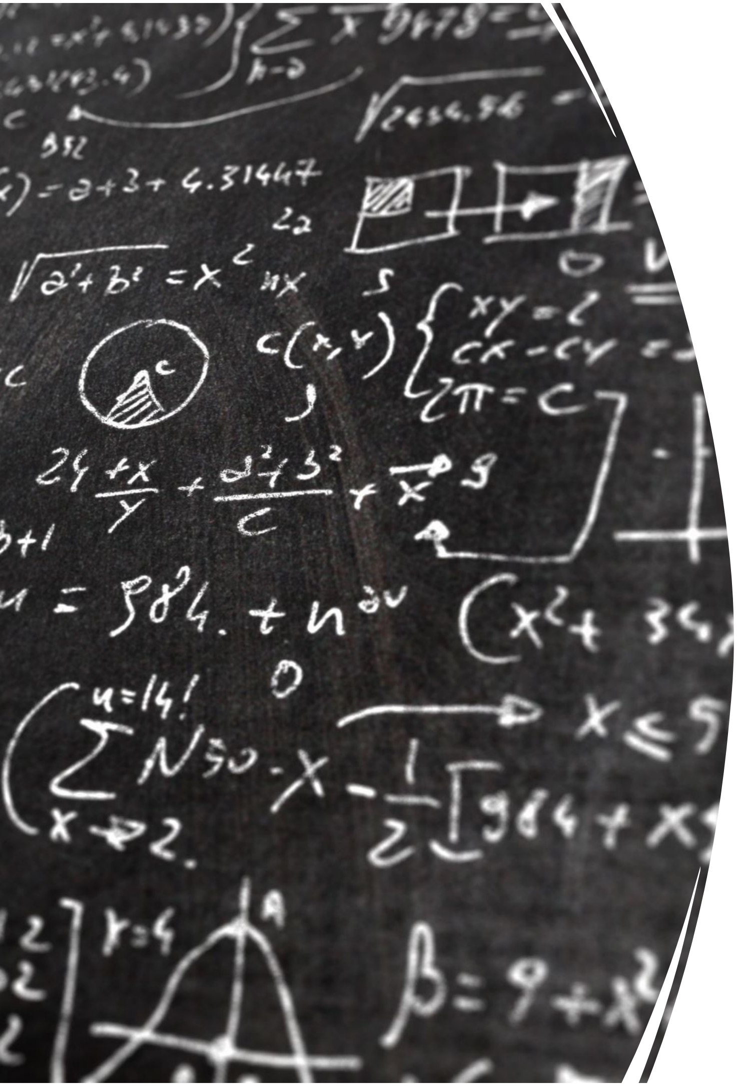


CS 383 – Computational Text Analysis

Lecture 3
Document Representation

Adam Poliak

01/25/2023

Slides adapted from Dan Jurafsky, Dirk Hovy

Announcements (1/2)

- Office Hours:
  - Thursdays 3-4:30pm

- HW00 late deadline tonight
- Reading01 late deadline tonight

- HW01 due Monday 01/30
  - Based on Monday’s lecture
- Reading02 released tonight, due Monday 01/30

Announcements (2/2)

- Monday 01/30 lecture
  - Lecture will start late, time tbd
  - Will use lab time for lecture too

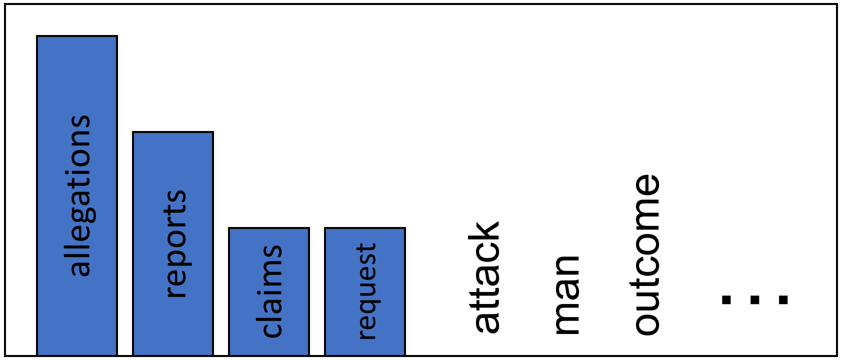
Outline

- LMs: smoothing, perplexity, <s>
- Document Representations
  - Document-Term Matrix
  - BoW
- Linear Algebra:
  - Vectors
  - Vector similarity
- tf-idf

The intuition of smoothing (from Dan Klein)

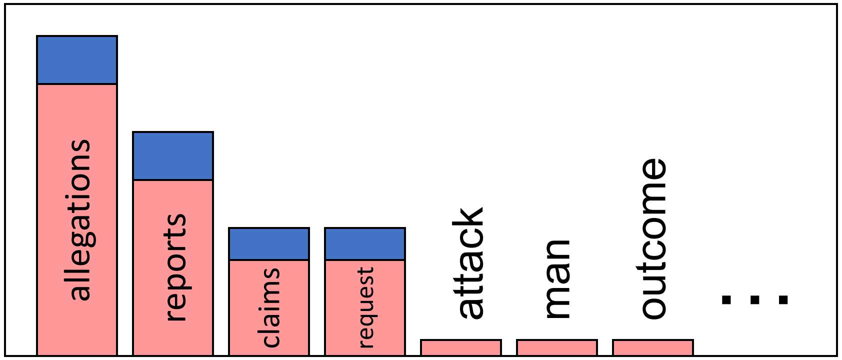
- When we have sparse statistics:

  P(w | denied the)
  3 allegations
  2 reports
  1 claims
  1 request
  7 total



- Steal probability mass to generalize better

  P(w | denied the)
  2.5 allegations
  1.5 reports
  0.5 claims
  0.5 request
  2 other
  7 total



Raw bigram probabilities

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Laplacian bigram probabilities

$$P_{Add-1}(w_i \mid w_{i-1}) = \frac{c(w_i, w_{i-1}) + 1}{c(w_{i-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Evaluating Language Models

Perplexity

$$Perplexity\ (w_1, w_2, w_3, \ldots, w_n) =$$

$$= P(w_1, w_2, w_3, \ldots, w_n)^{\frac{1}{n}}$$

$$= \sqrt[n]{\frac{1}{P(w_1, w_2, w_3, \ldots, w_n)}}$$

$P(w_1, w_2, w_3, \ldots, w_n)$ depends on the LM we use

The lower the perplexity, the better the model

Perplexity - implementation

$$Perplexity\,(w_1, w_2, w_3, \ldots, w_n) =$$

$$= P(w_1, w_2, w_3, \ldots, w_n)^{\frac{1}{n}}$$

$$= \sqrt[n]{\frac{1}{P(w_1,w_2,w_3,\ldots,w_n)}}$$

$$= \; e^{\frac{1}{n}\sum_{i=1}^{n} - \log P(w_1,w_2,w_3,\ldots,w_n)}$$

```
def perplexity(self, sentence, method):
    """
    Compute
    """
    return 2.0 ** (-1.0 * mean([method(context, word) for context, word in \
                                bigrams(self.tokenize_and_censor(sentence))]))
```

exponentiated average negative log-likelihood

Perplexity

$$Perplexity\ (w_1, w_2, w_3, \dots, w_n)\ =$$

$$= \sqrt[n]{\frac{1}{P(w_1, w_2, w_3, \dots, w_n)}}$$

The lower the perplexity, =>

the higher the probability =>

the model is less surprised by the text

This is based on P(M | T), i.e. we fit the model based on the training data

"less surprised" – based on just the training data, how shocked is the model when it sees $w_1, w_2, w_3, \dots, w_n$

Perplexity

$$= \sqrt[n]{\frac{1}{P(w_1, w_2, w_3, \ldots, w_n)}}$$

What we generally use for word sequence is the entire sequence of words in some test set. Since this sequence will cross many sentence boundaries, we need to include the begin- and end-sentence markers in the probability computation. We also need to include the end-of-sentence marker (but not the beginning-of-sentence marker) in the total count of word tokens N.

Training set: “a a b”

We add <s> and </s> to our example:

“<s> a a b </s>”

Unigram probabilities:

$P(< s >)$ $P(a)$ $P(b)$ $P(</s >)$

Training set: "a a b"

We add <s> and </s> to our example:

"<s> a a b </s>"

Unigram probabilities:

$P(< s >)$	$P(a)$	$P(b)$	$P(</s >)$
.2	.4	.2	.2

Training set: “a a b”

We add <s> and </s> to our example:

“<s > <s> a a b </s>”

Bigram probabilities:

$P(< s > \mid < s >)$ $P(a) \mid < s >$ $P(b \mid < s >)$ $P(</s > \mid < s >)$

$P(< s > \mid a)$ $P(a \mid a\ )$ $P(b \mid a)$ $P(</s > \mid a\ )$

$P(< s > \mid b)$ $P(\ a \mid b\ )$ $P(b \mid b)$ $P(</s > \mid ab)$

$P(< s > \mid </s >)$ $P(a) \mid </s >$ $P(b \mid </s >)$ $P(</s > \mid </s >)$

Training set: “a a b”

We add <s> and </s> to our example:

“<s > <s> a a b </s>”

Unigram probabilities now:

$P(< s >)$ $P(a)$ $P(b)$ $P(</s >)$

Training set: “a a b”

Dont treat c(<s>) as a regular token

We add <s> and </s> to our example:

“<s> <s> a a b </s>”

Unigram probabilities:

	$P(<s>)$	$P(a)$	$P(b)$	$P(</s>)$
Now:	.33	.33	.1667	.1667
Before:	.2	.4	.2	.2

Training set: “a a b”

Dont treat c(<s>) as a regular token

We add <s> and </s> to our example:

“<s> <s> a a b </s>”

Unigram probabilities:

	$P(<s>)$	$P(a)$	$P(b)$	$P(</s>)$
Correct:	-	.5	.25	.25

Why not include <s> in our counts?

Why do we include <s>?

Why do we include </s>?

Generating text perspective?

What happens if we use unigram to generate text?

Outline

- LMs: smoothing, perplexity, <s>, MLE vs EM
- Document Representations
  - Document-Term Matrix
  - BoW
- Linear Algebra:
  - Vectors
  - Vector similarity
- tf-idf

Maximum Likelihood Estimates

- The maximum likelihood estimate
  - of some parameter of a model M from a training set T
  - maximizes the likelihood of the training set T given the model M

Parameters are:

n-gram probabilities

Approach 2 – Combine the ‘grams

Context specific weights

Jelinek-Mercer smoothing (1980)

Lambdas are parameters too

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) + \lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) + \lambda_3(w_{n-2}^{n-1})P(w_n)$$

Approach 2 – Combine the ‘grams

Context specific weights

Jelinek-Mercer smoothing (1980)

Lambdas are parameters too

How do we decide what values our lambdas should be?

Approach 2 – Combine the ‘grams

Context specific weights

EM (expectation maximization)

Lambdas are parameters too

How do we decide what values our lambdas should be?

Split our data into train and evaluation sets

try different lambdas, compare model’s perplexity on the evaluation set

Outline

- LMs: smoothing, perplexity, <s>
- Document Representations
  - Document-Term Matrix
  - BoW
- Linear Algebra:
  - Vectors
  - Vector similarity
- tf-idf

Recap so far

The first class was all about ____ ____

Recap so far

The first class was all about counting ___

Recap so far

The first class was all about counting words

2nd class was about the power of counting words.

By counting words we can ___ ___

Recap so far

The first class was all about counting words

2nd class was about the power of counting words.

By counting words we can ___ ___

learn about language

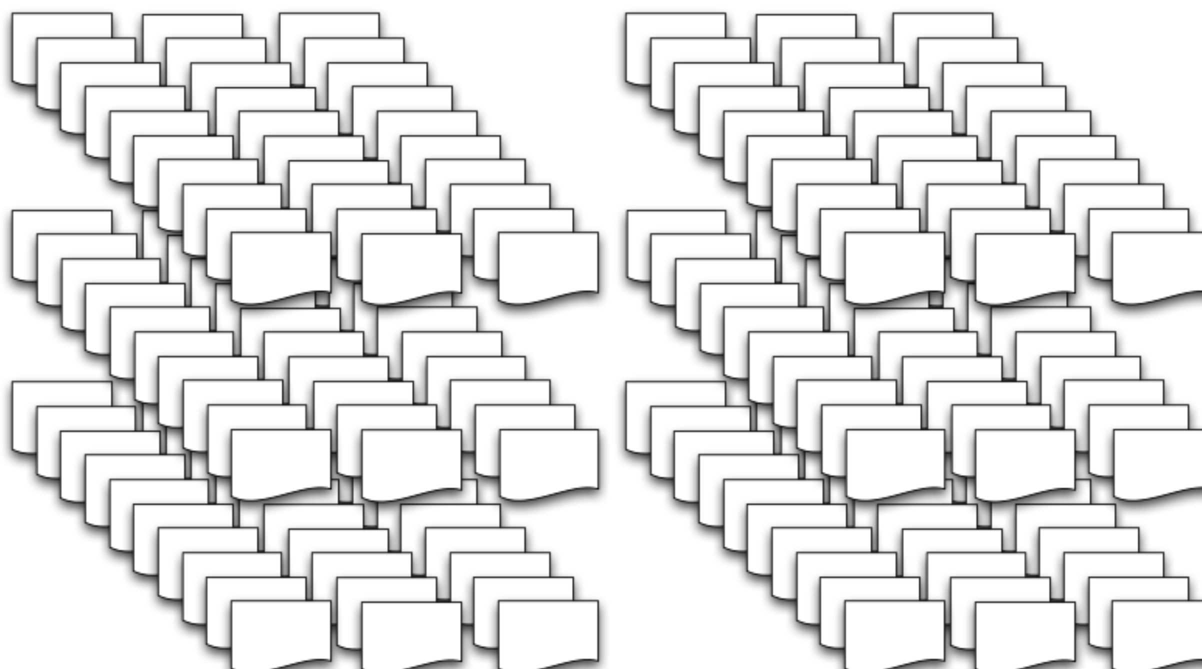
generate language

categorize language

Documents & Corpora

Terminology - Corpus

- **Corpus**:
  - A collection of documents
  - ***Corpora*** – plural of corpus



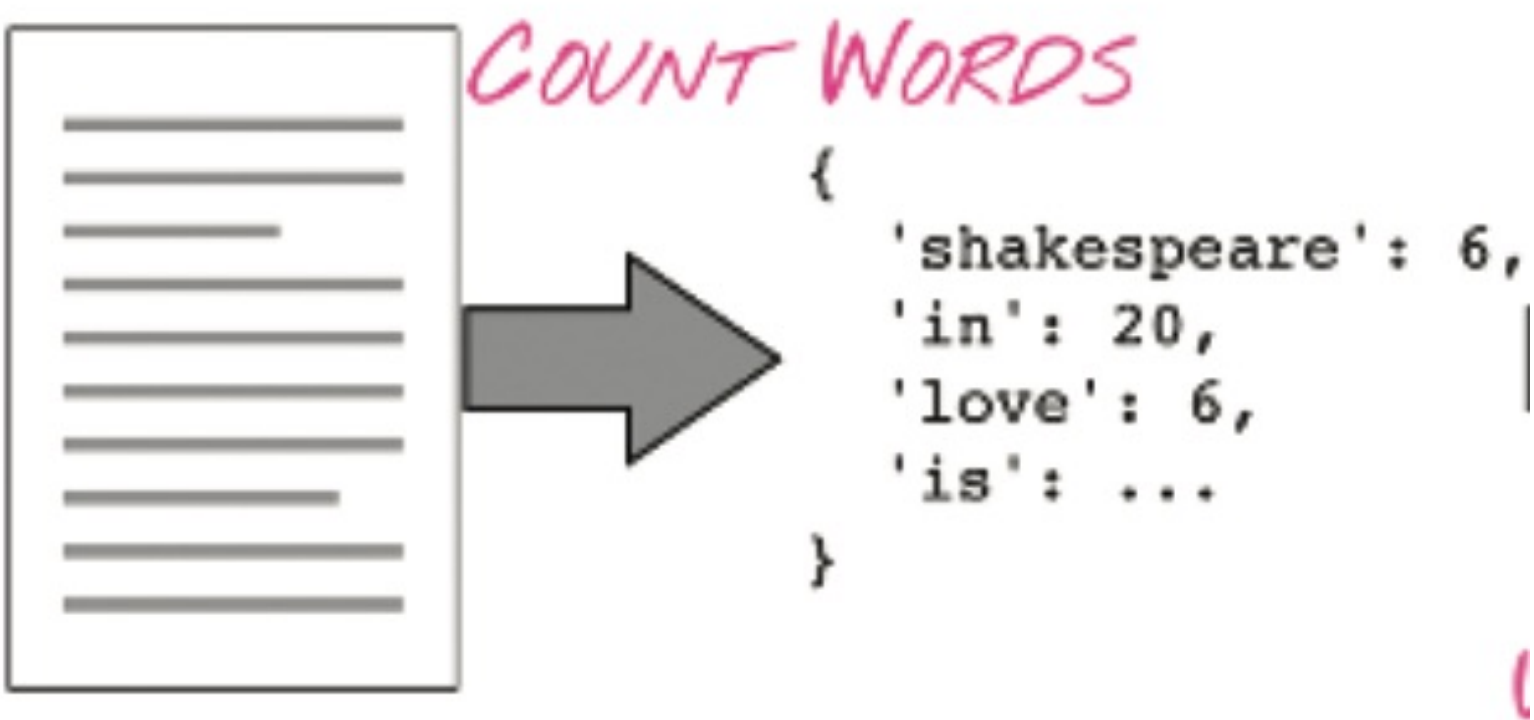
Terminology - Document

- **Document**:
  - Often unit of text of interest (dependent on RQ)
  - Often represents one data point

- Examples:
  - Book
  - Chapter
  - News article
  - Tweet
  - Product Review
  - ….

How do we represent documents?

Dictionaries of word counts



Often called ***Bag of Words***

Bag of Words – Start with document

Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up… very very very good movie.

Bag of Words – Break document into words

Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up... very very very good movie.

Bag of Words – compute word counts

Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up… very very very good movie.

('the', 8),
(',', 5),
('very', 4),
('.', 4),
('who', 4),
('and', 3),
('good', 2),
('it', 2),
('to', 2),
('a', 2),
('for', 2),
('can', 2),
('this', 2),
('of', 2),
('drama', 1),
('although', 1),
('appeared', 1),
('have', 1),
('few', 1),
('blank', 1)
…..

Bag of Words

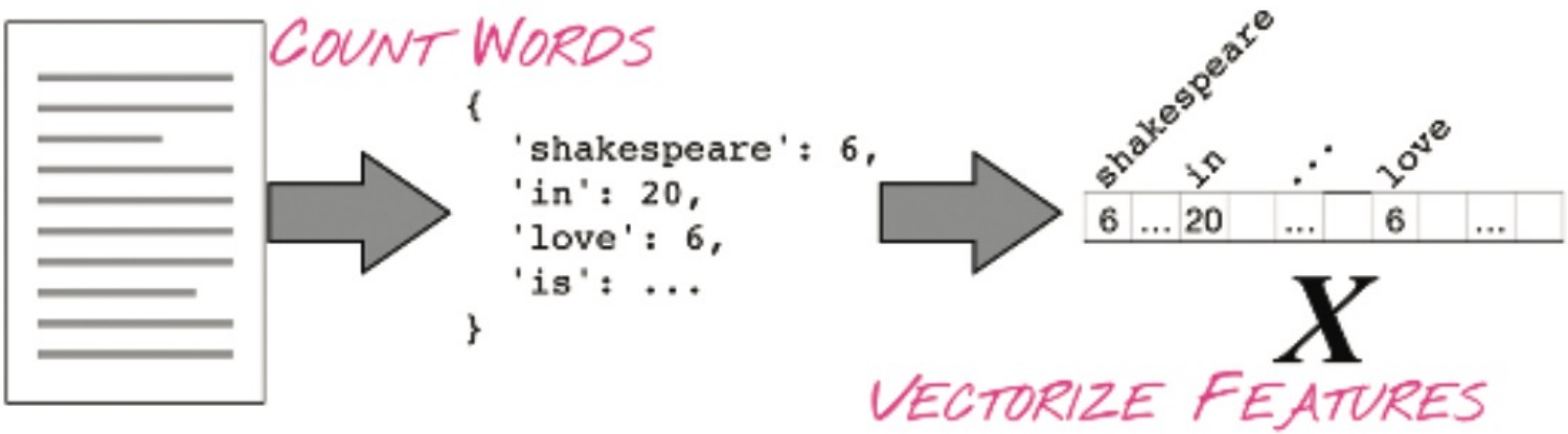
Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up… very very very good movie.

('the', 8),
(',', 5),
('very', 4),
('.', 4),
('who', 4),
('and', 3),
('good', 2),
('it', 2),
('to', 2),
('a', 2),
('for', 2),
('can', 2),
('this', 2),
('of', 2),
('drama', 1),
('although', 1),
('appeared', 1),
('have', 1),
('few', 1),
('blank', 1)
…..

Document vectors

Document vectors

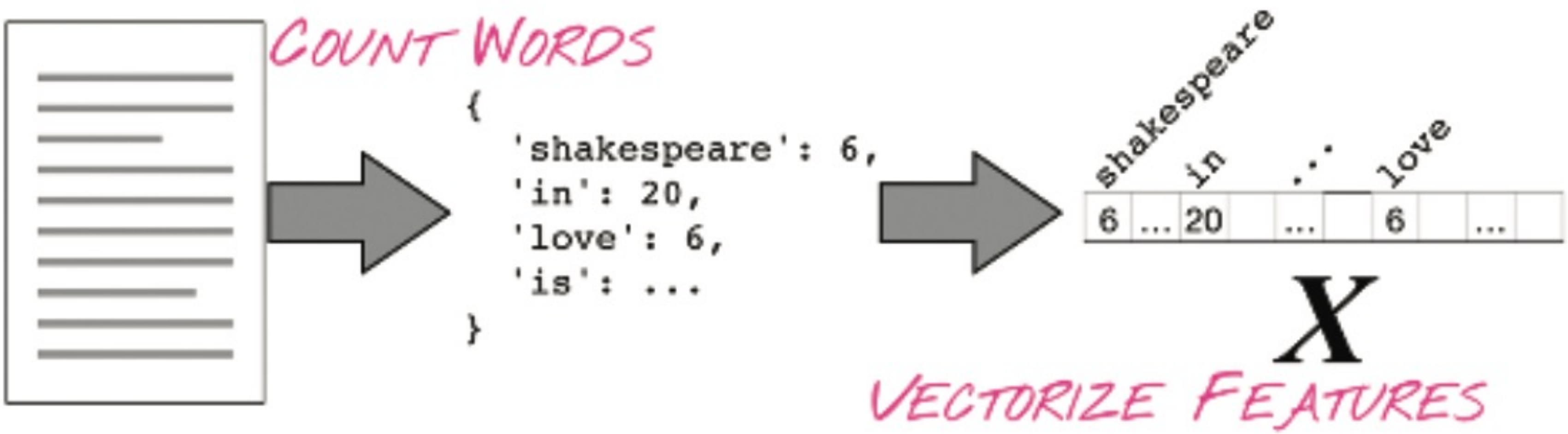
- Vector is just an array of numbers



- Index represents a word
- Value represents ….

Document vectors

- Vector is just an array of numbers



- Index represents a word
- Value represents something about that word
  - For now, unigrams

Outline

- LMs: smoothing, perplexity, <s>
- Document Representations
  - Document-Term Matrix
  - BoW
- Linear Algebra:
  - Vectors
  - Vector similarity
- tf-idf

Vectors

Physics:

arrow pointing in space

it has a length, and a direction its pointing

CS:

ordered lists of numbers

number of dimensions is size of the list

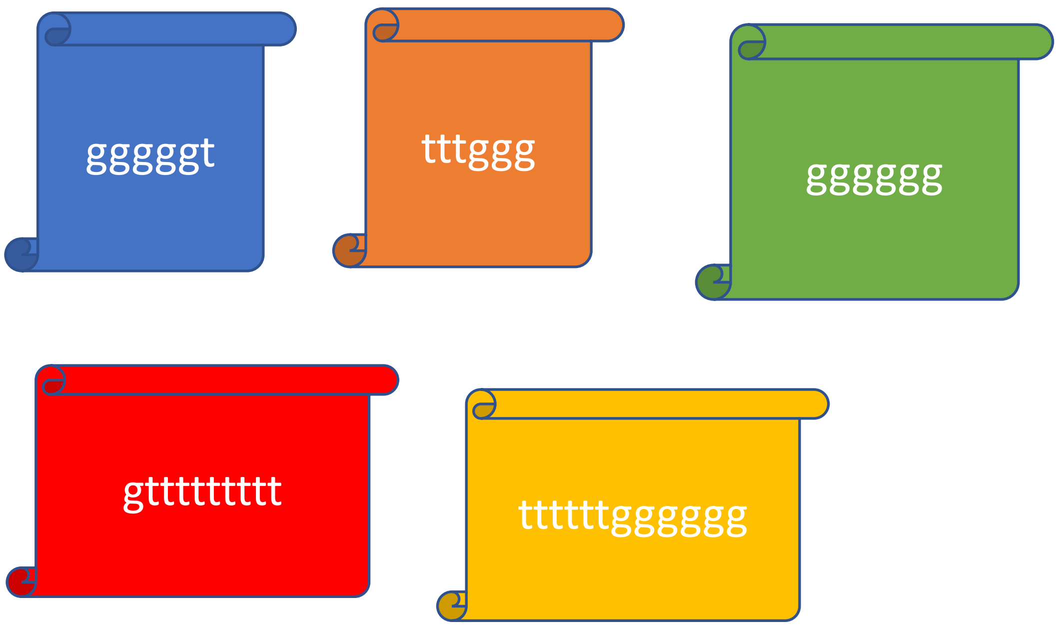
Math:

we can add them together

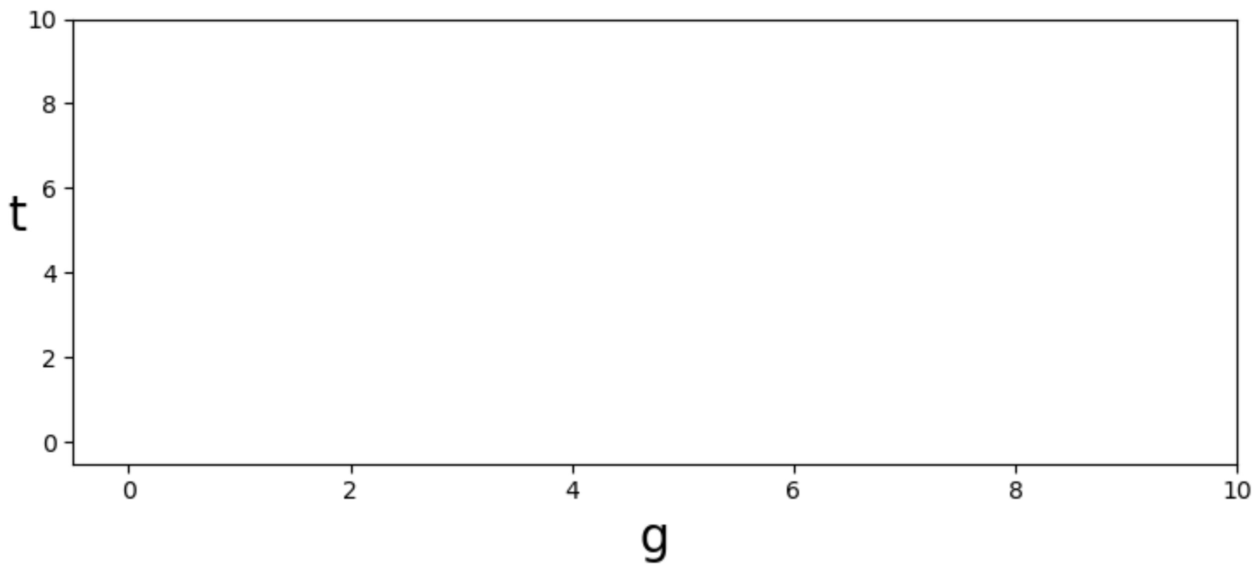
we can multiply them by a number

Document as a Vector

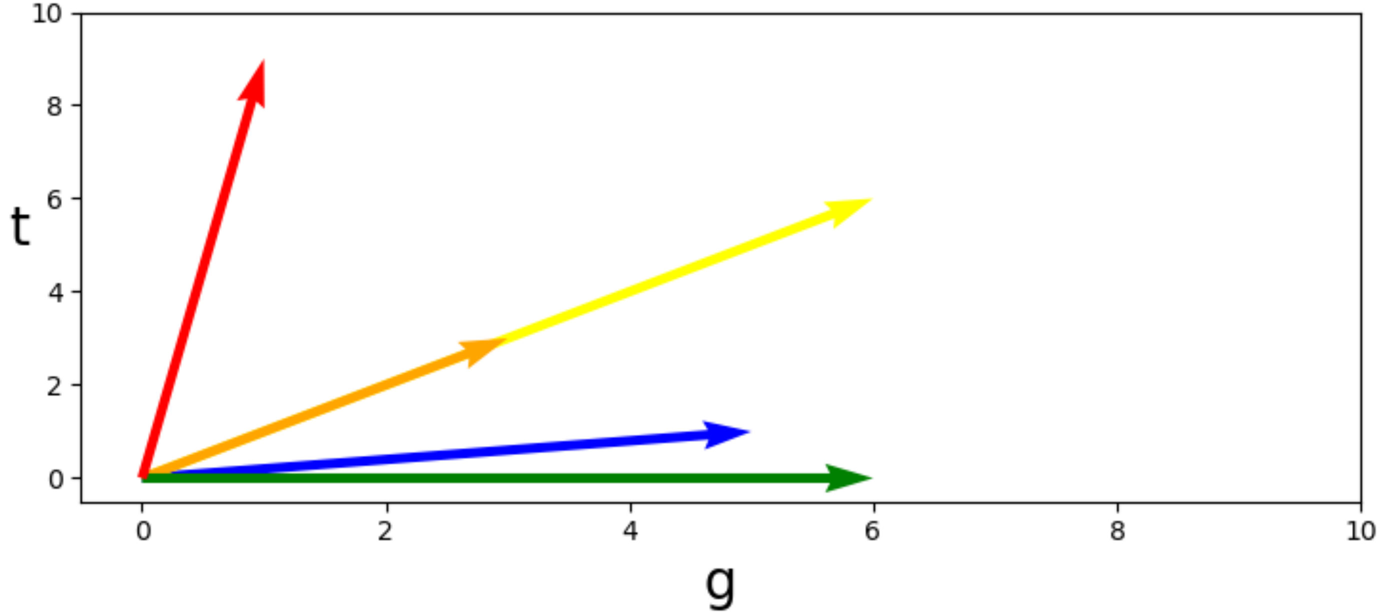
Vocabulary is {“g”, “t”}



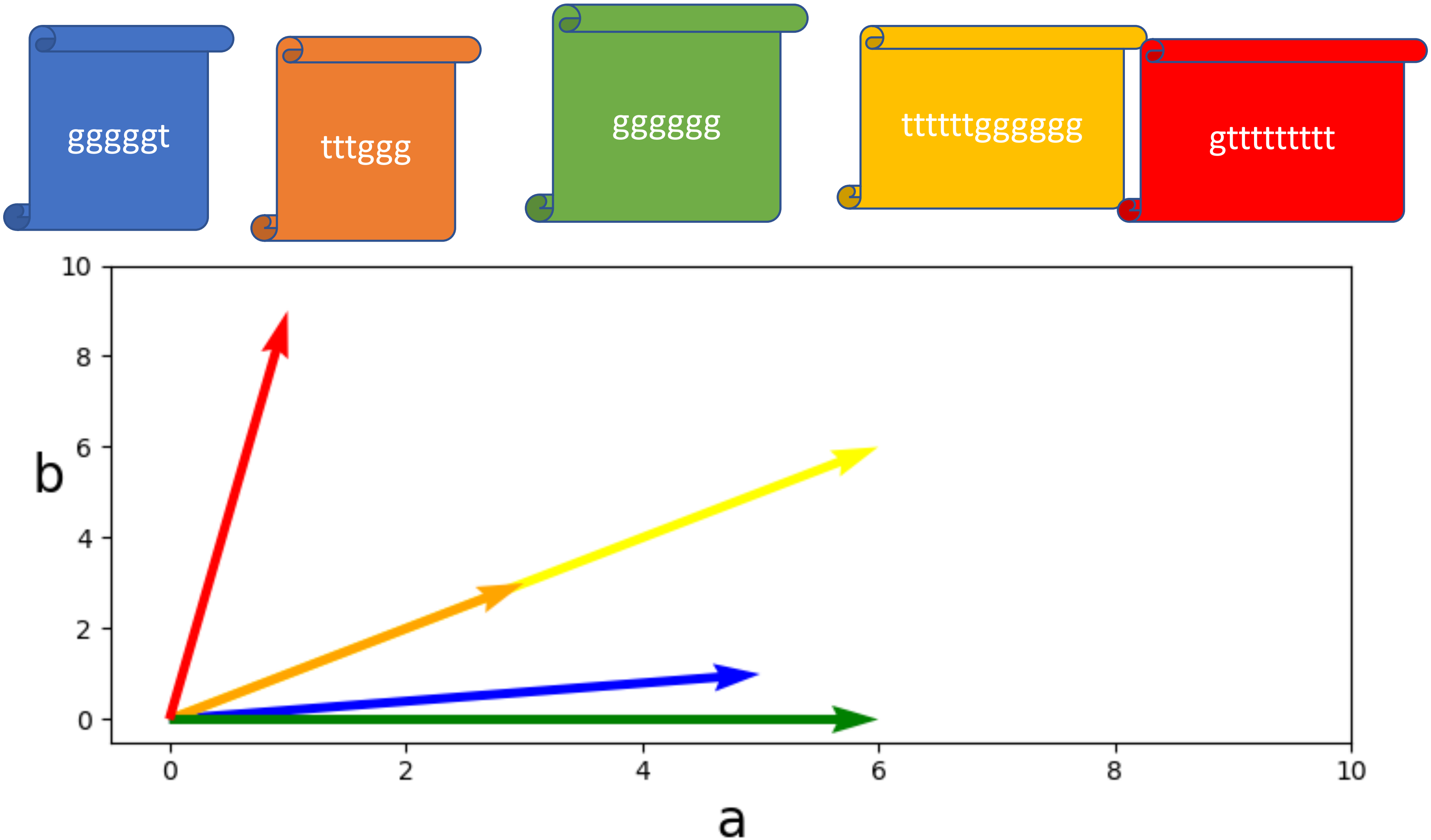
Document as a Vector



Document as a Vector



Which two are the most similar?



Vector similarity

Dot product of **a** and **b**:

$$\boldsymbol{a} * \boldsymbol{b} = \sum_{i}^{n} a_i b_i$$

$*$

$$= \#g_{blue} * \#g_{orange} + \#t_{blue} * \#t_{orange}$$

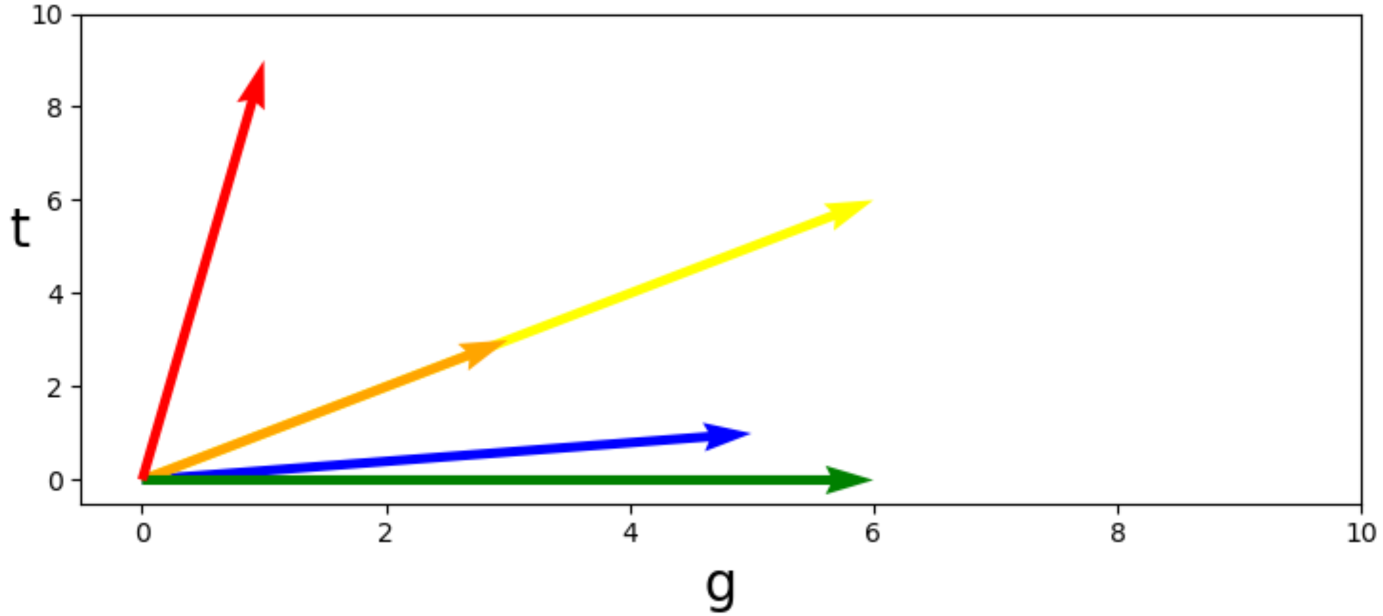
$$= 5 * 3 + 1 * 3$$

$$15 * 3$$

$$45$$

Issues with dot product

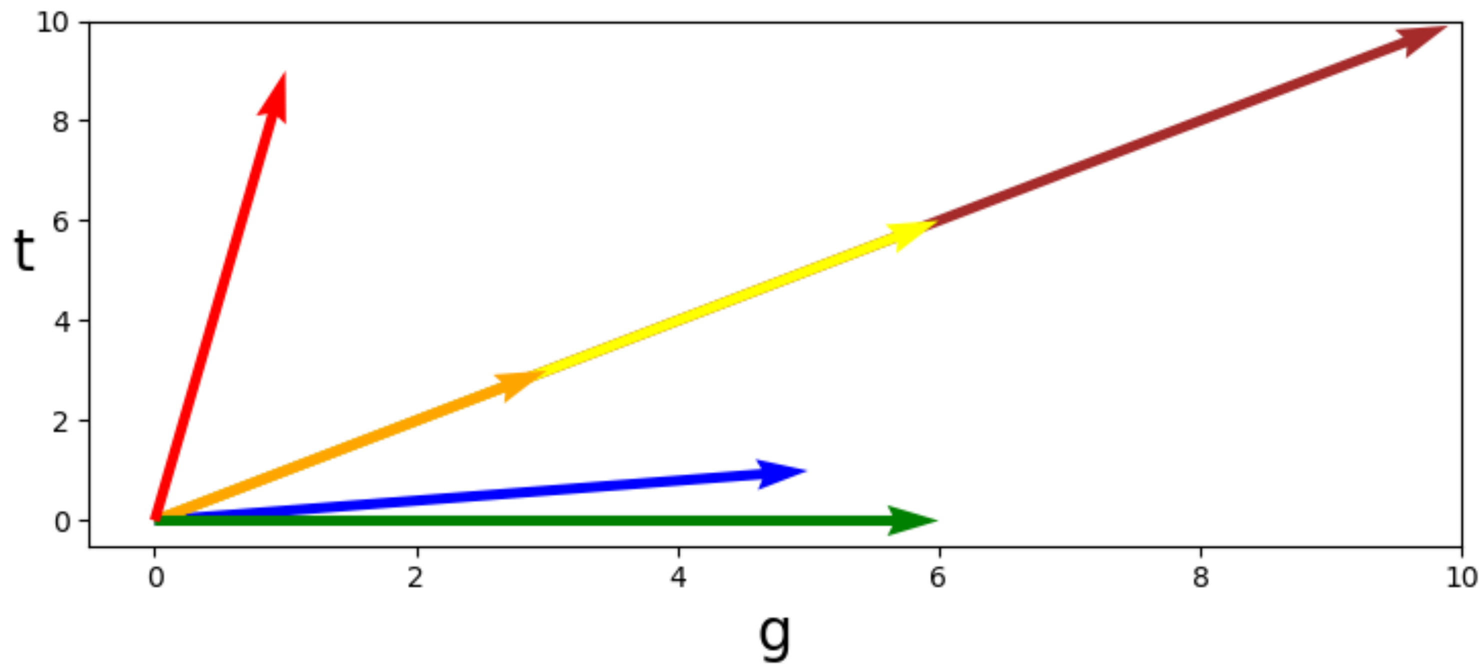
$$\boldsymbol{a} * \boldsymbol{b} = \sum_{i}^{n} a_i b_i$$



Issues with dot product

$$\boldsymbol{a} * \boldsymbol{b} = \sum_{i}^{n} a_i b_i$$

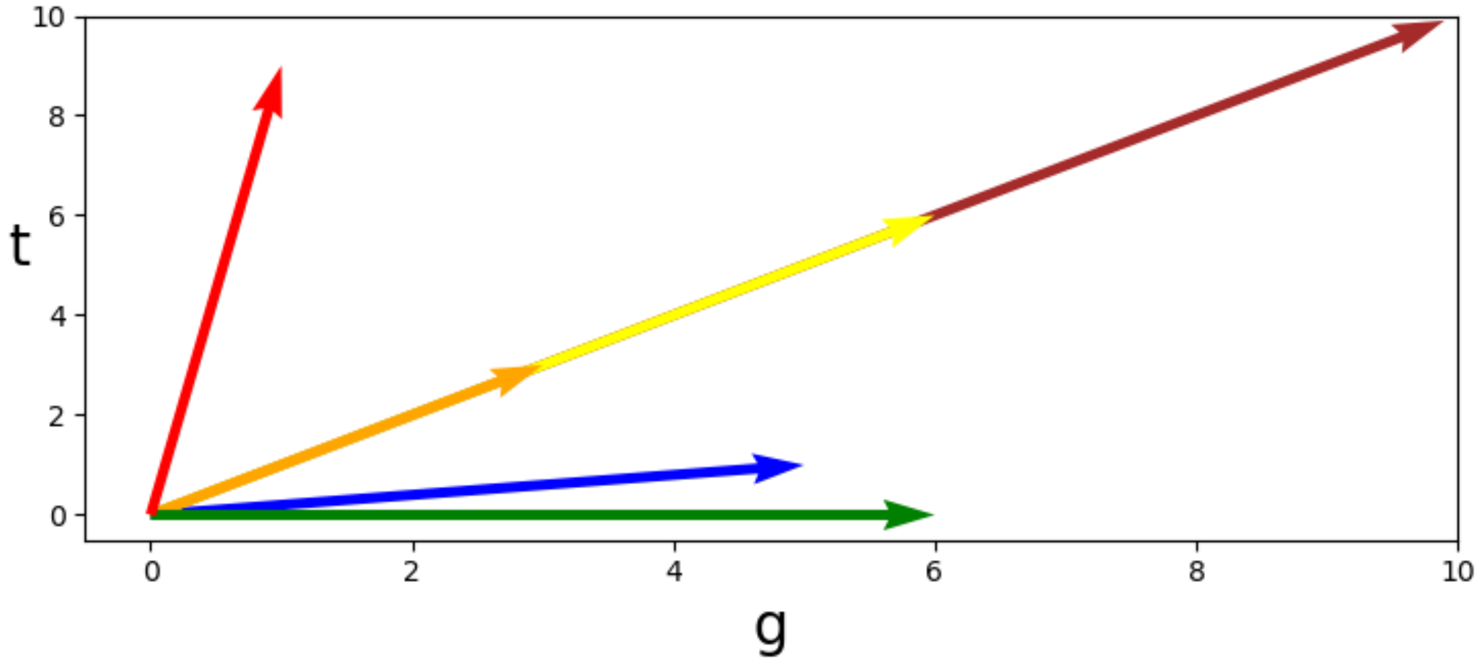
Which is most similar:
(brown, yellow), (brown, orange),
or (yellow, orange)?



Issues with dot product

$$\boldsymbol{a} * \boldsymbol{b} = \sum_{i}^{n} a_i b_i$$

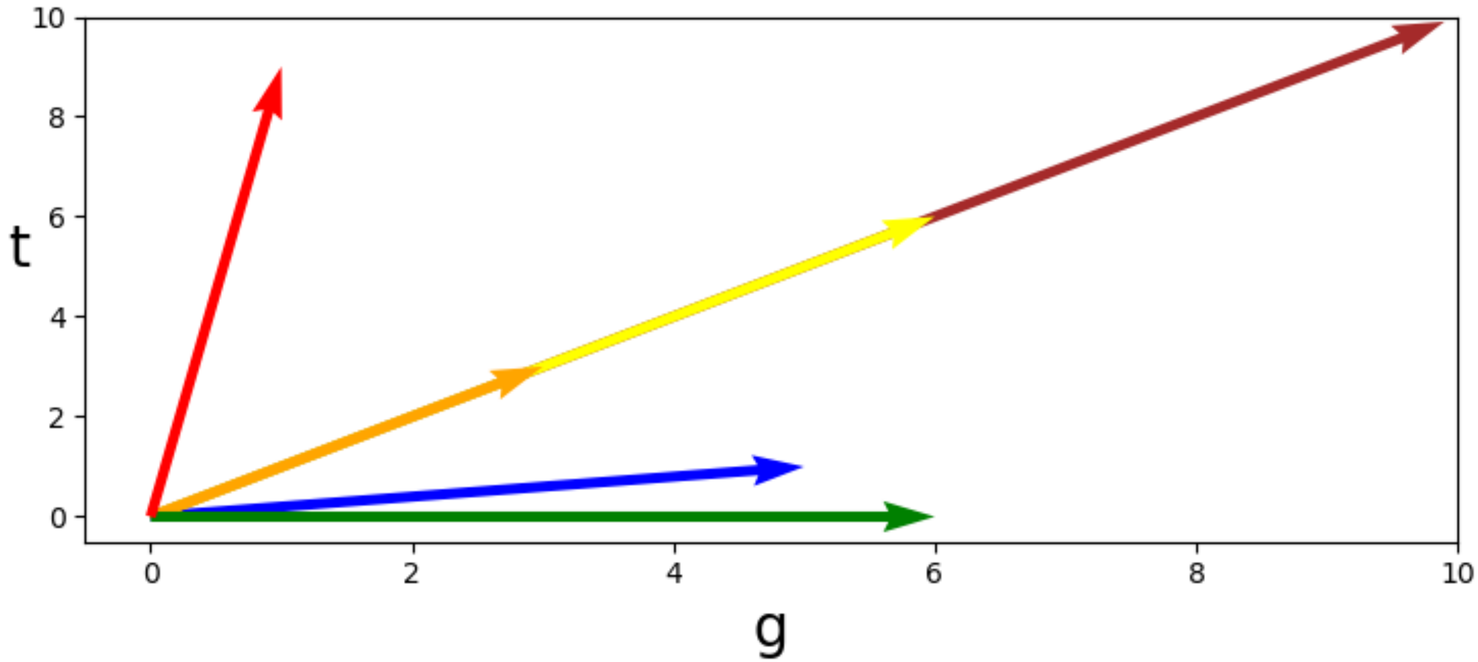
Which is most similar:
(brown, yellow), (brown, orange),
or (yellow, orange)?



Issues with dot product

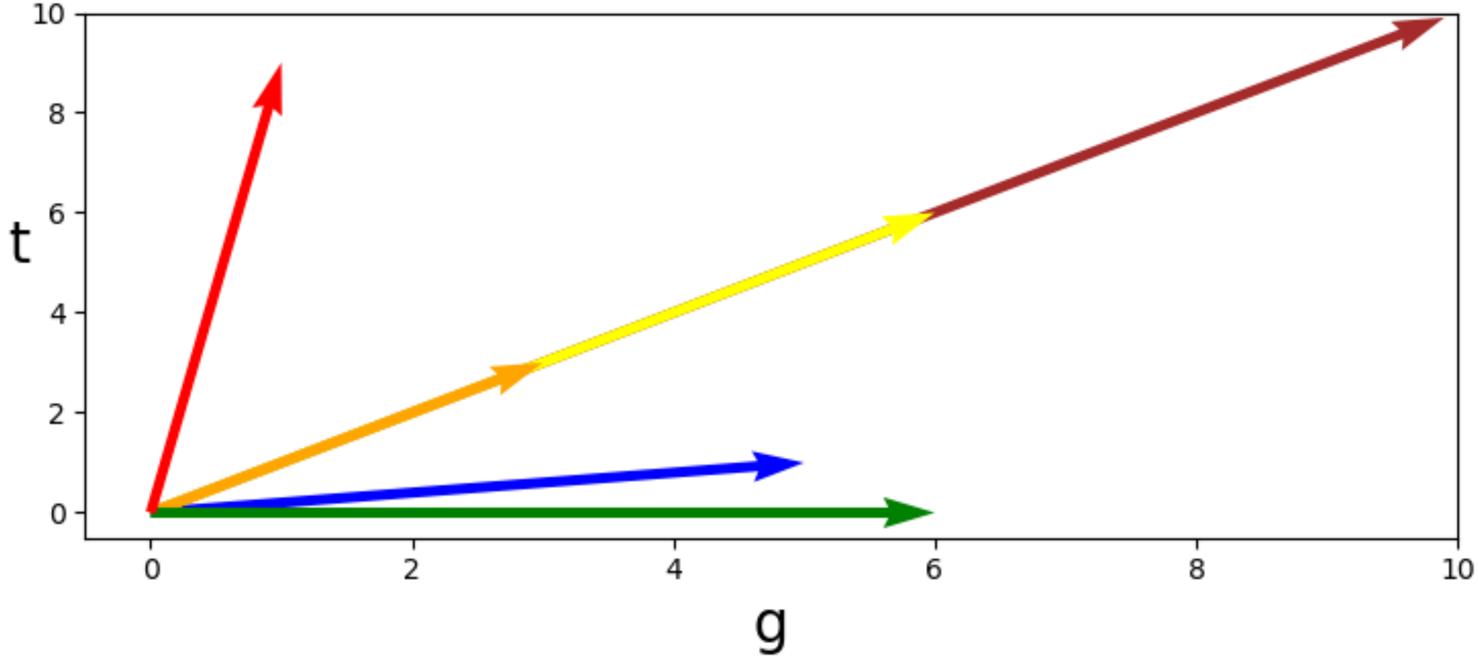
$$\boldsymbol{a} * \boldsymbol{b} = \sum_{i}^{n} a_i b_i$$

More weight given to longer vectors (documents)



Solution – normalize by length

$$\frac{\boldsymbol{a} * \boldsymbol{b}}{|\boldsymbol{a}||\boldsymbol{b}|}$$



How to compute the length of a vector

How long is

Option 1:

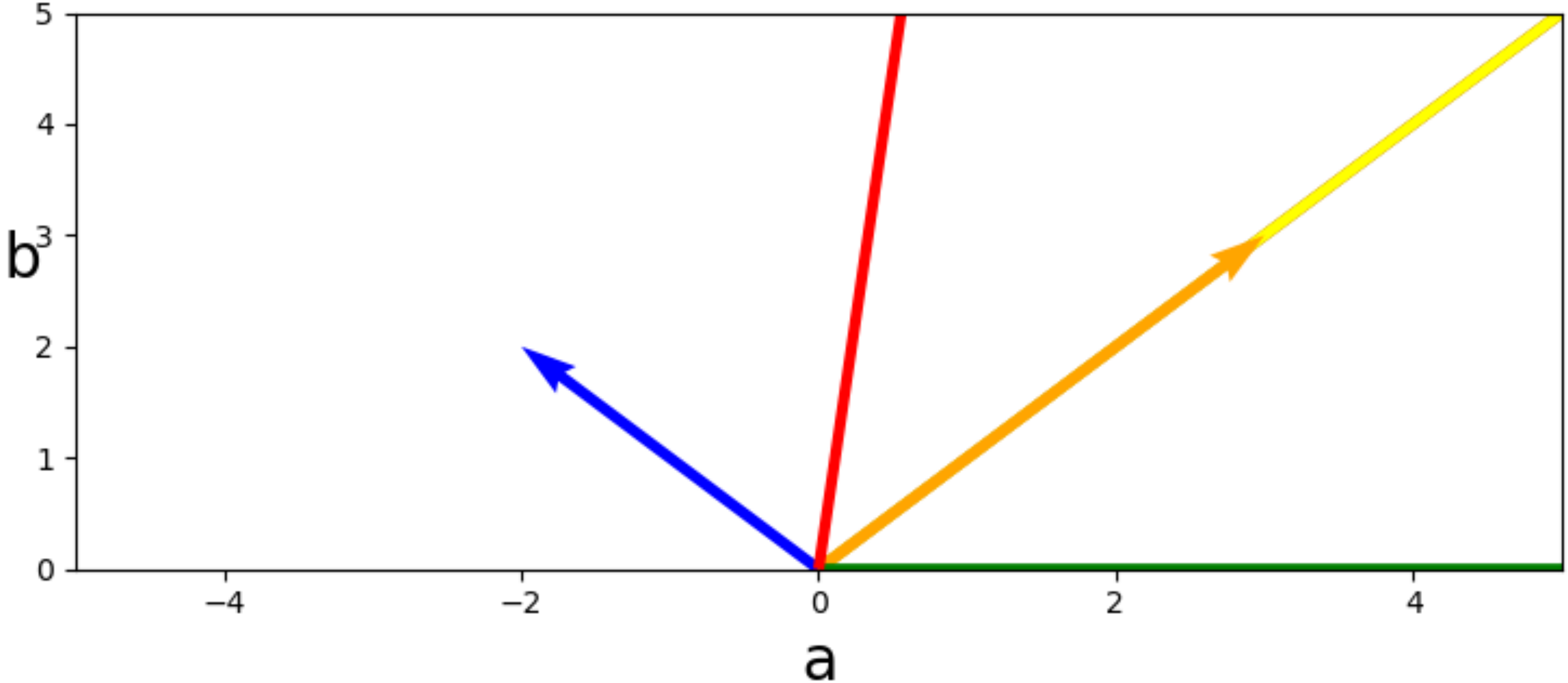
Just add the number of g's and the number of t's

In terms of documents, its just length of document

How to compute the length of a vector

What's the length of blue with option 1?

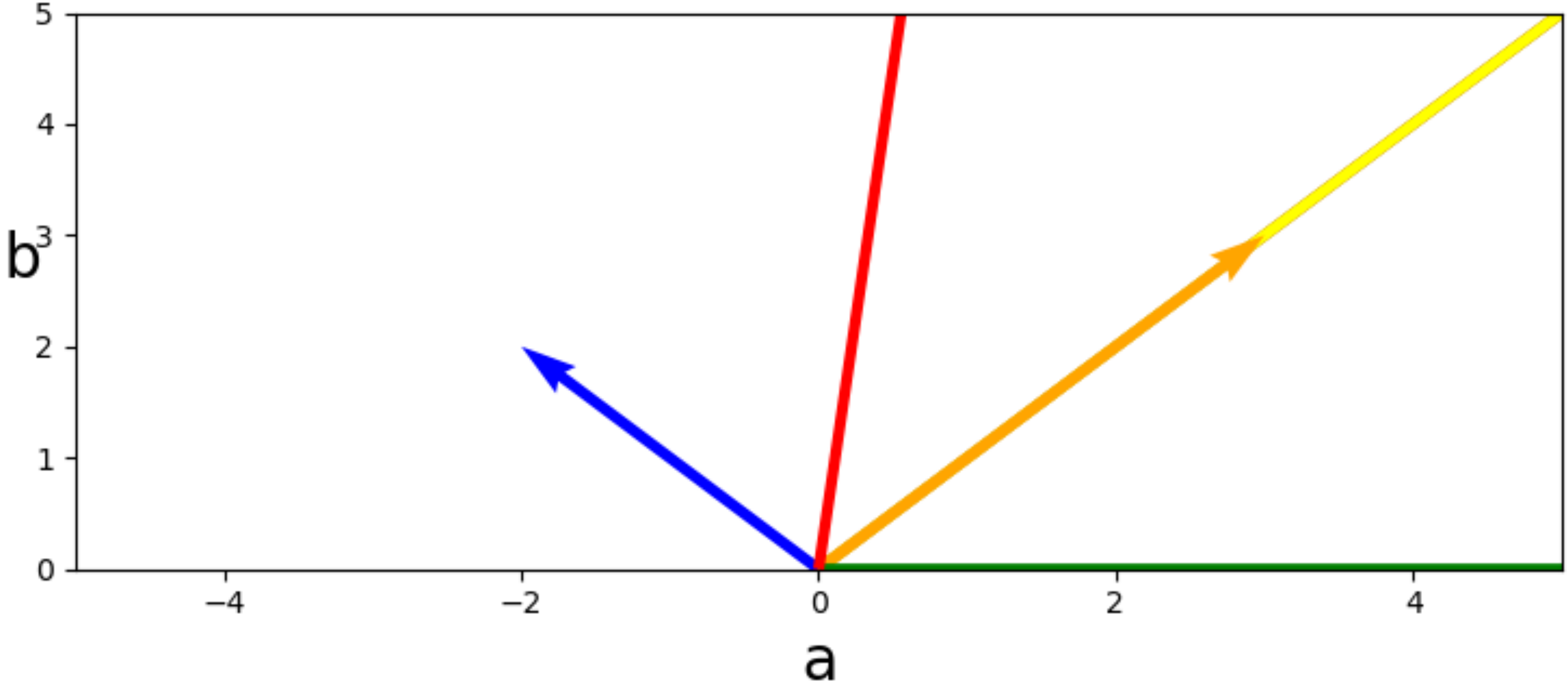
-1 + 1 = 0



How to compute the length of a vector

Option 2: add absolute values

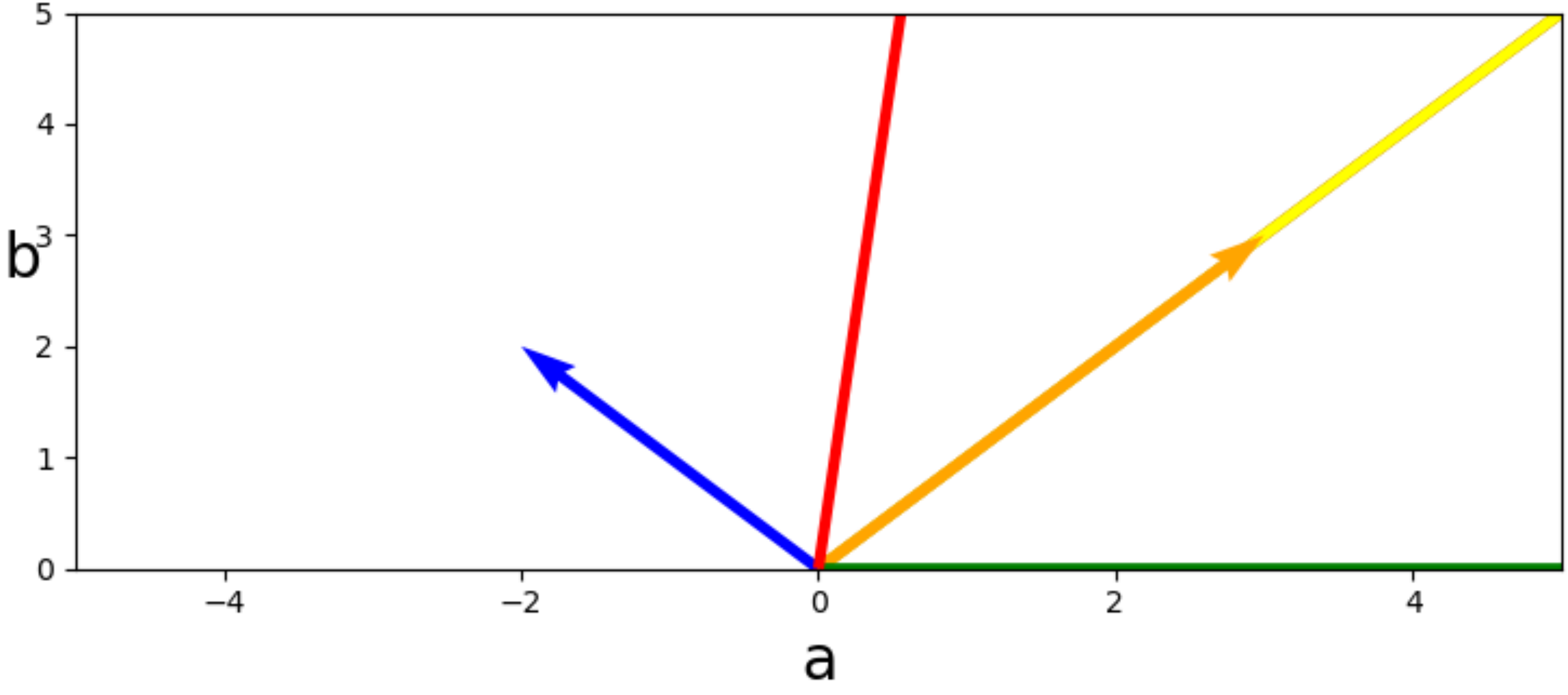
|-1| + |1| = 2



How to compute the length of a vector

Option 3: add squared values, then take square root

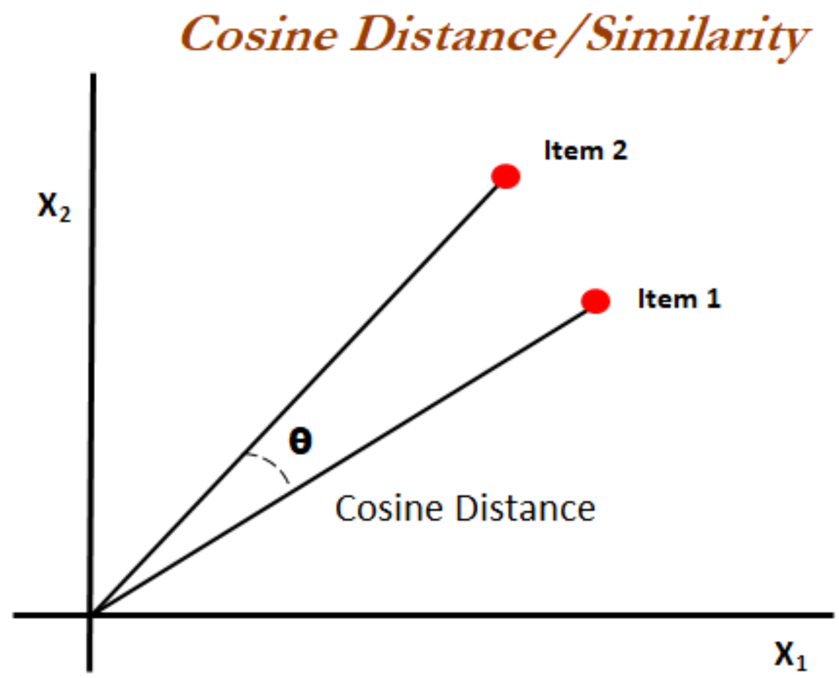
$$-1^2 \ + 1^2 = \sqrt{2}$$



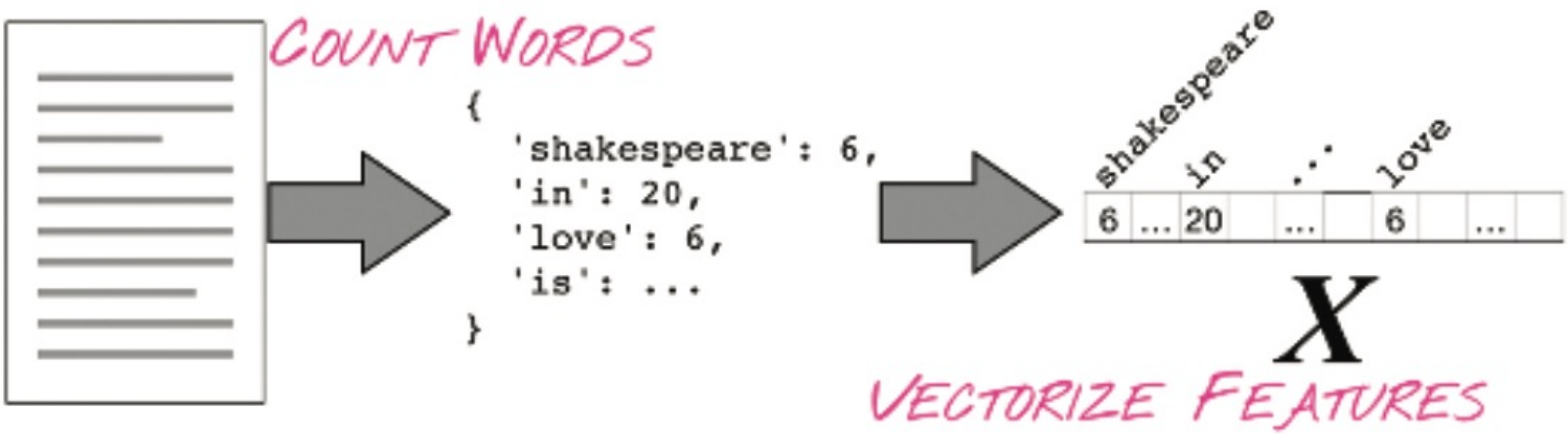
Cosine similarity

$$\frac{\boldsymbol{a} * \boldsymbol{b}}{|\boldsymbol{a}||\boldsymbol{b}|} = \cos\theta$$

Normalized dot product is the same as the cosine of the angle between the vectors



Document vectors

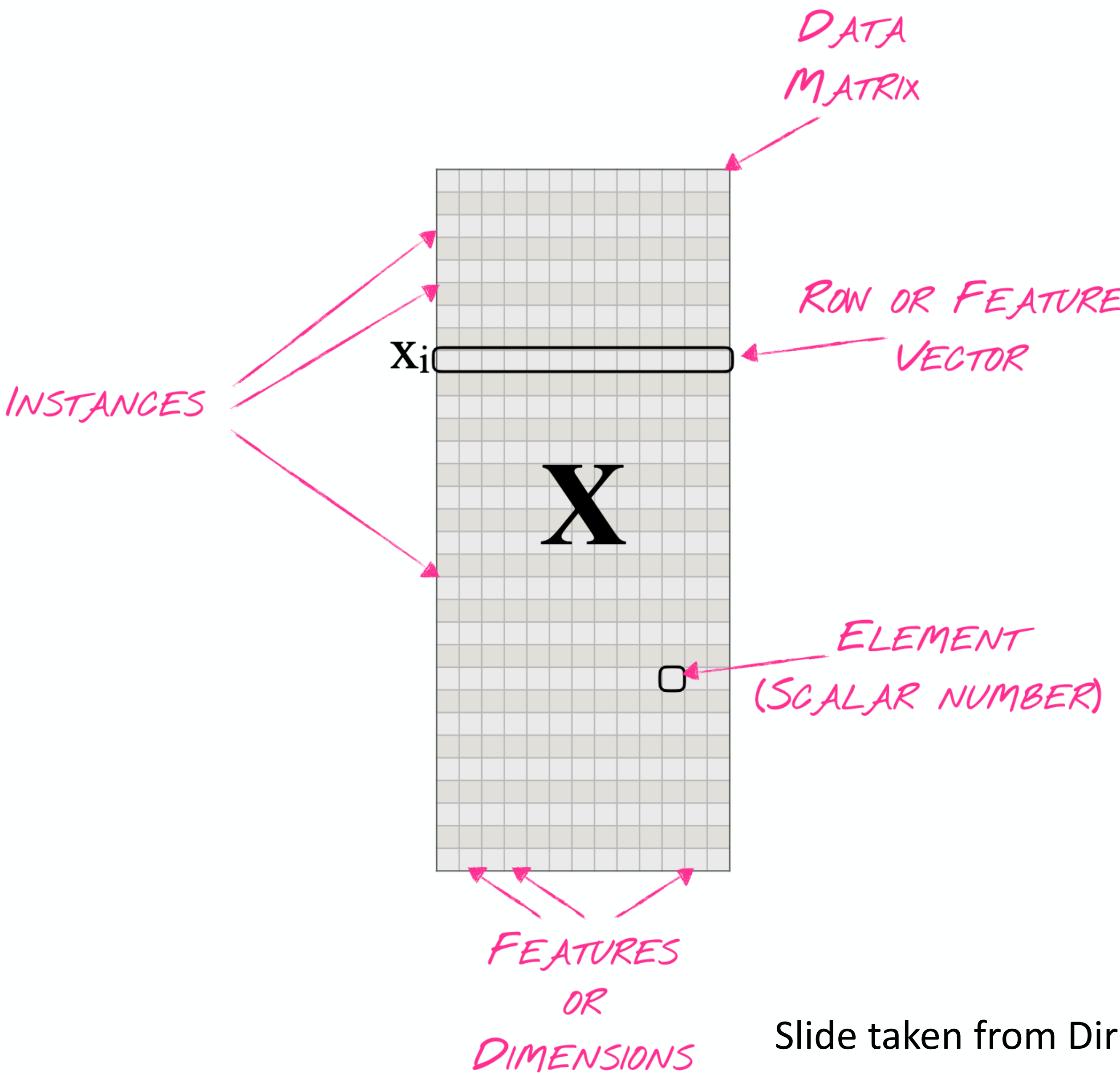


Finding most similar documents?

CTA examples of finding most similar documents

What did we use to represent documents?

Document Matrix



Slide taken from Dirk Hovy

Recap so far

The first class was all about counting words

2nd class was about the power of counting words.

By counting words we can ___ ___

learn about language

generate language

categorize language

group documents

What to count?
How to count?

Next lecture

HW02