### Classical text mining

Quiz, 5 questions

### **✓** Congratulations! You passed!

Next Item



1/1 point

1.

Choose true statements about text tokens.



A model without stemming/lemmatization can be the best

#### Correct

This is true. Word2vec embeddings, for instance, are trained on raw tokens.



Lemmatization needs more storage than stemming to work

### Correct

This is true, you have to store information about all possible word forms in the vocabulary.



Stemming can be done with heuristic rules

### Correct

Yeah, Porter stemmer works this way.



Lemmatization is always better than stemming

#### **Un-selected is correct**



1/1 point

2.

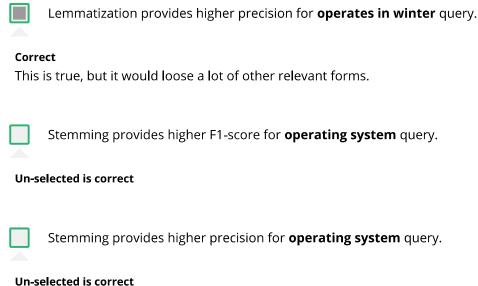
Imagine you have a texts database. Here are stemming and lemmatization results for some of the words:

Classical text mining	Stem	Lemma
Quiz, 5 questions operate	oper	operate
operating	oper	operating
operates	oper	operates
operation	oper	operation
operative	oper	operative
operatives	oper	operative
operational	oper	operational

Imagine you want to find results in your texts database using the following queries:

- 1. **operating system** (we are looking for articles about OS like Windows or Linux)
- 2. operates in winter (we are looking for machines that can be operated in winter)

Before execution of our search we apply either stemming or lemmatization to both query and texts. Compare stemming and lemmatization for a given query and choose the correct statements.



Stemming provides higher recall for operates in winter query.

#### Correct

This is true, lemmatization would only find exact matches with operates and lose a lot of relevant forms like operational.

Classical bag-of-words **vectorizer** (object that does vectorization) needs an amount of RAM at least proportional to T, which is the number of unique tokens in the dataset.

#### Correct

This is true, you have to store a hash map {token: index} to be able to vectorize new texts.

For bag-of-words features you need an amount of RAM at least proportional to  $N \times T$ , where N is the number of documents, T is the number of unique tokens in the dataset.

#### **Un-selected** is correct

Hashing **vectorizer** (object that does vectorization) needs an amount of RAM proportional to vocabulary size to operate.

#### **Un-selected** is correct



1/1 point

4.

Let's consider the following texts:

- · good movie
- · not a good movie
- · did not like
- i like it

• good one

### Classical text mining

Let's count **Term Frequency** here as a distribution over tokens in a particular text, for example for text "good one" Quiz, 5 questions we have TF = 0.5 for "good" and "one" tokens.

## Term frequency (TF)

- tf(t, d) frequency for term (or n-gram) t in document d
- Variants:

weighting scheme	TF weight
binary	0,1
raw count	$f_{t,d}$
term frequency	$f_{t,d}/\sum_{t'\in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$

# **Inverse document frequency (IDF)**

- N = |D| total number of documents in corpus
- $|\{d \in D: t \in d\}|$  number of documents where the term t appears

• 
$$idf(t,D) = log \frac{N}{|\{d \in D: t \in d\}|}$$

What is the **sum** of TF-IDF values for 1-grams in "good movie" text? Enter a math expression as an answer. Here's an example of a valid expression: log(1/2)\*0.1.

Preview

$$-0.5\log(3) - 0.5\log(2) + 1.0\log(5)$$

Clas	9sitad(5/8)x+0.fitling(512g
	5 questions
C	forrect Response
Y	our answer, 0.5*log(5/3)+0.5*log(5/2), is equivalent to the instructor's answer (0.5 * log(5/3))+(0.5 * log(5/2)).
<b>5</b> .	1 / 1 point
Wh	at models are usable on top of bag-of-words features (for 100000 words)?
	Naive Bayes
c	forrect
	Gradient Boosted Trees
U	In-selected is correct
	Logistic Regression
С	forrect
	SVM
C	orrect Control of the
	Decision Tree
U	In-selected is correct

# Classical text mining

Quiz, 5 questions