PGR 210 - Natural Language Processing Part

Kristiania University College

By Huamin Ren

Huamin.ren@kristiania.no

Høyskolen Kristiania

Outline of week 42

- ***Bag-of-Words and TF-IDF***
- Visualization and analysis

Outline of Lec 11

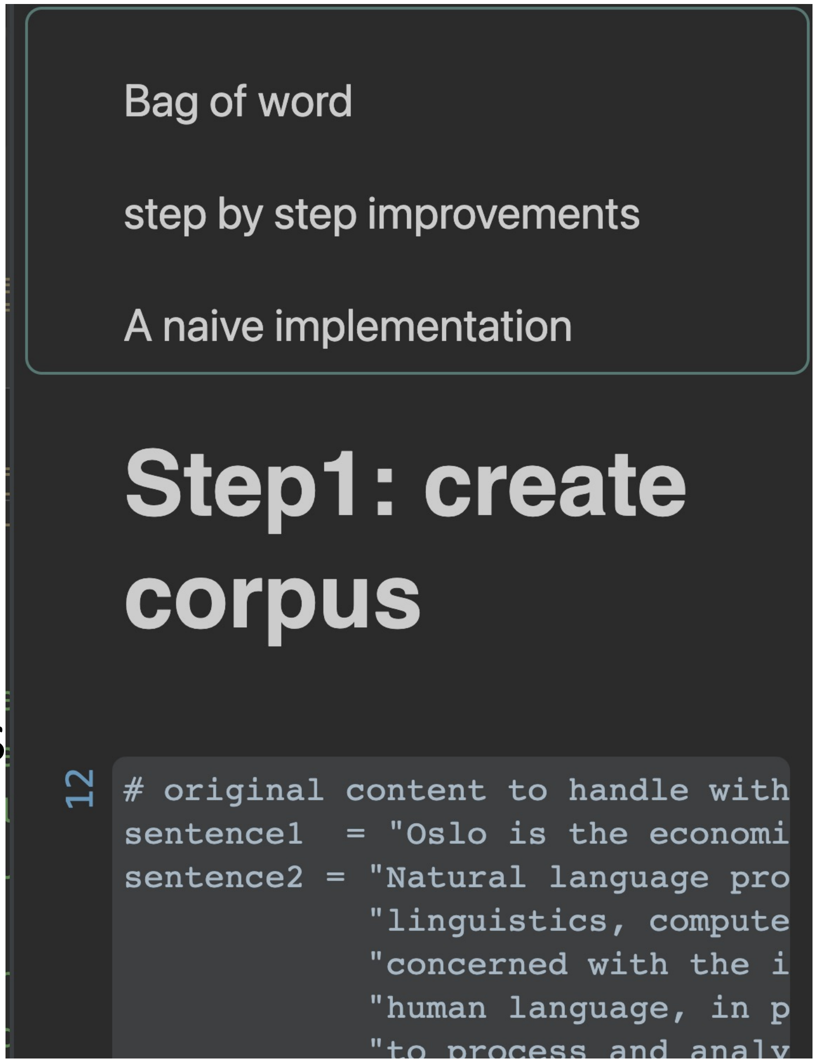
- Bag-of-words model
- TF-IDF
  - Implementing TF-IDF model
  - Understanding TF-IDF model
- Extracting features for new documents

Outline of Lec 12

- Document similarity
- Topic models
- Representative visualization and its assisted analysis

Outline of Lec 11

1. ***Bag-of-words model***
2. TF-IDF
   - Implementing TF-IDF model
   - Understanding TF-IDF model
3. Extracting features for new documents



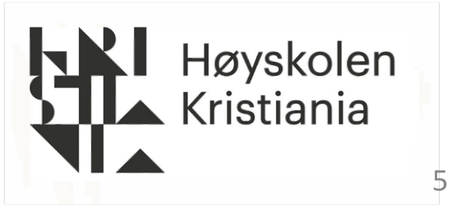
Bag of word

step by step improvements

A naive implementation

Step1: create corpus

```
# original content to handle with
sentence1  = "Oslo is the economi
sentence2 = "Natural language pro
            "linguistics, compute
            "concerned with the i
            "human language, in p
            "to process and analy
```



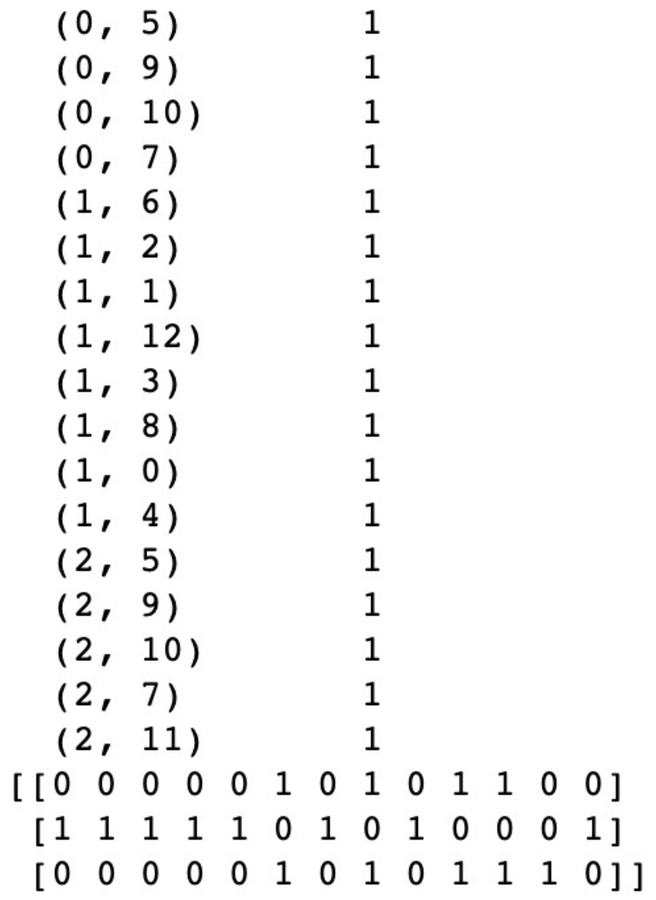
Outline of Lec 11

- Bag-of-words model
- ***TF-IDF***
  - Implementing TF-IDF model
  - Understanding TF-IDF model
- Extracting features for new documents

TF-IDF

- Stands for term frequency times inverse document frequency
  - Term frequencies are the counts of each word in a document
  - Inverse document frequency means that

TF-IDF

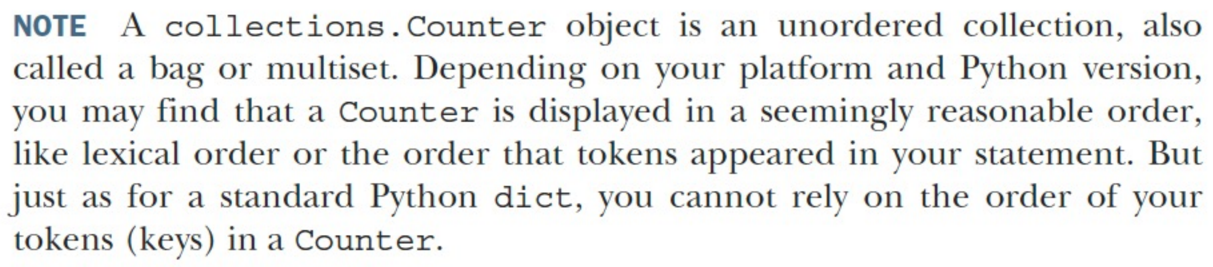


```
  (0, 5)	1
  (0, 9)	1
  (0, 10)	1
  (0, 7)	1
  (1, 6)	1
  (1, 2)	1
  (1, 1)	1
  (1, 12)	1
  (1, 3)	1
  (1, 8)	1
  (1, 0)	1
  (1, 4)	1
  (2, 5)	1
  (2, 9)	1
  (2, 10)	1
  (2, 7)	1
  (2, 11)	1
[[0 0 0 0 0 1 0 1 0 1 1 0 0]
 [1 1 1 1 1 0 1 0 1 0 0 0 1]
 [0 0 0 0 0 1 0 1 0 1 1 1 0]]
```

```
corpus = ['sentence1: similar to sentence3','sentence2: do'
        'not care what other sentences are saying','sentence3:
very similar to sentence1']
```

```
['are', 'care', 'donot', 'other', 'saying', 'sentence1', 'sentence2',
'sentence3', 'sentences', 'similar', 'to', 'very', 'what']
```

TF-IDF


NOTE A `collections.Counter` object is an unordered collection, also called a bag or multiset. Depending on your platform and Python version, you may find that a `Counter` is displayed in a seemingly reasonable order, like lexical order or the order that tokens appeared in your statement. But just as for a standard Python `dict`, you cannot rely on the order of your tokens (keys) in a `Counter`.

$$tf(w,D) = f_{w_D}$$

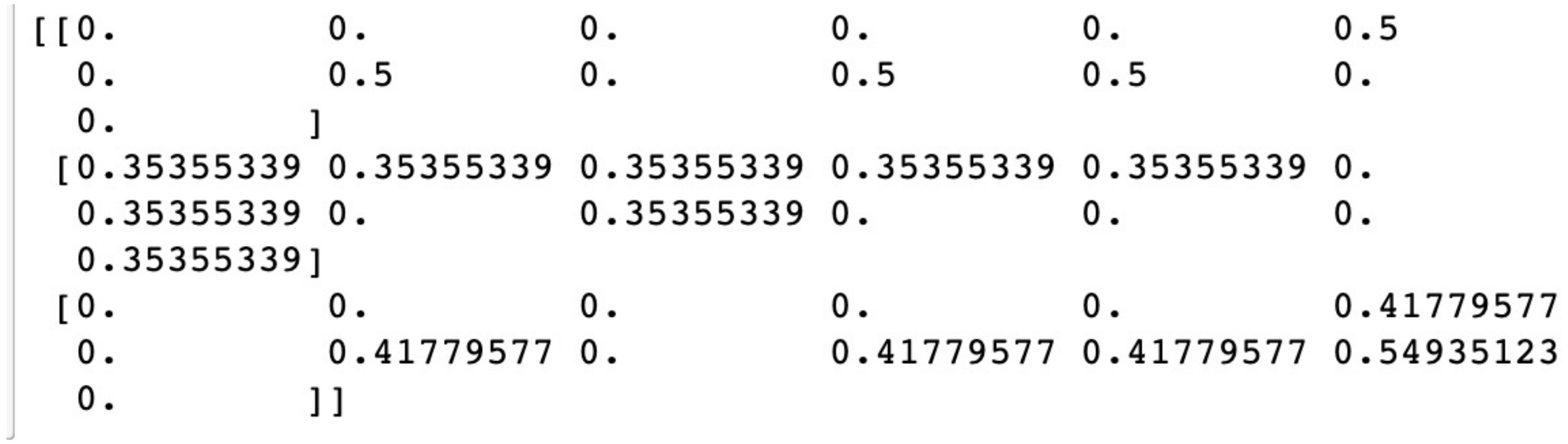
- TF: term frequency
- IDF: inverse document frequency

where f_{wD} denoted frequency for word **w** in document **D**, which becomes the term frequency (tf).

TF-IDF

$$\text{idf}(t, D) = \log \frac{\text{number of documents}}{\text{number of documents containing } t}$$

idf (w,D) represents the idf for the term/word w in document D, N represents the total number of documents in our corpus, and df(t) represents the number of documents in which the term w is present.



```
[[0.         0.         0.         0.         0.         0.5
  0.         0.5        0.         0.5        0.5        0.
  0.        ]
 [0.35355339 0.35355339 0.35355339 0.35355339 0.35355339 0.
  0.35355339 0.         0.35355339 0.         0.         0.
  0.35355339]
 [0.         0.         0.         0.         0.         0.41779577
  0.         0.41779577 0.         0.41779577 0.41779577 0.54935123
  0.        ]]
```

More thoughts

- By looking at an even more useful vector representation that counts the number of occurrences, or frequency, of each word in the given text, you assume that the more times a word occurs, the more meaning it must contribute to that document.
- Or if you have classified some words as expressing positive emotions—words like “good,” “best,” “joy,” and “fantastic”—the more a document that contains those words is likely to have positive “sentiment.”

Høyskolen Kristiania

Even more thoughts

- Let’s say you find the word “dog” 3 times in document A and 100 times in document B. Clearly “dog” is way more important to document B while using TF.
- Wait!

$$TF(\text{“dog,”}\ document_A) = 3/30 = .1$$
$$TF(\text{“dog,”}\ document_B) = 100/580000 = .00017$$

- Instead of raw word counts to describe your documents in a corpus, use normalized term frequencies.

  For example, calculate each word and get the relative importance to the document of that term.

- Suppose you have a corpus of 1 million documents, someone searches for the word “cat,” and in your 1 million documents you have exactly 1 document that contains the word “cat.” The raw IDF of this is

  1,000,000 / 1 = 1,000,000

- Suppose you have 10 documents with the word “dog” in them, the IDF for “dog” is:

  1,000,000 / 10 = 100,000

Høyskolen Kristiania

Similarly you can also implement IDF

- Be noted:

$$\mathrm{idf}(t, D) = \log \frac{\text{number of documents}}{\text{number of documents containing } t}$$

$$\mathrm{tf}(t, d) = \frac{\mathrm{count}(t)}{\mathrm{count}(d)}$$

$$\mathrm{idf}(t, D) = \log \frac{\text{number of documents}}{\text{number of documents containing } t}$$

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) * \mathrm{idf}(t, D)$$

Kristiania

- TF-IDF for a document: K-dimensional vector representation of each document in the corpus.
- Calculate the similarity: two vectors are considered similar if their cosine similarity is high, so minimize:

$$\cos \Theta = \frac{\mathbf{A} \cdot \mathbf{B}}{|A|\ |B|}$$

Two vectors, in a given vector space, can be said to be similar if they have a similar angle. If you imagine each vector starting at the origin and reaching out its prescribed distance and direction, the ones that reach out at the same angle are similar, even if they don't reach out to the same distance.

Høyskolen Kristiania

- The more times a word appears in the document, the TF (and hence the TF-IDF) will go up
- The number of documents that contain that word goes up, the IDF (and hence the TF-IDF) for that word will go down
- In some classes, all the calculations will be done in log space so that multiplications become additions and division becomes subtraction

Høyskolen Kristiania

Outline of Lec 11

- Bag-of-words model
- TF-IDF
    - Implementing TF-IDF model
    - Understanding TF-IDF model
- ***Extracting features for new documents***