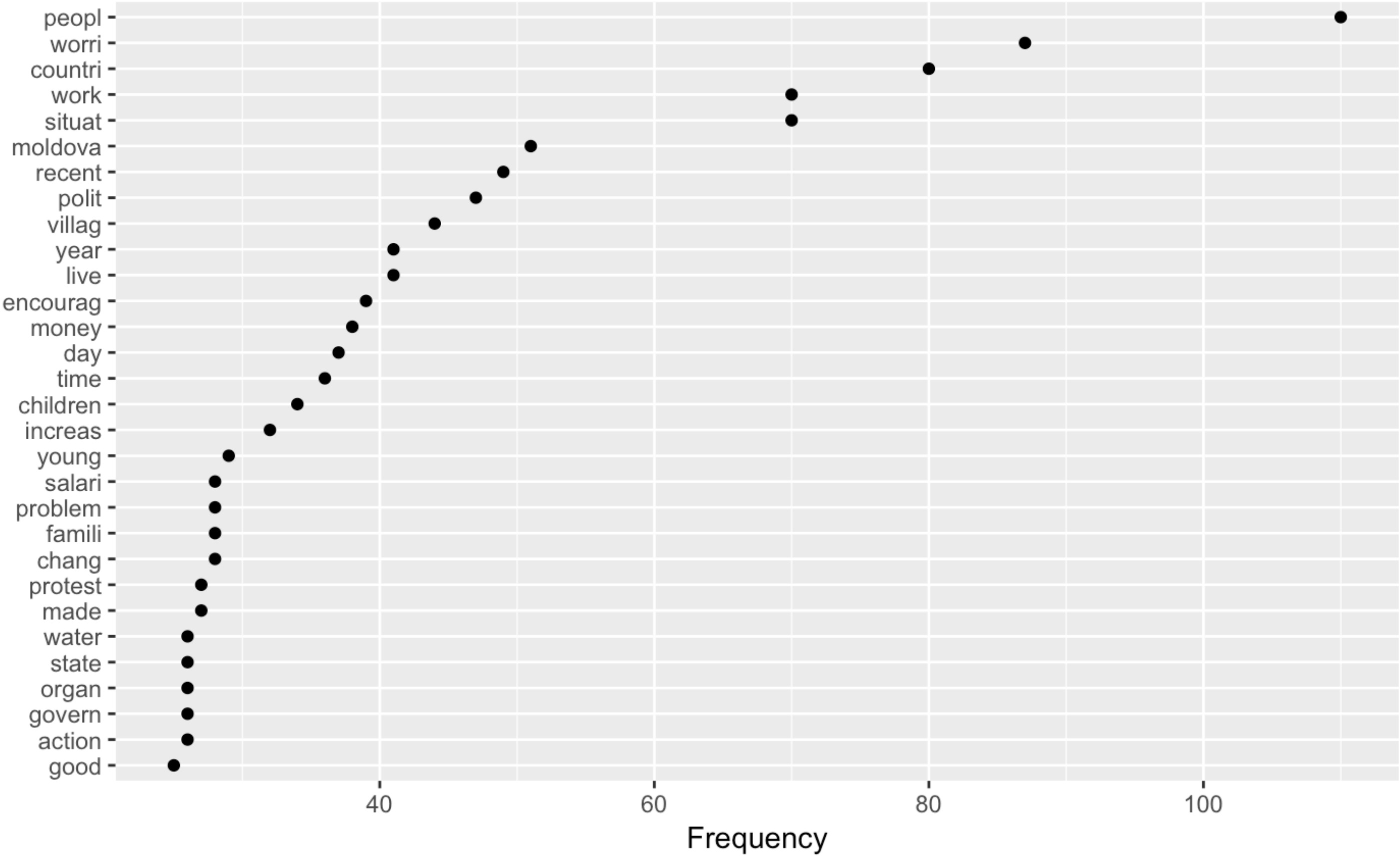
moldova	51	6	42
recent	49	7	49
polit	47	8	40
villag	44	9	29
live	41	10	36

1-10 of 1,012 rows Previous **1** 2 3 4 5 6 ... 100 Next

Visualising frequencies with a plot of 20 most frequent words:

Hide

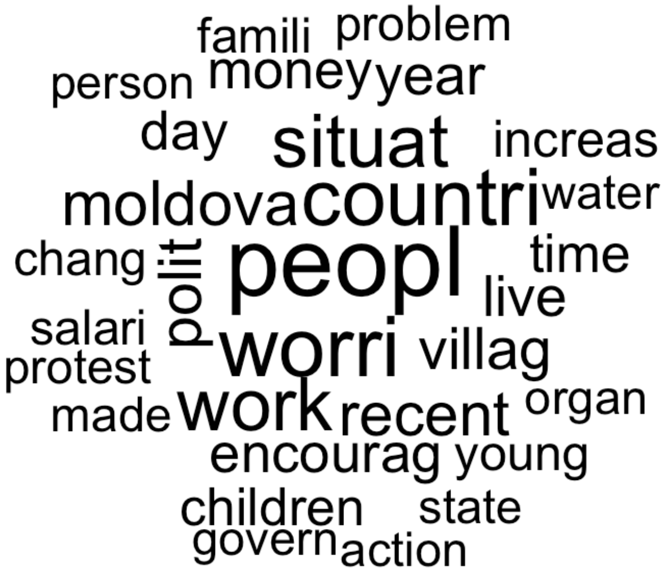
```
ggplot(freq[1:30, ], aes(x = reorder(feature, frequency), y = frequency)) +
    geom_point() +
    coord_flip() +
    labs(x = NULL, y = "Frequency")
```



Visualising frequencies with a plot of 20 most frequent words (traditional word cloud)

Hide

```
textplot_wordcloud(dfm.trim, max.words=30, scale=c(3,1), random.order=FALSE)
```



3.2 Keyness analysis

Exploring which key terms appear in the corpus more frequently than by chance (see https://en.wikipedia.org/wiki/Keyword_(linguistics) (https://en.wikipedia.org/wiki/Keyword_(linguistics)); http://docs.quanteda.io/reference/textstat_keyness.html (http://docs.quanteda.io/reference/textstat_keyness.html))

Assessing keyness between females and males: first, “target” document needs to be identified (“target” refers to the gender variable in the original CSV dataset); second, if keyness among ‘male’ respondents is looked at, ‘female’ respondents serve as baseline. The outputs are sorted in descending order by the association measure (chi2 here).

Hide

```
keyness_gender <- textstat_keyness(dfm.trim,
                                   docvars(moldova_corpus, "DQ2.Gender") == "male",
                                   sort = TRUE)
keyness_gender
```

	chi2 <dbl>	p <dbl>	n_target <dbl>	n_reference <dbl>
interest	1.204246e+01	0.0005200229	9	1
prime	1.065881e+01	0.0010954762	12	5
nation	9.203782e+00	0.0024151562	6	0
snow	8.283959e+00	0.0039996831	7	1

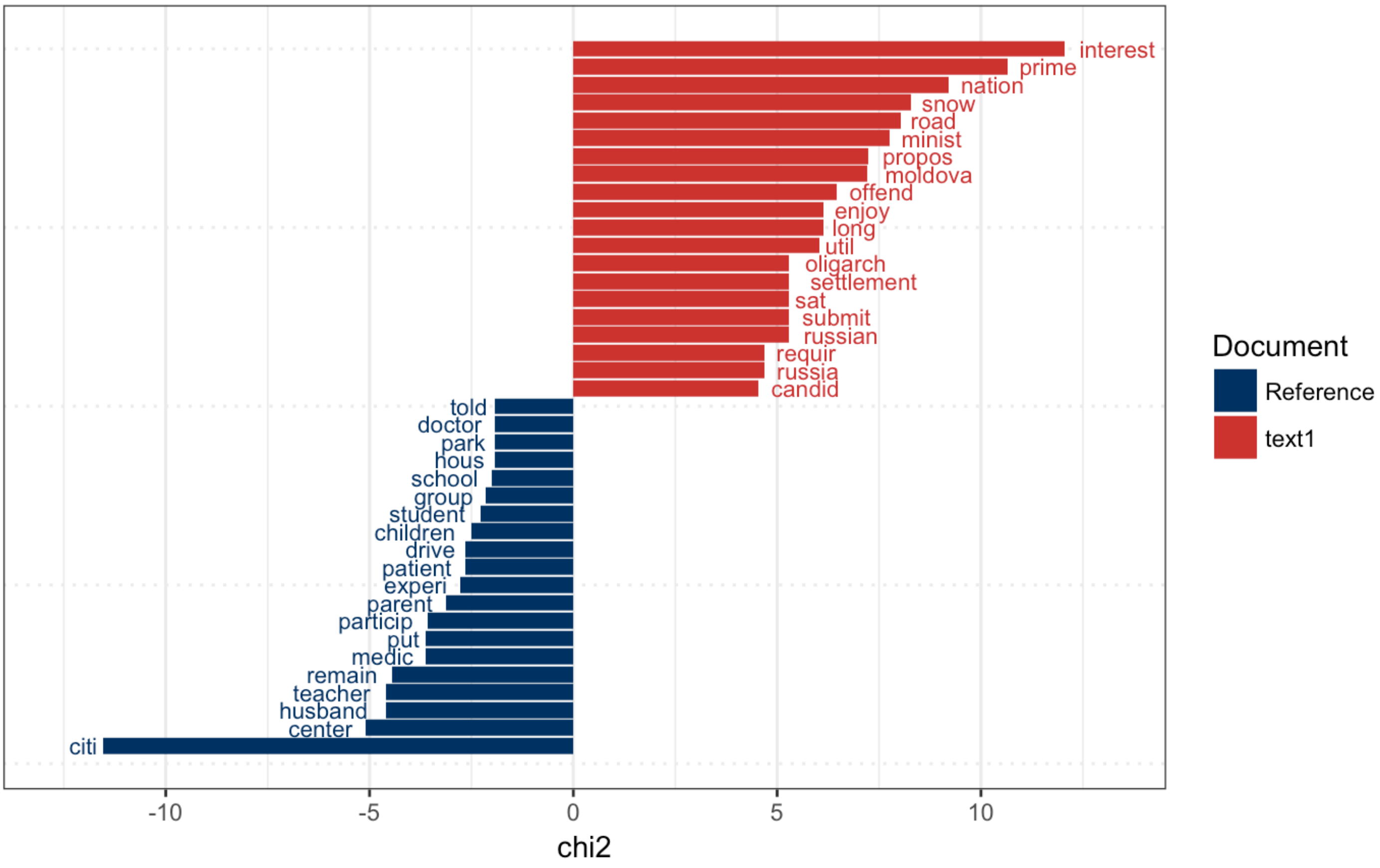
road	8.044242e+00	0.0045648526	15	10
minist	7.751277e+00	0.0053674570	13	8
propos	7.236749e+00	0.0071426191	5	0
moldova	7.221579e+00	0.0072032329	26	25
offend	6.459938e+00	0.0110333484	6	1
enjoy	6.140545e+00	0.0132116606	7	2

1-10 of 1,012 rows Previous **1** 2 3 4 5 6 ... 100 Next

To visualise keyness between males and females (see http://docs.quanteda.io/reference/textplot_keyness.html (http://docs.quanteda.io/reference/textplot_keyness.html)). “Reference” refers to keywords usage by female respondents; “text1” - by male respondents.

Hide

```
textplot_keyness(keyness_gender, show_reference = TRUE, n = 20L, min_count = 2L)
```



Example of assessing keyness between rural and urban areas:

Hide

```
keyness_urban <- textstat_keyness(dfm.trim,
                                   docvars(moldova_corpus, "DQ5.Live") == "urban ar
ea",
                                   sort = TRUE)
keyness_urban
```

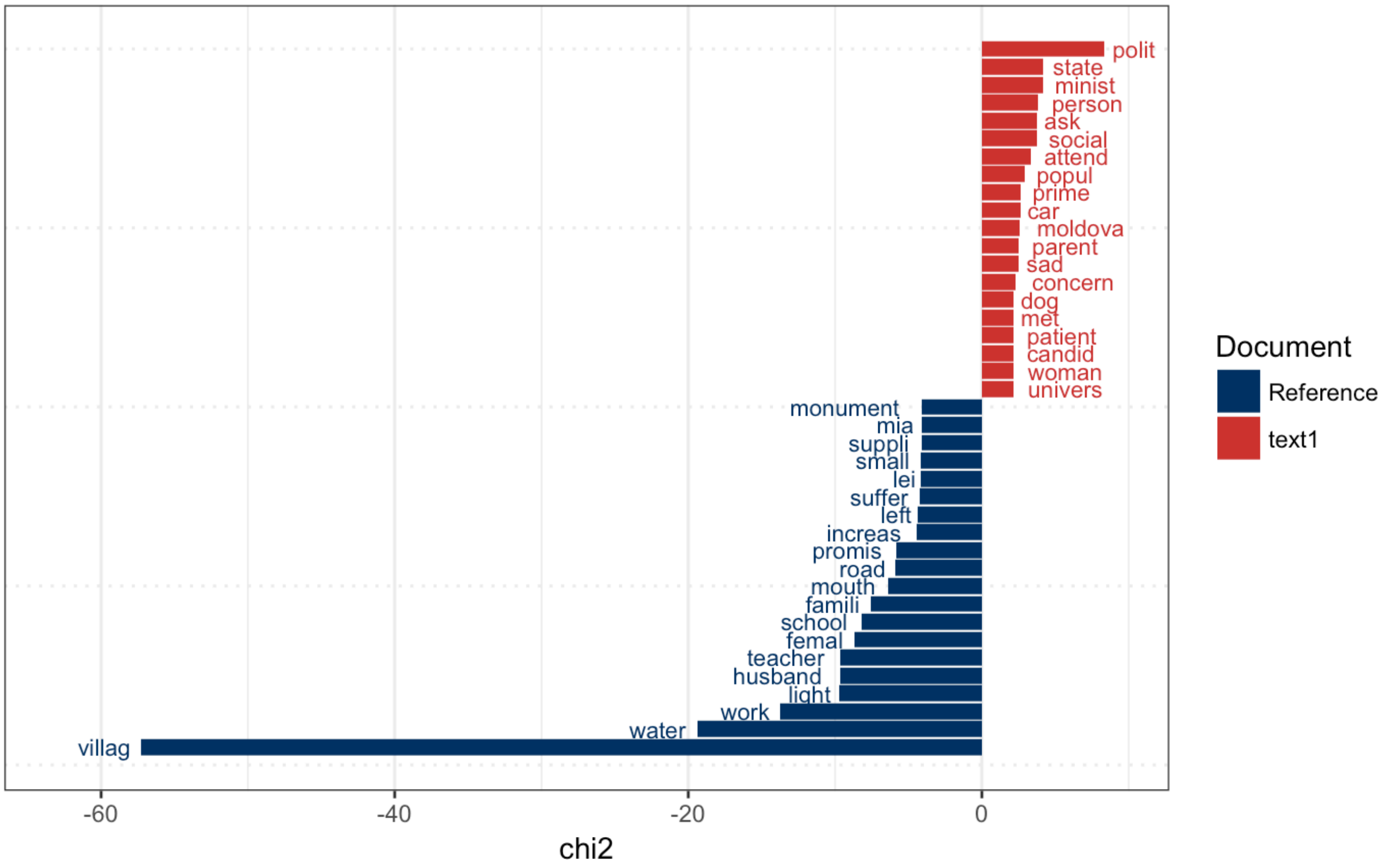
	chi2 <dbl>	p <dbl>	n_target <dbl>	n_reference <dbl>
polit	8.325647e+00	0.003908921	42	5
state	4.169227e+00	0.041164570	23	3
minist	4.147235e+00	0.041702563	19	2
person	3.808376e+00	0.050996883	22	3
ask	3.780567e+00	0.051851083	12	0
social	3.765959e+00	0.052305856	18	2
attend	3.363383e+00	0.066661001	11	0
popul	2.948147e+00	0.085976091	10	0
prime	2.655086e+00	0.103219146	15	2
car	2.655086e+00	0.103219146	15	2

1-10 of 1,012 rows   Previous **1** 2 3 4 5 6 ... 100 Next

To visualise keyness between urban and rural areas:

Hide

```
textplot_keyness(keyness_urban, show_reference = TRUE, n = 20L, min_count = 2L)
```



3.3 Word and document similarity

A basis of NLP is that words populate the vector space (see https://en.wikipedia.org/wiki/Vector_space_model (https://en.wikipedia.org/wiki/Vector_space_model)) and simple geometry can be used to assess similarities between words and equivalently distances.

For example, if the word “work” is of interest, words people use that are the closest to “work” can be identified - one way to think about it is in terms of semantic similarity. Here the cosine similarity measure (and Euclidean distance below) is used (see http://docs.quanteda.io/reference/textstat_simil.html (http://docs.quanteda.io/reference/textstat_simil.html))

Hide

```
work_simil <- textstat_simil(dfm.trim, "work", method = "cosine", margin = "feature
s")
as.list(work_simil, n = 15)
```

```
$work
   employ    depart    reduct    retire   appreci       job    worker    social    c
ompani     field 
0.3600411 0.3402069 0.2886751 0.2777778 0.2721655 0.2608360 0.2566001 0.2364027 0.2
236068 0.2222222 
     hire   minimum     retir     engag     month 
0.2222222 0.2222222 0.2182179 0.2151657 0.2100420 
```

The words respondents use that are the distant to the word “work”:

Hide

```
work_dist <- textstat_dist(dfm.trim, "work", method = "euclidean", margin = "featur
es")
as.list(work_dist, n = 15)
```

```
$work
   peopl  countri   villag    worri   situat    water  moldova    polit encourag 
money     road 
15.16575 14.96663 13.71131 13.45362 13.34166 13.19091 13.07670 13.07670 12.44990 12
.32883 12.28821 
  recent    mayor   offend  student 
12.12436 12.04159 12.04159 12.00000 
```

3.4 Keyword in context (KWIC)

A useful way to see the context of a keyword is to use KWIC (see http://docs.quanteda.io/reference/kwic.html (http://docs.quanteda.io/reference/kwic.html)) .

For example, if the focus is on the words “work”, “salary”, and “job”, a small dictionary with these three words can be created (see http://docs.quanteda.io/reference/dictionary.html (http://docs.quanteda.io/reference/dictionary.html)). Here, specifying valuetypes when creating dictionary (“work“*,”salar*”, “job*“) allows to pick up any versions of tokens.

Hide

```
dict <- dictionary(list(us = c("work*", "salar*", "job*")))
phrase(dict)
```

```
[[1]]
[1] "work*"

[[2]]
[1] "salar*"

[[3]]
[1] "job*"
```

Hide

```
kwic_work <- kwic(moldova_corpus, dict, window = 5, valuetype = "glob")
head(kwic_work, n =10)
```

```

                                                                                        
  [text1, 29]     aroused the interest of the |  work   | and the future developmen
t of
  [text2, 76]          turned out he wants to |  work   | but too many people are
  [text18, 3]                       I receive | salary  | in lei. And the
 [text18, 35]   in relation to currencies and | salary  | remains the same. If
  [text26, 6]      nehochu go abroad but this |  work   | is not very worried
 [text40, 20]                 . Just a lot of |  work   | yard by the end of
  [text47, 2]                        Changing |  jobs   | raised concerns me if it
 [text57, 29] retirees others were allowed to |  work   | release; workers with hig
h
 [text57, 32]        allowed to work release; | workers | with high stint working i
n
 [text57, 36]       ; workers with high stint | working | in a field were moved
```

Frequency analysis discussed earlier can also be used here to see the most frequent words appearing in the context of “work”. To do so, the tokens are cleaned and a DFM of the “work”-related tokens is created (but removing our work-related dictionary terms):

Hide

```
tok.work <- tokens(as.tokens(kwic_work), what = "word",
              remove_punct = TRUE,
              remove_symbols = TRUE,
              remove_numbers = TRUE,
              remove_twitter = TRUE,
              remove_url = TRUE,
              remove_hyphens = TRUE,
              verbose = TRUE)
dfm.work <- dfm(tok.work,
                tolower = TRUE,
           remove= c(stopwords("SMART"), "work*", "job*", "salary*"),
           verbose = TRUE)
```

```
Creating a dfm from a tokens input...
   ... lowercasing
   ... found 130 documents, 438 features
   ... removed 133 features
   ... created a 130 x 305 sparse dfm
   ... complete.
Elapsed time: 0.044 seconds.
```

To create frequencies:

Hide

```
freq_work <- textstat_frequency(dfm.work)
freq_work
```

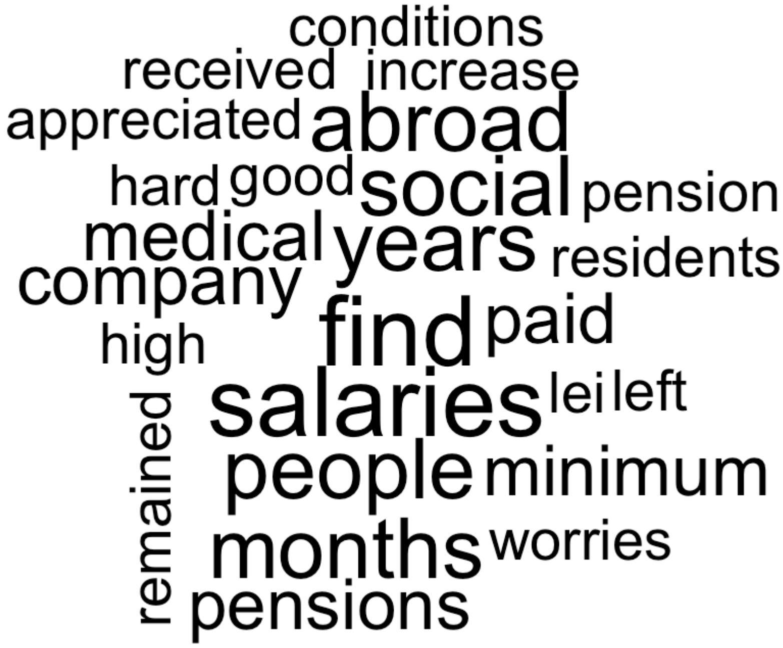
feature	frequency	rank	docfreq
<chr>	<dbl>	<int>	<dbl>
find	6	1	5
salaries	6	2	6
years	5	3	5
people	5	4	5
social	5	5	5
months	5	6	5
abroad	5	7	5
paid	4	8	4
medical	4	9	4
company	4	10	4

1-10 of 305 rows Previous **1** 2 3 4 5 6 ... 31 Next

To visulaise frequencies with a word cloud:

Hide

```
textplot_wordcloud(dfm.work, scale=c(3,.5), random.order=FALSE)
```



3.5 Structural topic modeling

Structural topic model or STM (Roberts et al., 2015) is a type of probabilistic topic models (Blei et al. 2003) that allows to assess the effect of covariates (see http://www.structuraltopicmodel.com (http://www.structuraltopicmodel.com); for an introduction and a nice overview of topic modeling see http://www.cs.columbia.edu/~blei/papers/Blei2012.pdf (http://www.cs.columbia.edu/~blei/papers/Blei2012.pdf)).

For example, STM allows to see whether gender has an effect on the content of narratives (topic prevalence).

First, we convert a DFM into a format that is used by the “stm” package:

Hide

```
stm.dfm <- convert(dfm.trim, to = "stm",  docvars = docvars(moldova_corpus))
```

```
Dropped empty document(s): text19, text20, text141, text153, text250, text262
```

3.5.1 Searching for optimal number of topics

One key input into the topic modeling algorithm is specifying the number of topics the algorithm needs to uncover in the corpus. This can be done with a manual input, using human expert judgement to determine the number of topics. Alternatively, this can be done by focusing on semantic coherence (see Mimno et al., 2011) and exclusivity (see Bischof and Airoldi, 2012) measures. Highly frequent words in a given topic that don’t appear too often in other topics are said to make that topic exclusive. Cohesive and exclusive topics are more semantically useful.

The steps are: generate a set of candidate models, here ranging between 3 and 10; plot exclusivity and semantic coherence; and choose the optimal number of topics as a balance between these two measures (see Roberts et al., 2015).

Hide

```
search <- searchK(stm.dfm$documents, stm.dfm$vocab,
                  K = c(3:10),
                  data = stm.dfm$meta)
```

Plot the exclusivity and semantic coherence (numbers closer to zero indicate higher coherence), and select a model on the semantic coherence-exclusivity “frontier” (where no model strictly dominates another in terms of semantic coherence and exclusivity).

Hide

```
par(mar=c(5,4,4,5)+.1)
plot(search$results$K,search$results$exclus,type="l",col="red",
     xlab="Number of topics", ylab="Exclusivity")
axis(side=1,at=seq(0,50,5))
```

Hide

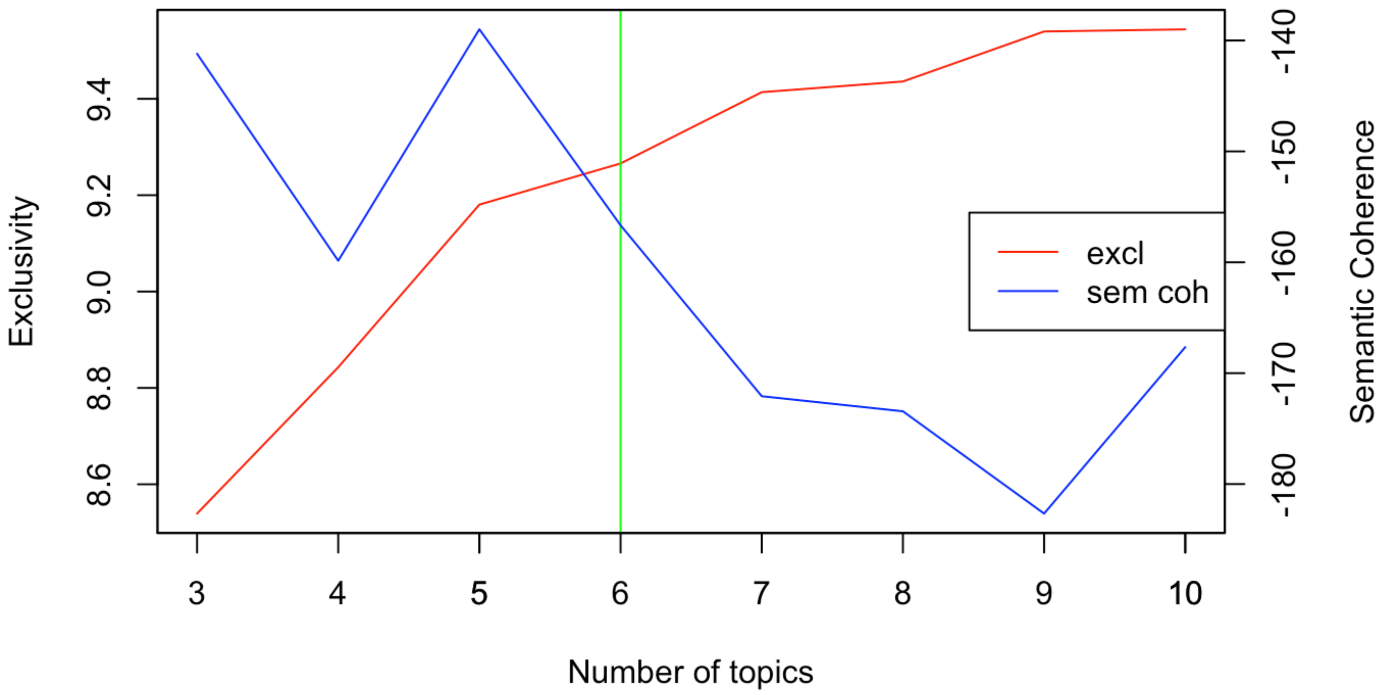
```
abline(v=6, col="green")
par(new=TRUE)
```

Hide

```
plot(search$results$K, search$results$semcoh,
     type="l",col="blue",xaxt="n",yaxt="n",xlab="",ylab="")
axis(4)
```

Hide

```
mtext("Semantic Coherence",side=4,line=3)
legend("right",col=c("red","blue"),lty=1,legend=c("excl","sem coh"))
```



The model with six topics is selected for our analysis (highlighted with vertical line). There's a drop in semantic coherence after $k = 6$. The model is also estimated with gender and rural/urban indicator. Both covariates are categorical so they have to come into the model as factor variables.

Hide

```
topics6 <- stm(stm.dfm$documents, stm.dfm$vocab,
               prevalence = ~ factor(DQ5.Live) + factor(DQ2.Gender) ,
               data = stm.dfm$meta,
               K = 6, init.type = "Spectral")
```

3.5.2 Exploring words associated with each topic

One way to summarize topics is to combine term frequency and exclusivity to that topic into a univariate summary statistic.

In STM package this is implemented as FREX (see Bischof and Airoldi, 2012 and Airoldi and Bischof, 2016). The logic behind this measure is that both frequency and exclusivity are important factors in determining semantic content of a word and form a two dimensional summary of topical content. FREX is the geometric average of frequency and exclusivity and can be viewed as a univariate measure of topical importance.

STM authors suggest that nonexclusive words are less likely to carry topic-specific content, while infrequent words occur too rarely to form the semantic core of a topic. FREX is therefore combining information from the most frequent words in the corpus that are also likely to have been generated from the topic of interest to summarize its content. In practice, topic quality is usually evaluated by highest probability words.

To look at both FREX and highest probability words:

Hide

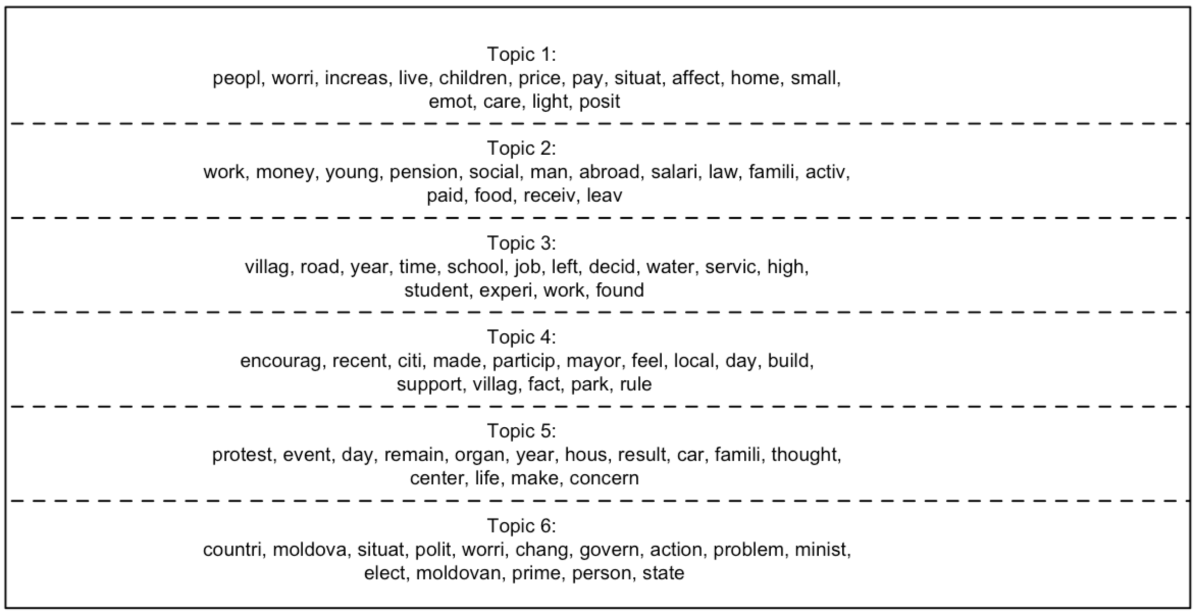
```
labelTopics(topics6)
```

```
Topic 1 Top Words:
     Highest Prob: peopl, worri, increas, live, children, price, pay
     FREX: peopl, price, affect, care, util, product, mother
     Lift: carpet, casino, requir, sum, affect, price, accid
     Score: peopl, price, increas, femal, small, employe, affect
Topic 2 Top Words:
     Highest Prob: work, money, young, pension, social, man, abroad
     FREX: money, pension, social, man, abroad, law, activ
     Lift: activ, law, pension, russia, social, visit, man
     Score: work, money, pension, social, young, paid, man
Topic 3 Top Words:
     Highest Prob: villag, road, year, time, school, job, left
     FREX: road, job, decid, experi, found, husband, learn
     Lift: abandon, assist, assum, balti, basi, block, brutal
     Score: road, job, employ, villag, colleg, water, wait
Topic 4 Top Words:
     Highest Prob: encourag, recent, citi, made, particip, mayor, feel
     FREX: particip, local, build, fact, park, rule, apart
     Lift: alley, build, construct, creativ, discuss, donat, inherit
     Score: snow, project, encourag, citi, build, essay, republ
Topic 5 Top Words:
     Highest Prob: protest, event, day, remain, organ, year, hous
     FREX: protest, event, remain, result, center, indiffer, room
     Lift: adult, alexand, approach, area, assembl, bag, basement
     Score: sister, offend, dog, protest, aggress, event, water
Topic 6 Top Words:
     Highest Prob: countri, moldova, situat, polit, worri, chang, govern
     FREX: countri, moldova, polit, govern, minist, elect, moldovan
     Lift: countri, govern, accept, affair, airport, alegiril, arous
     Score: polit, countri, moldova, minist, prime, govern, billion
```

Plotting the same:

Hide

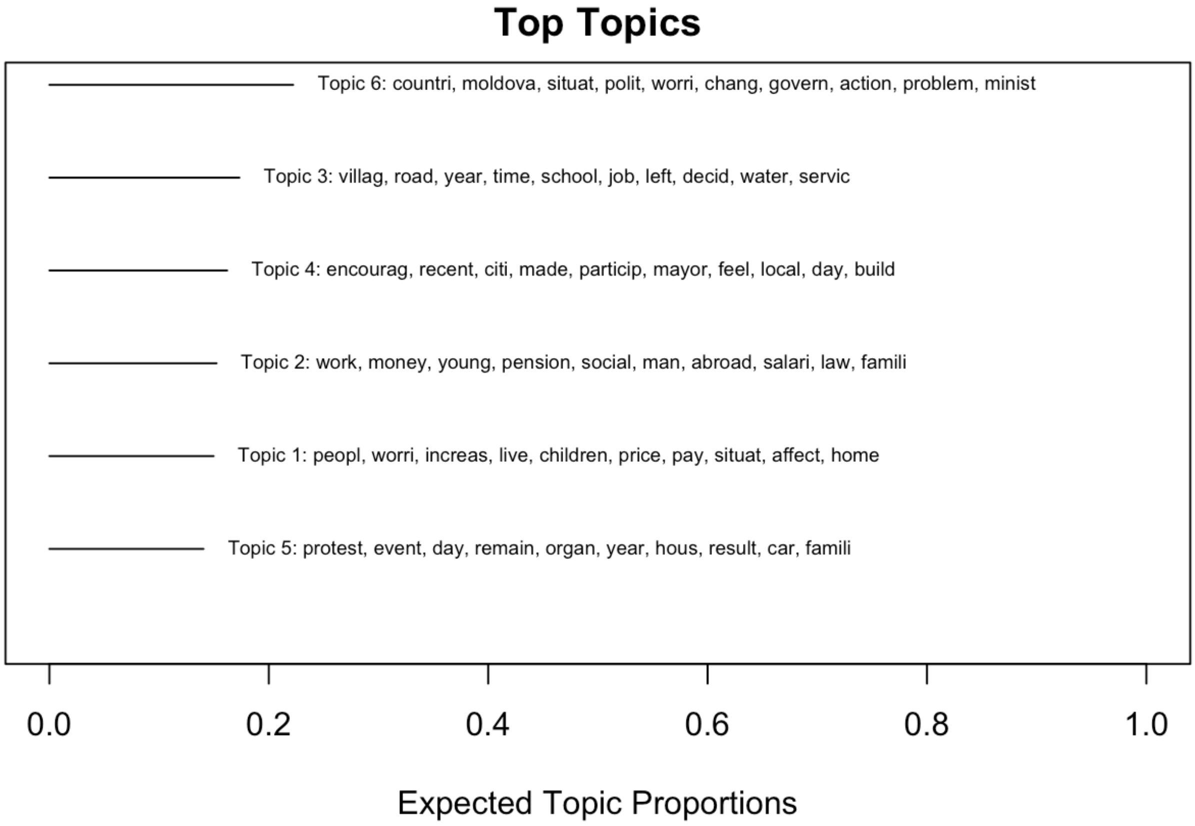
```
plot(topics6,type="labels", n = 15, text.cex = .6)
```



3.5.3 Graphically displaying estimated topic proportions

Hide

```
plot(topics6,type="summary", xlim = c(0, 1), n = 10, text.cex = .6)
```



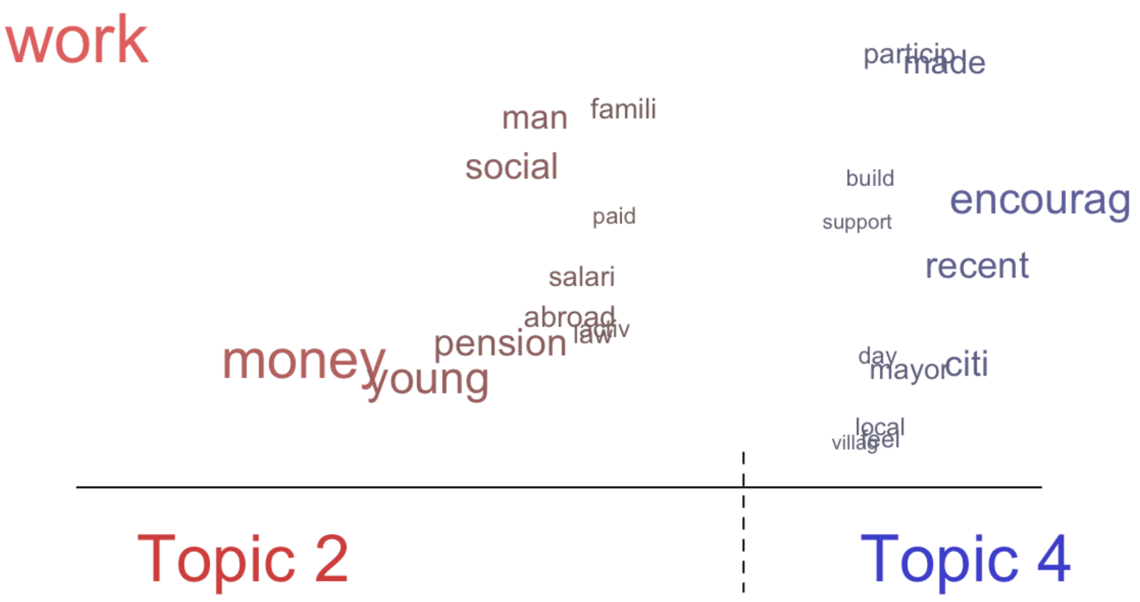
3.5.4 Comparing topics

If topics have similar top probability words, contrast in words across two topics can be plotted by calculating the difference in probability of a word for the two topics, and normalizing the maximum difference in probability of any word between the two topics.

To look at comparison between topics 2 and 4:

Hide

```
plot(topics6, type = "perspectives", topics = c(2,4))
```

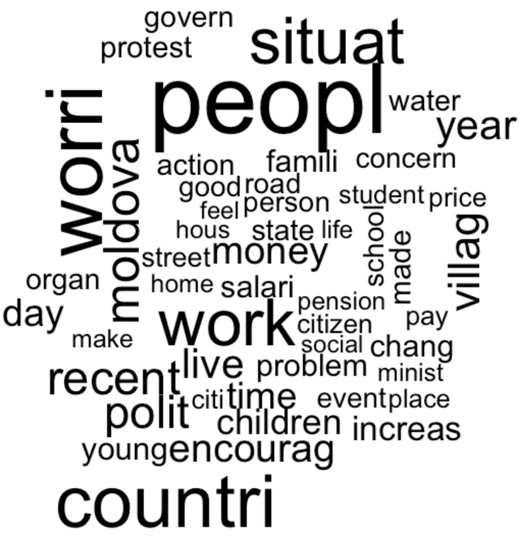


3.5.5 Creating word clouds for topics

To plot top 50 words:

Hide

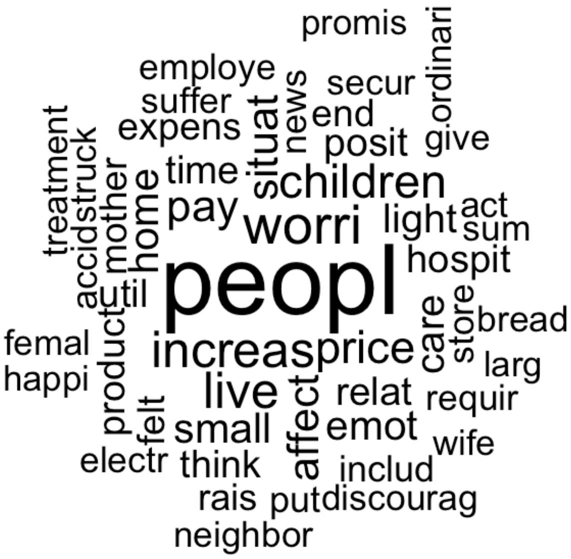
```
cloud(topics6, topic = NULL, scale = c(3, .25), max.words = 50)
```



To plot top probability words for Topic 1:

Hide

```
cloud(topics6, topic = 1, scale = c(3, 1), random.order = FALSE,rot.per = .3, max.w
ords = 50)
```



3.5.6 Estimating relationship between metadata and topic prevalence

3.5.6.1: Estimating effect of gender

Hide

```
con.eff <- estimateEffect( ~ factor(DQ2.Gender),
                          topics6, meta = stm.dfm$meta,
                          uncertainty = "Global")
```

To plot the results of the analysis as the difference in topic proportions for two different values of our factor variable (male vs female). Point estimates and 95% confidence intervals:

Hide

```
plot(con.eff, covariate = "DQ2.Gender",
     model = topics6, method = "difference",
     cov.value1 = "male", cov.value2 = "female", verbose.labels = FALSE,
     main = "Effect of Gender")
```

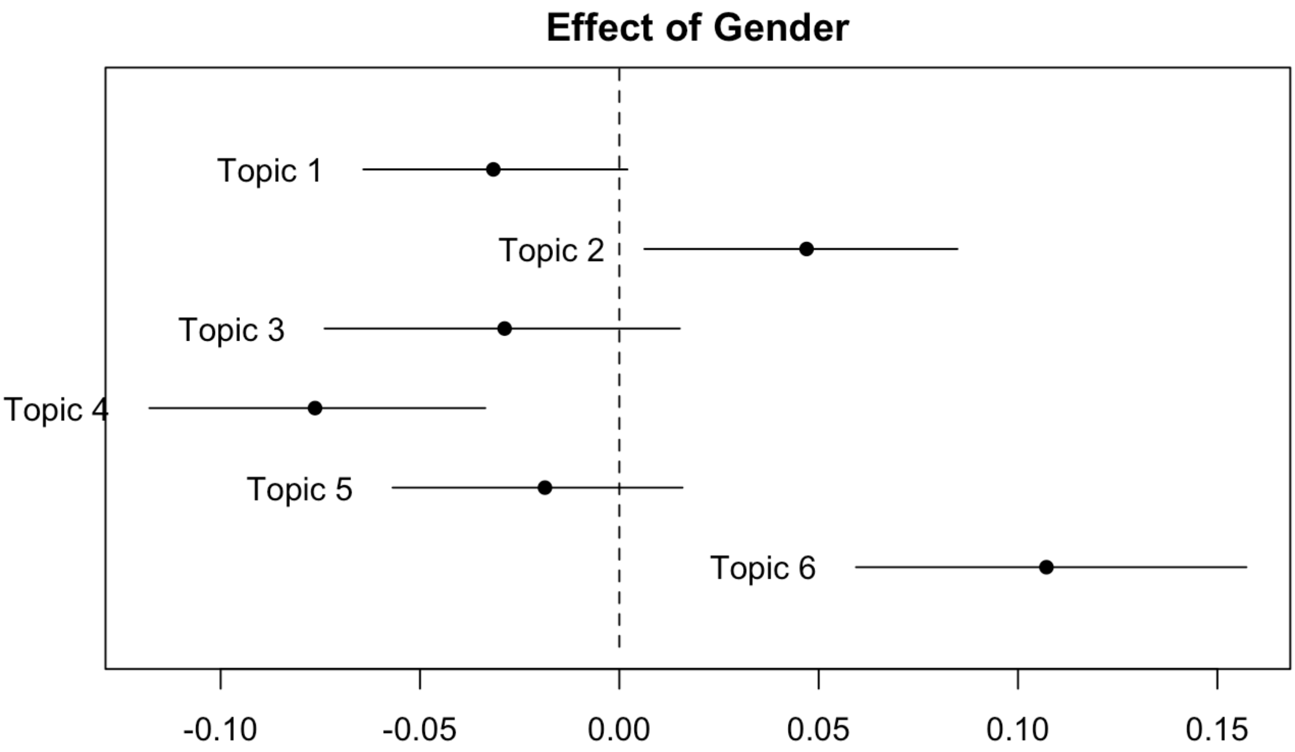


Figure shows a treatment effect of gender (male==1 vs female==0) in topics 2, 4, and 6. This can be assessed by looking that 95% confidence interval bars do not overlap the zero line, i.e. the effect is statistically distinct from zero. Compared to females, men are more likely to discuss topics 2 and 6, and less likely topic 4.

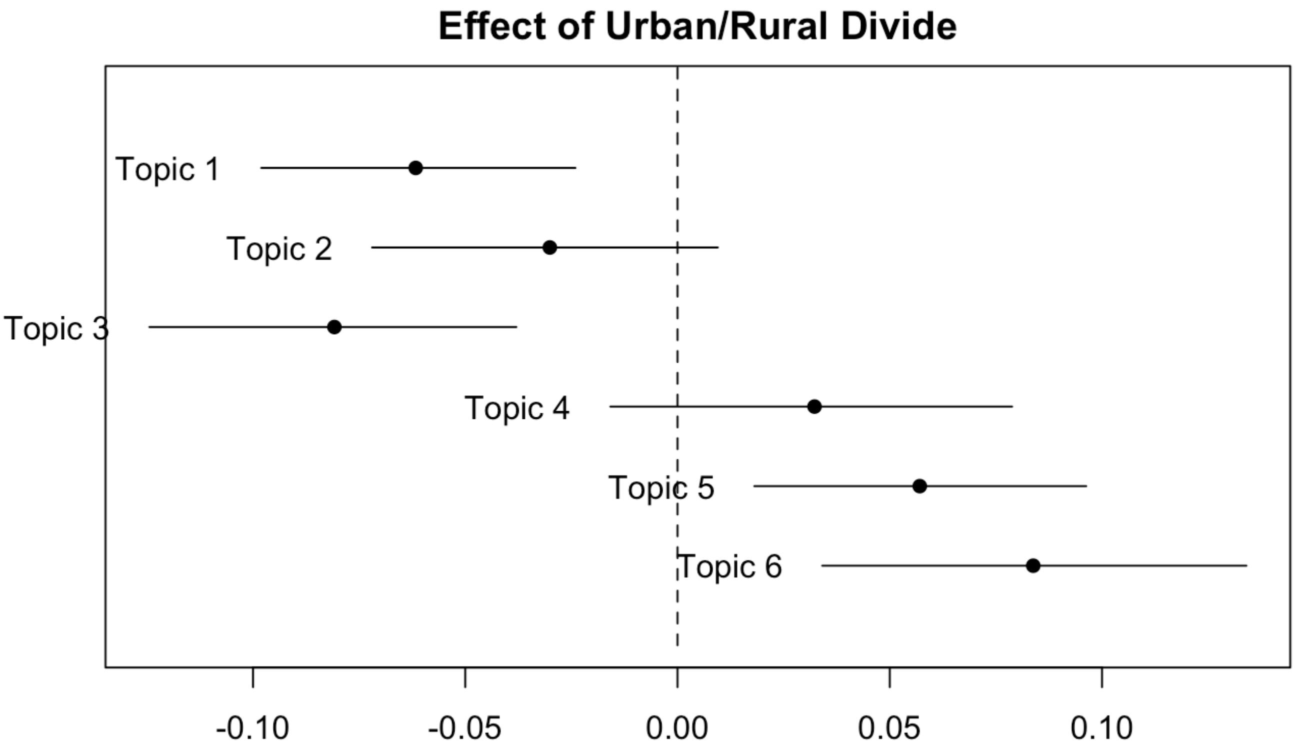
3.5.6.2: Estimating effect of rural vs urbal area

Hide

```
con.eff <- estimateEffect( ~ factor(DQ5.Live),
                          topics6, meta = stm.dfm$meta,
                          uncertainty = "Global")
```

Hide

```
plot(con.eff, covariate = "DQ5.Live",
     model = topics6, method = "difference",
     cov.value1 = "urban area", cov.value2 = "rural area", verbose.labels = FALSE,
     main = "Effect of Urban/Rural Divide")
```



The figure shows a treatment effect of urbanisation (urban area==1 vs rural area==0) in topics 1, 3, 5, and 6. This can be judged by assessing that 95% confidence interval bars do not overlap the zero line, i.e. the effect is statistically distinct from zero. Compared to rural areas, respondents in urban areas are more likely to discuss topics 5 and 6, and less likely topics 1 and 3.

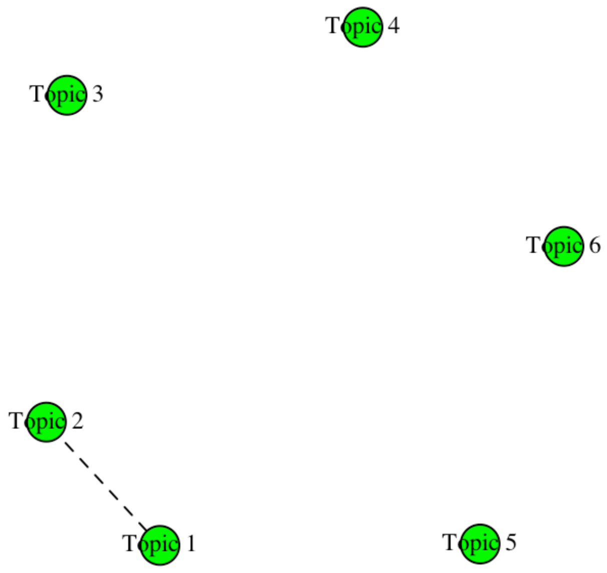
Previous word cloud results of topics can be looked at to assess whether the results hold face validity.

3.5.7: Establishing correlation between topics

Positive correlations between topics suggest that both topics are likely to be covered within a narrative.

Hide

```
topic.cor <- topicCorr(topics6)
plot(topic.cor)
```



It appears that topics 2 and 1 are linked at the default 0.01 correlation cutoff.