Outline

- Data preprocessing
- Decomposing a dataset: instances and features
- Basic data descriptors
- Feature spaces and proximity (similarity, distance) measures
- Feature transformation for text data

Feature transformations for text data 1/6

- Text represented as a set of terms (“Bag-Of-Words“ model)
  - Terms:
    - Single words (“cluster“, “analysis“..)
      or
    - bigrams, trigrams, …n-grams (“cluster analysis“..)
  - Transformation of a document d in a vector $r(d) = (h_1, \ldots, h_d)$, $h_i \geq 0$: the frequency of term t_i in d

The region is preparing for blizzard conditions Friday, with the potential for more than two feet of snow in the Fairfax City area. Conditions are expected to deteriorate Friday afternoon, with the biggest snowfall, wind gusts and life-threatening conditions Friday night and Saturday.

…	…
blizzard	1
Friday	3
and	2
Zombie	0
…	…

$h_{\text{“Friday“}}$

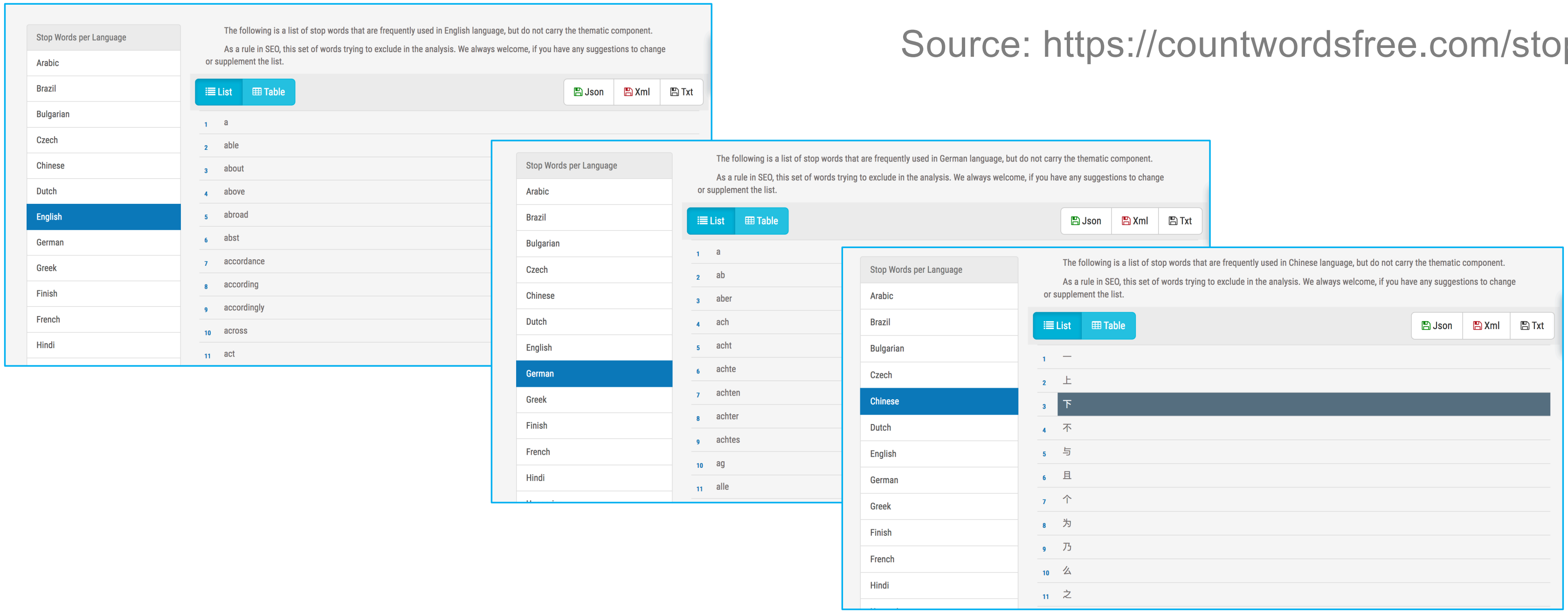
Feature transformations for text data 2/6

- Challenges/Problems in Text Mining:
  1. Common words (“e.g.”, “the”, “and”, “for”, “me”)
  2. Words with the same root (“fish”, “fisher”, “fishing”,…)
  3. Very high-dimensional space (dimensionality d > 10.000)
  4. Not all terms are equally important
  5. Most term frequencies $h_i = 0$ (“sparse feature space“)
- More challenges due to language:
  - Different words have same meaning (synonyms)
    - “freedom” – “liberty”
  - Words have more than one meanings
    - e.g. “java”, “mouse”

Feature transformations for text data 3/6

- Problem 1: Common words (“e.g.”, “the”, “and”, “for”, “me”)
  - Solution: ignore these terms (stop-words) or remove stop-words

    There are stop-words lists for all languages in WWW.



Source: https://countwordsfree.com/stopwords

Feature transformations for text data 3/6

- Problem 2: Words with the same root (“fish”, “fisher”, “fishing”,…)
  - Solution: Stemming

    Map the words to their root
    example: "fishing", "fished", "fish", and "fisher" to the root word, "fish".
    The root of the words is the output of stemming.

    Challenge: Avoid overstemming and/or understemming

    Example for obverstemming:
    university, universal, universities, and universe → “univers” meaning of the word/term is lost

    Example for understemming:
    stemming “data” → “dat” and “datum” → “datu” is too weak (understemming),
    Better solution would be data, datum → “dat”, but then we would have the problem to treat “date” that would be reduced to “dat” as well.

    For English, the Porter stemmer is widely used. (Stemming solutions exist for other languages also)
    (Porters Stemming Algorithms: http://tartarus.org/~martin/PorterStemmer/index.html)

    More advanced stemmer: snowball stemmer (Porter2), lancaster stemmer

Feature transformations for text data 4/6

- Problem 3: Too many features/ terms
  - Solution: Select the most important features ("Feature Selection")
  - Example: average document frequency for a term
    - Very frequent terms appear in almost all documents
    - Very rare terms appear in only a few documents

Ranking procedure:

$$DF(t_i) = \frac{\#Docs\ containing\ t_i}{\#All\ documents}$$

1. Compute document frequency for all terms t_i :
2. Sort terms w.r.t. $DF(t_i)$ and get $rank(t_i)$
3. Sort terms by score(t_i)= $DF(t_i)$ · rank(t_i)
   e.g. score(t_{23}) = 0.82 · 1 = 0.82
   score(t_{17}) = 0.65 · 2 = 1.3
4. Select the k terms with the largest score(t_i)

Rank	Term	DF
1.	t_{23}	*0.82*
2.	t_{17}	*0.65*
3.	t_{14}	*0.52*
4.	…	…

Feature transformations for text data 5/6

- Problem 4: Not all terms that are frequent are equally important, what value should be assigned to a term (feature)?
  - Idea: Very frequent terms that are frequent in all documents are less informative than less frequent words. Define such a term weighting schema.
  - Solution: TF-IDF (Term Frequency · Inverse Document Frequency)

  Consider both the importance of the term in the document and in the whole collection of documents.

$$TF(t,d) = \frac{n(t,d)}{\sum_{w \in d} n(w,d)}$$

The relative frequency of term t in d [n(t,d) = # t in d]

$$IDF(t) = \log\left(\frac{|DB|}{\left|\{d \mid d \in DB \wedge t \in d\}\right|}\right)$$

Inverse frequency of term t in all DB

$$TF \times IDF = TF(t,d)IDF(t)$$

Feature vector with TF IDF : $r(d) = (TF(t_1, d) \cdot IDF(t_1), \ldots, TF(t_n, d) \cdot IDF(t_n))$

Feature transformations for text data 6/6

- Problem 5: in each document, most of the terms have a frequency of $h_i = 0$
  What distance measure should we use to compare documents?
  - Euclidean distance is not a good idea: it is influenced by vectors lengths (many 0-0 matchings)
  - Idea: use more appropriate distance measures

Jaccard Coefficient: Ignore terms absent in both documents (without prior feature transformation)

$$d_{Jaccard}(d_1, d_2) = 1 - \frac{|d_1 \cap d_2|}{|d_1 \cup d_2|} = \frac{|\{t | t \in d_1 \wedge t \in d_2\}|}{|\{t | t \in d_1 \vee t \in d_2\}|}$$

Cosine Coefficient: Consider feature-transformed documents (e.g. TF-IDF vectors)

$$d_{\cos inus}(d_1, d_2) = 1 - \frac{\langle d_1, d_2 \rangle}{\|d_1\| \cdot \|d_2\|} = 1 - \frac{\sum_{i=0}^{n} (d_{1,i} \cdot d_{2,i})}{\sqrt{\sum_{i=0}^{n} d_{1,i}^{\ 2}} \cdot \sqrt{\sum_{i=0}^{n} d_{2,i}^{\ 2}}}$$