Introduction to Data Science for Public Policy

Class 8: Text as Data

Thomas Monk

t.d.monk@lse.ac.uk

with thanks to Dr. Melissa Sands, LSE, from whose slides these are adapted.

30 August 2022

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.
- But this is a data science course, at a university doing really exciting **research** in this area.

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.
- But this is a data science course, at a university doing really exciting **research** in this area.
- So I think it's important to push you as close as I can to the limit of research, now that I've given you the tools to learn the technicalities yourself.

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.
- But this is a data science course, at a university doing really exciting **research** in this area.
- So I think it's important to push you as close as I can to the limit of research, now that I've given you the tools to learn the technicalities yourself.
- The areas which are left behind:

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.
- But this is a data science course, at a university doing really exciting **research** in this area.
- So I think it's important to push you as close as I can to the limit of research, now that I've given you the tools to learn the technicalities yourself.
- The areas which are left behind:
  - Panda's split-apply-combine paradigm. Extremely comprehensive documentation is available, and I will add a problem set for you to practice these skills along with some useful links.

New Plan

- The plan was to teach you some more advanced Pandas techniques today, along with some basic statistical analysis.
- But this is a data science course, at a university doing really exciting **research** in this area.
- So I think it's important to push you as close as I can to the limit of research, now that I've given you the tools to learn the technicalities yourself.
- The areas which are left behind:
  - Panda's split-apply-combine paradigm. Extremely comprehensive documentation is available, and I will add a problem set for you to practice these skills along with some useful links.
  - Scipy regression (perhaps will pick up tomorrow) -

    ```
    from sklearn.linear_model import LinearRegression
    reg = LinearRegression().fit(X, y)
    ```

Text as Data

- A huge amount of economics & politics happens through **language**:

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.
- What could we do with all this data?

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.
- What could we do with all this data?
  - classify documents as belonging to particular types

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.
- What could we do with all this data?
  - classify documents as belonging to particular types
  - describe the content and sentiment of text

Text as Data

- A huge amount of economics & politics happens through **language**:
    - Job adverts, job descriptions, employee task descriptions.
    - News articles, opinion columns
    - Text-based correspondence
    - Open-ended survey questions, interviews, focus groups
    - Twitter, Facebook, Reddit, blogs, other social media
    - Bills, laws, treaties, constitutions, court opinions and judgements
    - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.
- What could we do with all this data?
    - classify documents as belonging to particular types
    - describe the content and sentiment of text
    - use text to inform our coding of variables

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs $\rightarrow$ **automated text analysis**.
- What could we do with all this data?
  - classify documents as belonging to particular types
  - describe the content and sentiment of text
  - use text to inform our coding of variables
  - ... and more!

Text as Data

- A huge amount of economics & politics happens through **language**:
  - Job adverts, job descriptions, employee task descriptions.
  - News articles, opinion columns
  - Text-based correspondence
  - Open-ended survey questions, interviews, focus groups
  - Twitter, Facebook, Reddit, blogs, other social media
  - Bills, laws, treaties, constitutions, court opinions and judgements
  - Political speeches, legislative debates, committee transcripts, party manifestos, lobbying documents, campaign websites
- These are (often) **big data**: computational costs → **automated text analysis**.
- What could we do with all this data?
  - classify documents as belonging to particular types
  - describe the content and sentiment of text
  - use text to inform our coding of variables
  - ... and more!
- **Note:** This is an active area of research → new methods developed all the time.

From Documents to Data

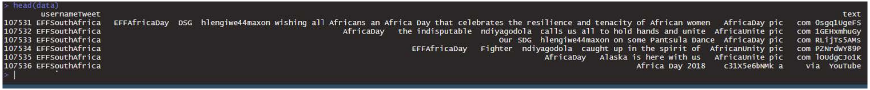
- The data begin as a structured document (a tweet, a bill, a speech).

> **Economic Freedom Fighters**
> @EFFSouthAfrica
>
> #EFFAfricaDay DSG @hlengiwe44maxon wishing all Africans an Africa Day that celebrates the resilience and tenacity of African women. #AfricaDay

From Documents to Data

- The data begin as a structured document (a tweet, a bill, a speech).

- Multiple documents make a **corpus** (a collection of docs):



```
> head(data)
       usernameTweet                                                                                                          text
107531 EFFSouthAfrica   EFFAfricaDay  DSG  hlengiwe44maxon wishing all Africans an Africa Day that celebrates the resilience and tenacity of African women   AfricaDay pic   com 0sgq1UgeFS
107532 EFFSouthAfrica                                                  AfricaDay   the indisputable  ndiyagodola  calls us all to hold hands and unite  AfricaUnite pic   com 1GEHxmhuGy
107533 EFFSouthAfrica                                                                                 Our SDG  hlengiwe44maxon on some Pantsula Dance  AfricaDay pic   com RLijTs5AMs
107534 EFFSouthAfrica                                                                EFFAfricaDay   Fighter  ndiyagodola  caught up in the spirit of  AfricanUnity pic   com PZNrdWY89P
107535 EFFSouthAfrica                                                                                           AfricaDay   Alaska is here with us   AfricaUnite pic   com lOUdgCJo1K
107536 EFFSouthAfrica                                                                                                     Africa Day 2018    c31X5e6bNMk a     via  YouTube
> |
```

From Documents to Data

- The data begin as a structured document (a tweet, a bill, a speech).

- Multiple documents make a **corpus** (a collection of docs):

```
> head(data)
        usernameTweet                                                                                                                          text
107531 EFFSouthAfrica   EFFAfricaDay  DSG  hlengiwe44maxon wishing all Africans an Africa Day that celebrates the resilience and tenacity of African women   AfricaDay pic   com 0sgq1UgeFS
107532 EFFSouthAfrica                                                  AfricaDay   the indisputable  ndiyagodola  calls us all to hold hands and unite  AfricaUnite pic   com 1GEHxmhuGy
107533 EFFSouthAfrica                                                                             Our SDG  hlengiwe44maxon on some Pantsula Dance  AfricaDay pic   com RLijTs5AMs
107534 EFFSouthAfrica                                                                   EFFAfricaDay   Fighter  ndiyagodola  caught up in the spirit of  AfricanUnity pic   com PZNrdWY89P
107535 EFFSouthAfrica                                                                                     AfricaDay   Alaska is here with us   AfricaUnite pic   com lOUdgCJo1K
107536 EFFSouthAfrica                                                                                                   Africa Day 2018    c31X5e6bNMk a     via  YouTube
> |
```

- Naively, this is just a dataset with a character column of “text”

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
           Docs
Terms        1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed        1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp       1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get        1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good       1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
           Docs
Terms       49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed        0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get        0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good       0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                    asPlain = TRUE,
+                                                    stopwords = TRUE,
+                                                    tolower = TRUE,
+                                                    removeNumbers = TRUE,
+                                                    stemWords = TRUE,
+                                                    removePunctuation = TRUE,
+                                                    removeSeparators = TRUE,
+                                                    removeTwitter = TRUE,
+                                                    stem = TRUE,
+                                                    stripWhitespace = TRUE,
+                                                    wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
       Docs
Terms  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed  1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get  1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
       Docs
Terms  49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
          Docs
Terms       1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed       1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp      1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get       1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good      1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
          Docs
Terms      49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp      0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good      0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
      Docs
Terms  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed  1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get  1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
      Docs
Terms  49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
      Docs
Terms  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed  1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get  1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good 1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
      Docs
Terms  49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                    asPlain = TRUE,
+                                                    stopwords = TRUE,
+                                                    tolower = TRUE,
+                                                    removeNumbers = TRUE,
+                                                    stemWords = TRUE,
+                                                    removePunctuation = TRUE,
+                                                    removeSeparators = TRUE,
+                                                    removeTwitter = TRUE,
+                                                    stem = TRUE,
+                                                    stripWhitespace = TRUE,
+                                                    wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
        Docs
Terms    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed    1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  corp   1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  get    1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  good   1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        Docs
Terms    49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  corp   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  get    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  good   0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context? Lots of zeros. So?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
            Docs
Terms       1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed       1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp      1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get       1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good      1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
            Docs
Terms       49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed        0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get        0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good       0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context? Lots of zeros. So? There is a large variety of words used, with some used ultra-frequently, and some used hardly at all.
- What do we miss by understanding the data in this way?

From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                     asPlain = TRUE,
+                                                     stopwords = TRUE,
+                                                     tolower = TRUE,
+                                                     removeNumbers = TRUE,
+                                                     stemWords = TRUE,
+                                                     removePunctuation = TRUE,
+                                                     removeSeparators = TRUE,
+                                                     removeTwitter = TRUE,
+                                                     stem = TRUE,
+                                                     stripWhitespace = TRUE,
+                                                     wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
        Docs
Terms    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed    1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp   1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get    1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good   1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
        Docs
Terms    49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed     0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get     0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good    0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context? Lots of zeros. So? There is a large variety of words used, with some used ultra-frequently, and some used hardly at all.
- What do we miss by understanding the data in this way?

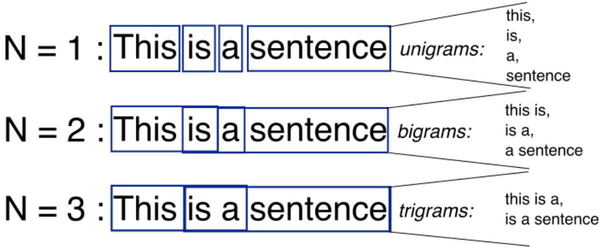
From Documents to Data

- We can then convert this to a **(cleaned, sparse)** term-document matrix:

```
> tdm = TermDocumentMatrix(data_text, control = list(verbose = TRUE,
+                                                    asPlain = TRUE,
+                                                    stopwords = TRUE,
+                                                    tolower = TRUE,
+                                                    removeNumbers = TRUE,
+                                                    stemWords = TRUE,
+                                                    removePunctuation = TRUE,
+                                                    removeSeparators = TRUE,
+                                                    removeTwitter = TRUE,
+                                                    stem = TRUE,
+                                                    stripWhitespace = TRUE,
+                                                    wordLengths=c(3, Inf)))
>   m - as.matrix(tdm)
> head(m)
          Docs
Terms      1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
  bed      1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp     1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get      1 0 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good     1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
          Docs
Terms      49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93
  bed       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  corp      0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get       0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  good      0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

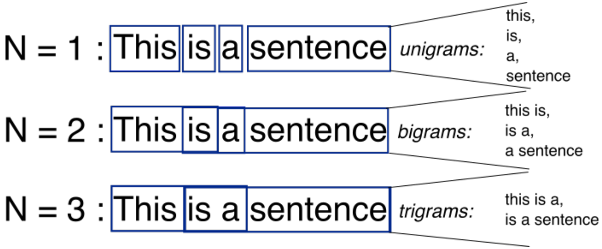
- How do we interpret this matrix? The matrix dhows the count of each term (axis 0) on each document (axis 1).
- What could sparse mean in this context? Lots of zeros. So? There is a large variety of words used, with some used ultra-frequently, and some used hardly at all.
- What do we miss by understanding the data in this way? **Structure** - how the word is used in **context**. We see these as **n-grams**.

Aside: ngram



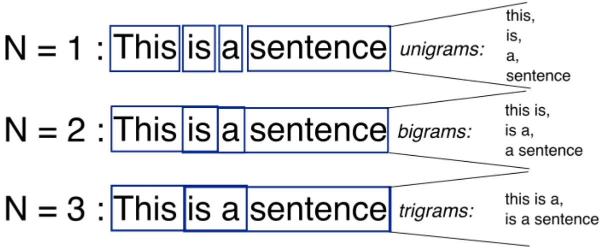
- What's an n-gram?

Aside: ngram



- What's an n-gram?

Aside: ngram



- What's an n-gram? Just a sequence of **words** within our data set.
- We call the unit of analysis **tokens** in text analysis - this can be a n-gram of any size.

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.
2. Tomorrow: supervised learning (more broad than just text by nature).

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.
2. Tomorrow: supervised learning (more broad than just text by nature).

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.
2. Tomorrow: supervised learning (more broad than just text by nature).

Supervised learning methods:

1. Construct training set - i.e. we teach the program what tokens belong in what category, for a **subset** of tokens.

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.
2. Tomorrow: supervised learning (more broad than just text by nature).

Supervised learning methods:

1. Construct training set - i.e. we teach the program what tokens belong in what category, for a **subset** of tokens.
2. Apply supervised learning method - the program then looks at **all** tokens, and classifies each according to some algorithm.

Classification

We now have an enumeration of:

- all words or n-grams (the **tokens**)
- whether they are present in each corpus entry (the **documents**, or **units**)

For classification (defined set of categories), two main approaches:

1. Today: dictionary methods.
2. Tomorrow: supervised learning (more broad than just text by nature).

Supervised learning methods:

1. Construct training set - i.e. we teach the program what tokens belong in what category, for a **subset** of tokens.
2. Apply supervised learning method - the program then looks at **all** tokens, and classifies each according to some algorithm.
3. Validate - check how we did.

Example: Sentiment Analysis (Bleich & van der Veen 2021)

- Authors ask whether Muslims are portrayed more negatively than other religious groups in the media, and, if so, under what conditions.

Example: Sentiment Analysis (Bleich & van der Veen 2021)

- Authors ask whether Muslims are portrayed more negatively than other religious groups in the media, and, if so, under what conditions.
- Collect 850,000 articles that mention Muslims, Hindus, Jews, or Catholics in 17 US newspapers from 1996–2015 and compare them to a representative baseline of articles

Example: Sentiment Analysis (Bleich & van der Veen 2021)

- Authors ask whether Muslims are portrayed more negatively than other religious groups in the media, and, if so, under what conditions.
- Collect 850,000 articles that mention Muslims, Hindus, Jews, or Catholics in 17 US newspapers from 1996–2015 and compare them to a representative baseline of articles
- **Approach**: dictionary (lexicon-based) sentiment analysis

Example: Sentiment Analysis (Bleich & van der Veen 2021)

- Authors ask whether Muslims are portrayed more negatively than other religious groups in the media, and, if so, under what conditions.
- Collect 850,000 articles that mention Muslims, Hindus, Jews, or Catholics in 17 US newspapers from 1996–2015 and compare them to a representative baseline of articles
- **Approach**: dictionary (lexicon-based) sentiment analysis
- **Covariates**: location (US vs. non-US), after Sept 11 attacks, partisan lean of the newspaper, broadsheet vs. tabloid

Example: Sentiment Analysis (Bleich & van der Veen 2021)

- Authors ask whether Muslims are portrayed more negatively than other religious groups in the media, and, if so, under what conditions.
- Collect 850,000 articles that mention Muslims, Hindus, Jews, or Catholics in 17 US newspapers from 1996–2015 and compare them to a representative baseline of articles
- **Approach**: dictionary (lexicon-based) sentiment analysis
- **Covariates**: location (US vs. non-US), after Sept 11 attacks, partisan lean of the newspaper, broadsheet vs. tabloid
- **Main finding**: average tone of articles about Muslims is more negative than the baseline and compared to articles about other groups

Sentiment Analysis (Bleich & van der Veen 2021)

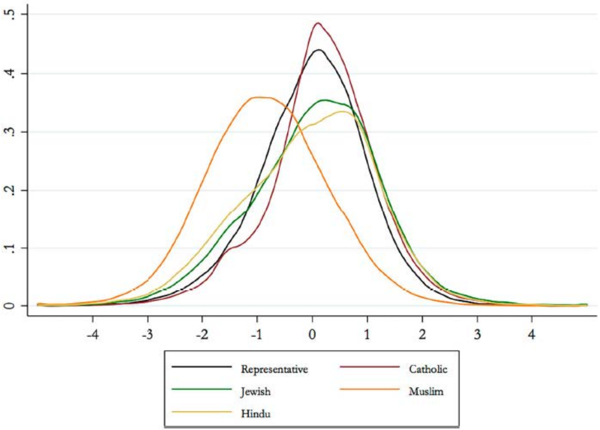


Figure: Valence (sentiment) distribution estimates (kernel density) of articles by religion compared to the representative corpus - i.e the proportion negative/positive.

Bleich, E. and van der Veen, A.M., 2021. Media portrayals of Muslims: a comparative sentiment analysis of American newspapers, 1996–2015. Politics, Groups, and Identities, 9(1), pp.20-39.

Sentiment Analysis (Bleich & van der Veen 2021)

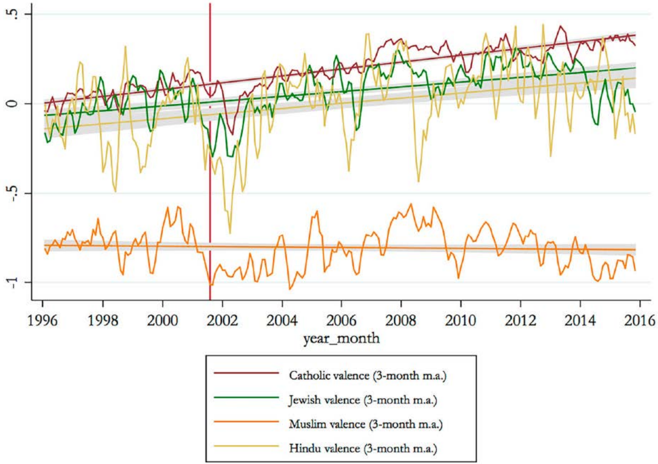


Figure: Three-month moving average valences of articles mentioning religious groups.

What is this showing?

Sentiment Analysis (Bleich & van der Veen 2021)

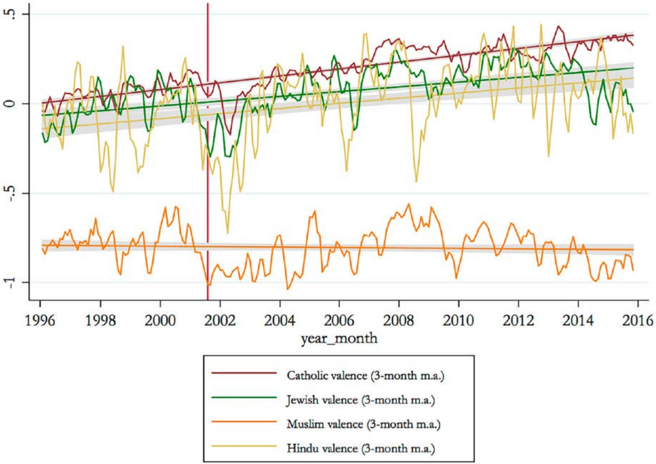


Figure: Three-month moving average valences of articles mentioning religious groups.

What is this showing? Sentiment towards Muslims remains negative over time.

Sentiment Analysis (Bleich & van der Veen 2021)

Table 3. Sample preprocessed sentences at standard deviation valence intervals.

Valence score	Sample preprocessed sentences
−2	• That has been followed by the taking over of most of the country by the Taliban a fundamentalist Islamic movement that considers much of western culture to be evil • A Muslim man charged with setting fire to a Marietta mosque may be in the country illegally law enforcement officials confirmed Thursday
−1	• Muslims have been silent and even maybe passive members of the society Rehana Jan said • Separately Bosnian Muslims rallied yesterday to demand to return to a Serb-held town
0	• As a foreign correspondent I spent 13 of 17 years close to Muslims as a colleague and friend first at the UN then in India, Pakistan and Afghanistan • I am talking about the major world religions Judaism, Islam, and Christianity
1	• Colleagues said they expected her to ask secretary of state Colin I Powell one of the administration's most popular figures to embark on a listening tour in crucial Muslim nations • The journey is not a required part of the annual pilgrimage or hajj but many Muslims take the chance to visit Islam's second most holy site
2	• Arizona is home to infinite spiritual expressions from the centuries-old Catholic tradition brought by the early Jesuit and Franciscan priests to the Muslims who call modern-day Arizona home • Each food kit was enriched with extra vitamins and designed to be acceptable to the Muslim diet while also supplying 2300 calories enough nutrition for about a day

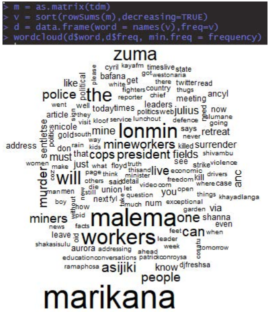
Bleich, E. and van der Veen, A.M., 2021. Media portrayals of Muslims: a comparative sentiment analysis of American newspapers, 1996–2015. Politics, Groups, and Identities, 9(1), pp.20-39.

Description and Unsupervised Learning

Classification is useful, but text lends itself well to rich description:

- What topics are present?
- Who is talking about what?
- Are there thematic linkages in the corpus, etc.?

- We might start with **frequencies**: what words are common?
- Many options here, but one easy and attractive approach is **wordclouds**.



But quite impressionistic, only useful when small number of topics exist, and doesn't allow for hypothesis testing.

Topic modelling is the next step (LDA or Structual Topic Modelling).