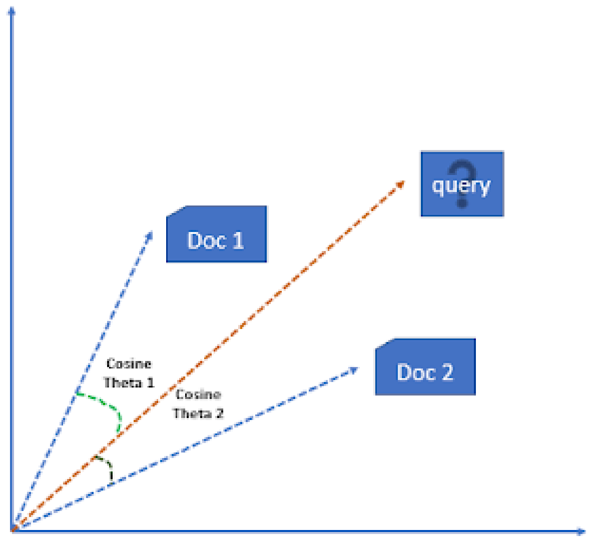
Document Similarity

Natural Language Processing



Agenda

- What is Document Similarity
- Methods to measure Document Similarity
- Cosine Similarity Method

Goal

- Given a set of documents and search term(s)/query we need to retrieve relevant documents that are **similar to the search query**.

Information Retrieval

- One of the fundamental problems with having a lot of data is finding what you’re looking for.
- This is called information retrieval.

Document

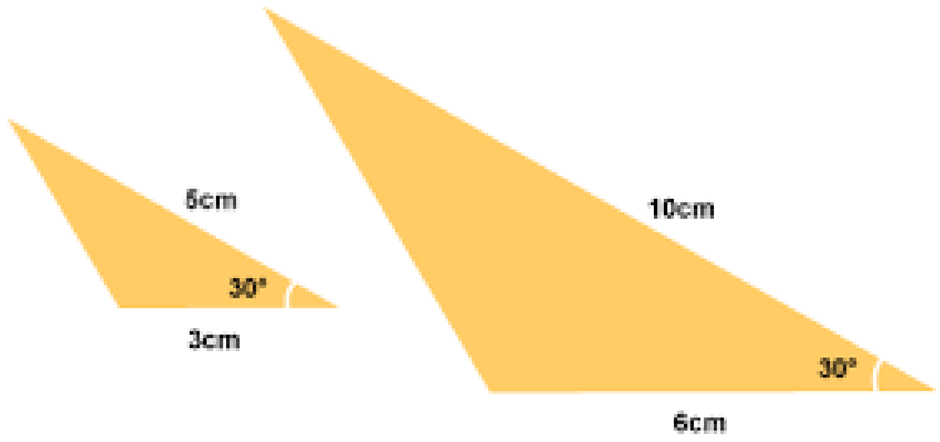
- A document is a piece of electronic matter that provides information or evidence or that serves as an official record.
- Examples of different document
  - Book
  - Online article
  - Newspaper article
  - Photography
  - Letter
  - Movie

Document Comparision

- Comparison between things, like clothes, food, products and even people, is an integral part of our everyday life.
- It is done by assessing similarity (or differences) between two or more things.
- Apart from its usual usage as an aid in selecting a thing-product, the comparisons are useful in searching things ‘similar’ to what you have and in classifying things based on similarity.

Similarity

- Similarity is the state or fact of being similar while similar is referring to a resemblance in appearance, character, or quantity, without being identical
- Example
  - Rectangles which are similar do not necessarily have the same size.



Document Similarity

- Document similarity is a metric defined over a set of documents, where the idea of distance between them is based on the likeness of their meaning or semantic content.
- Set of attributes to be used to compare documents :
    - Author
    - Category
    - Content

Match Relevant Documents

- a measure of similarity that can be used to
  - compare documents or
  - provide a ranking of documents
- with respect to a given vector of query words.

SIMILARITY VS. EXACT

- Identical duplicate documents are generally very easy to detect, for example, using a simple hash algorithm.
- Finding documents that are similar, or near-duplicates, requires more effort.

Similarity

- Similarity is the state or fact of being similar
- similar refers to a resemblance in
  - appearance,
  - character, or
  - quantity,
- without being identical

Methods to measure Document Similarity

- Jacard similarity measure
- Metric similarity measure
- Euclidean Distance measure
- Cosine similarity measure

Cosine similarity measure

- Cosine similarity documents are represented as term vectors.
- The similarity of two documents corresponds to the correlation between the vectors.
- This is quantified as the cosine of the angle between vectors - known as cosine similarity.
- Cosine similarity is one of the most popular similarity measure applied to text documents

Cosine Similarity Method

- Vector Space Model
- TF , TDF …
- Cosine Similarity Method Calculation

Vector Space Model (VSM)

- Vector Space Model (VSM) is a way of representing documents through the words that they contain
- It is a standard technique in Information Retrieval

VSM – Working

- Each document is broken down into a word frequency table
- The tables are called vectors and can be stored as arrays
- A vocabulary is built from all the words in all documents in the system
- Each document is represented as a vector based against the vocabulary

VSM – Working

- Each document is broken down into a word frequency table
- The tables are called vectors and can be stored as arrays

Document A

“A dog and a cat.”

a	dog	and	cat
2	1	1	1

Document B

“A frog.”

a	frog
1	1

Example

- A vocabulary is built from all the words in all documents in the system
- Each document is represented as a vector based against the vocabulary

Document A: “A dog and a cat.”

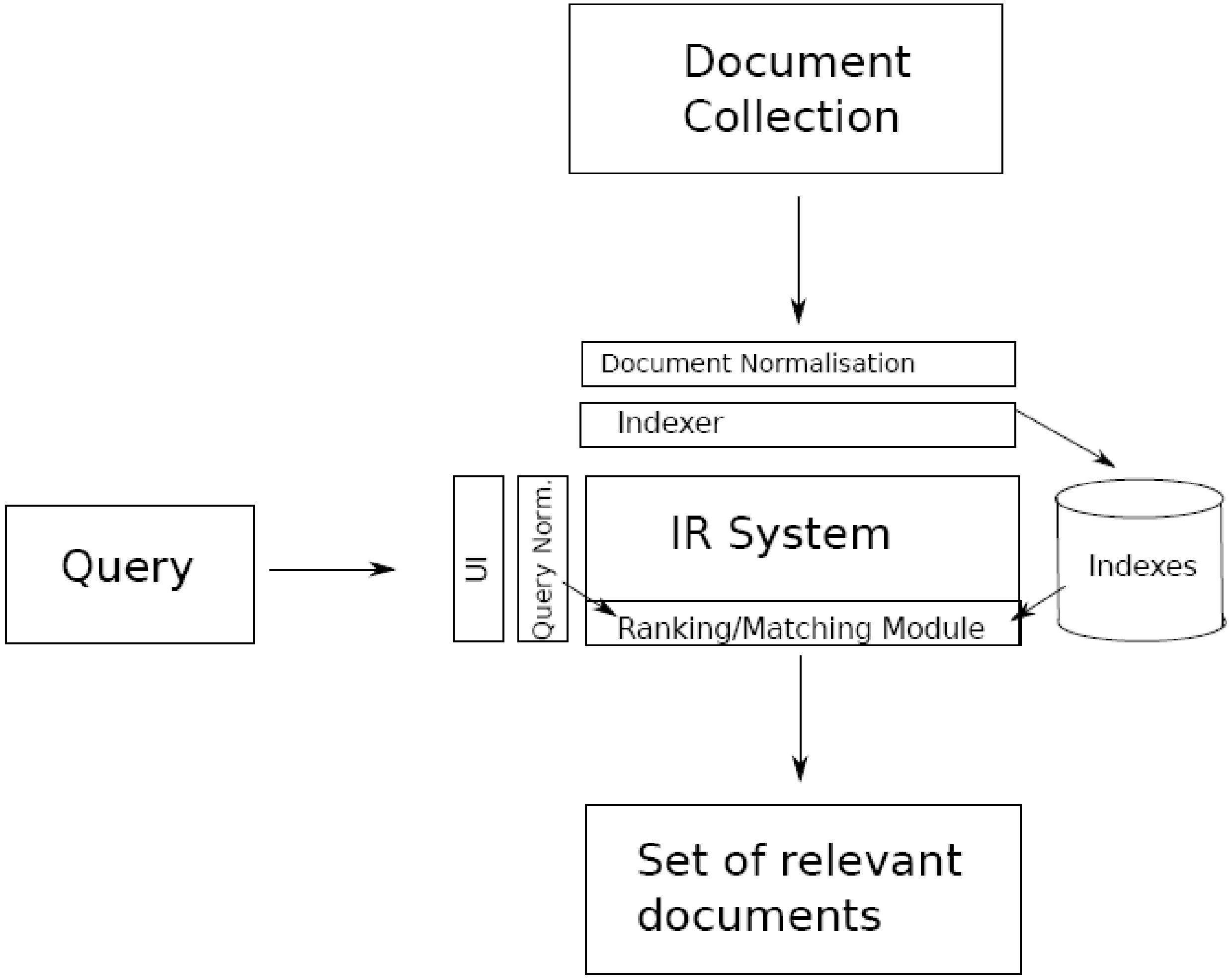
a	and	cat	dog	frog
2	1	1	1	0

Vector: (2,1,1,1,0)

Document B: “A frog.”

a	and	cat	dog	frog
1	0	0	0	1

Vector: (1,0,0,0,1)


Document Collection
Document Normalisation
Indexer
Query
UI
Query Norm.
IR System
Ranking/Matching Module
Indexes
Set of relevant documents

Queries

- Queries can be represented as vectors in the same way as documents:
  - Dog = (0,0,0,1,0)
  - Frog = ( )
  - Dog and frog = ( )

Unstructured data

- Which plays of Shakespeare contain the words Brutus AND Caesar but NOT Calpurnia ?
- One could grep all of Shakespeare's plays for Brutus and Caesar, then remove out lines containing Calpurnia ?
  - Slow (for large corpora)
  - flexible matching operations
  - allow ranked retrieval

- Antony and Cleopatra, Act III, Scene ii

Agrippa [Aside to DOMITIUS ENOBARBUS]: Why, Enobarbus,
When Antony found Julius ***Caesar*** dead,
He cried almost to roaring; and he wept
When at Philippi he found ***Brutus*** slain.

- Hamlet, Act III, Scene ii

Lord Polonius: I did enact Julius ***Caesar*** I was killed i' the
Capitol; ***Brutus*** killed me.

Term-document incidence matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Antony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

Matrix element (t,d) is 1 if the play in column d contains the word in row t , and is 0 otherwise

Incidence vectors

- So we have a 0/1 vector for each term.
- To answer query: take the vectors for ***Brutus, Caesar*** and ***Calpurnia*** (complemented) ➔ bitwise *AND*.
  - 110100 *AND*
  - 110111 *AND*
  - 101111 =
  - **100100**

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

tf-idf example: lnc.ltn

Query: “best car insurance”. Document: “car insurance auto insurance”.

word	query					document				product
	tf-raw	tf-wght	df	idf	tf-idf weight	tf-raw	tf-wght	tf-wght	n'lized	
auto	0	0	5000	2.3	0	1	1	1	0.52	0
best	1	1	50000	1.3	1.3	0	0	0	0	0
car	1	1	10000	2.0	2.0	1	1	1	0.52	1.04
insurance	1	1	1000	3.0	3.0	2	1.3	1.3	0.68	2.04

term frequency, df: document frequency, idf: inverse document frequency, weight:the final weight of the term in the query or document, n’lized: document weights after cosine normalization, product: the product of final query weight and final document weight

1/1.92 0.52

1.3/1.92 0.68 Final similarity score between query and

document: $\sqrt{1^2 + 0^2 + 1^2 + 1.3^2} \approx 1.92$ + 2.04 = 3.08

$\sum$

Steps - Cosine Similarity

- Step 1 : Term frequency (TF)
- Step 2 : Inverse Document Frequency(IDF)
- Step 3 : TF * IDF
- Step 4 : Vector Space Model Cosine Similarity

Example

- Document 1: The game of life is a game of everlasting learning
- Document 2: The unexamined life is not worth living
- Document 3: Never stop learning

Step 1 : Term frequency (TF)

- The term frequency $\mathrm{tf}_{t,d}$ of term t in document d is defined as the number of times that t occurs in d.

Step 1 : Term frequency (TF)

Document1	the	game	of	life	is	a	everlasting	learning
Term Frequency	1	2	2	1	1	1	1	1

Document2	the	unexamined	life	is	not	worth	living
Term Frequency	1	1	1	1	1	1	1

Document3	never	stop	learning
Term Frequency	1	1	1

- Document 1: The game of life is a game of everlasting learning
- Document 2: The unexamined life is not worth living
- Document 3: Never stop learning

Term frequency tf

- In reality each document will be of different size.
- On a large document the frequency of the terms will be much higher than the smaller ones.
- Hence we need to normalize the document based on its size.
- One method - divide the term frequency by the total number of terms.
- For example in Document 1 the term game occurs two times.
- The total number of terms in the document is 10.
- Hence the normalized term frequency is 2 / 10 = 0.2.

Normalized TF

Document1	the	game	of	life	is	a	everlasting	learning
Normalized TF	0.1	0.2	0.2	0.1	0.1	0.1	0.1	0.1

Document2	the	unexamined	life	is	not	worth	living
Normalized TF	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857

Document3	never	stop	learning
Normalized TF	0.333333	0.333333	0.333333

Step2:Inverse Document Frequency(IDF)

- The main purpose of doing a search is to find out **relevant documents** matching the query.
- In the first step all terms are considered equally important.
- Certain terms that occur too frequently have little power in determining the relevance.
- We need a way to **weigh down** the effects of too frequently occurring terms.
- Also the terms that occur less in the document can be more relevant.
- We need a way to **weigh up** the effects of less frequently occurring terms.
- Logarithms helps to solve this problem.

Logarithms -

helps to shrink the numbers of very high magnitude to a smaller one which our brains can deal with easily.

- $\log_{10}(100)$ is 2 because $10^2 = 100$
- $\log_{10}(1000)$ is 3 because $10^3 = 1000$
- $\log_{10}(10000)$ is 4 because $10^4 = 10000$

- 1 in 5,300 dies each year due to car crash.
- 1 in 800 dies each year due to diseases caused by smoking.
- 1 in 2,000,000 is killed by lightning.

idf - Inverse Document Frequency

- "inverse document frequency"
- measures how common a word is among all documents in bloblist.
- More common a word is, the lower its idf.
- We take the ratio of the total number of documents to the number of documents containing word, then take the log of that.
- Add 1 to the divisor to prevent division by zero.

- IDF(**game**) = 1 + $\log_e$(Total Number Of Documents / Number Of Documents with term **game** in it)
- There are 3 documents in all
    - Document1, Document2, Document3
- The term game appears in Document1
- IDF(**game**) = 1 + $\log_e(3 / 1)$ = 1 + 1.098726209 = 2.098726209

Terms	IDF
the	1.405507153
game	2.098726209
of	2.098726209
life	1.405507153
is	1.405507153
a	2.098726209
everlasting	2.098726209
learning	1.405507153
unexamined	2.098726209
not	2.098726209
worth	2.098726209
living	2.098726209
never	2.098726209
stop	2.098726209

Step 3 : TF * IDF

- to find out relevant documents for the query: life learning
- For each term in the query multiply its normalized term frequency with its IDF on each document.
- In Document1 for the term life the normalized term frequency is 0.1 and its IDF is 1.405507153.
- Multiplying them together we get 0.140550715 (0.1 * 1.405507153).

Step 3 : TF * IDF

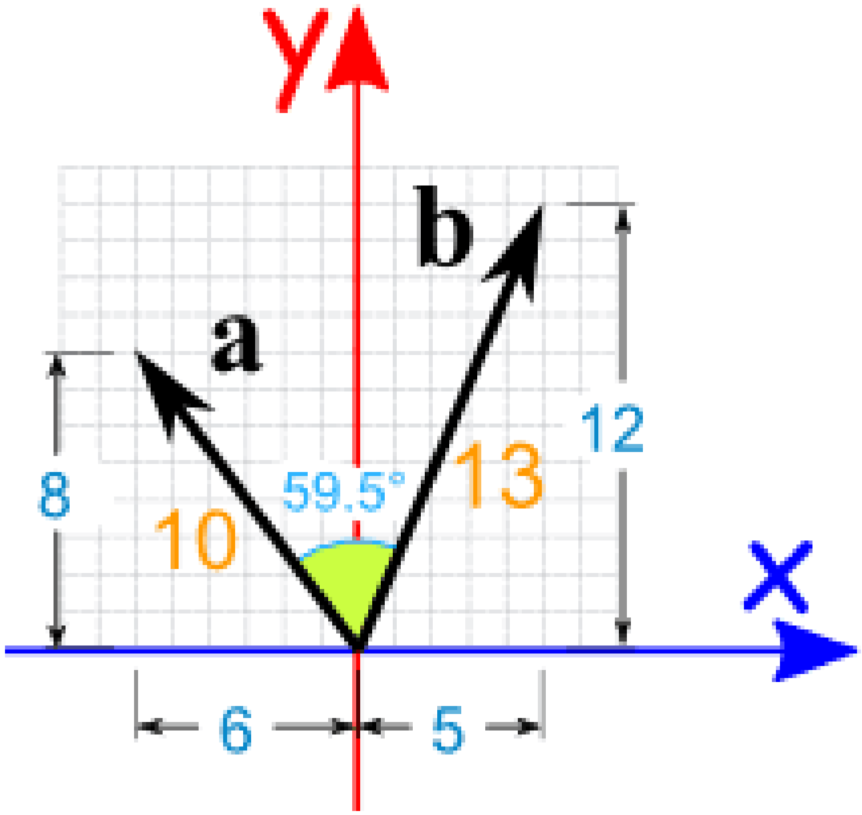
	Document1	Document2	Document3
life	0.140550715	0.200786736	0
learning	0.140550715	0	0.468502384

Step 4:Vector Space Model

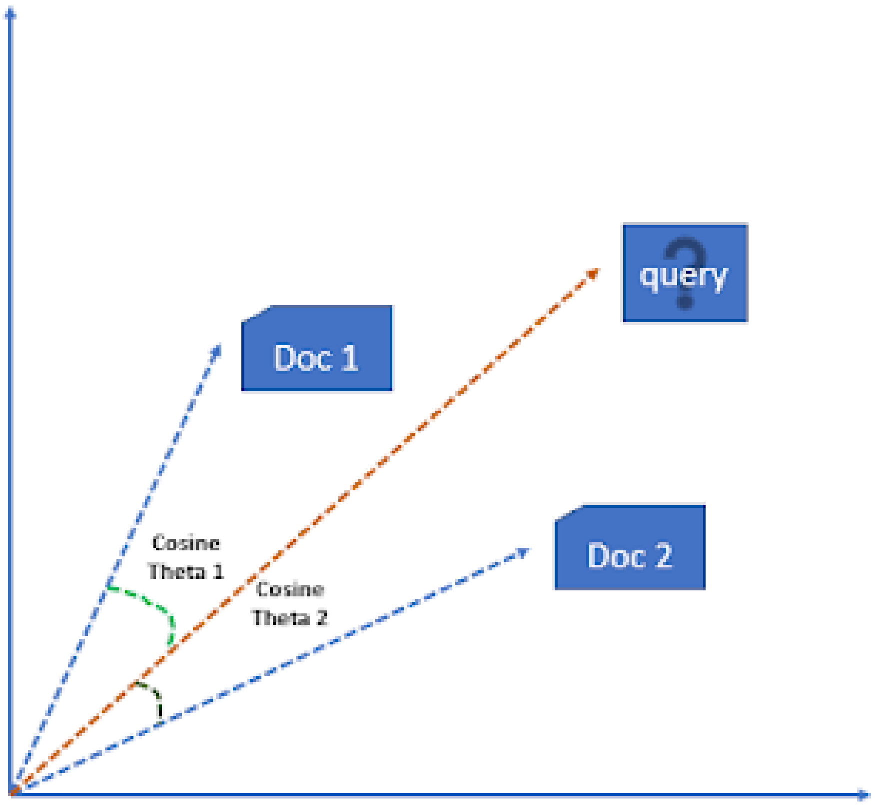
- The representation of a set of documents as vectors in a common vector space is known as the *vector space model*
- It is fundamental to a host of information retrieval operations ranging from
  - scoring documents on a query,
  - document classification and
  - document clustering.

Step 4:Vector Space Model Cosine Similarity

- From each document we derive a vector.
- The set of documents in a collection then is viewed as a set of vectors in a vector space.
- Each term will have its own axis.



y
b
a
12
8
59.5°
13
10
x
6
5



query
Doc 1
Cosine
Theta 1
Cosine
Theta 2
Doc 2

Overview

The Vector Space Model (VSM) is a way of representing documents through the words that they contain

It is a standard technique in Information Retrieval

The VSM allows decisions to be made about which documents are similar to each other and to keyword queries

How it works: Overview

Each document is broken down into a word frequency table

The tables are called vectors and can be stored as arrays

A vocabulary is built from all the words in all documents in the system

Each document is represented as a vector based against the vocabulary

Example

Document A

“A dog and a cat.”

Document B

“A frog.”

a	dog	and	cat
2	1	1	1

a	frog
1	1

Example, continued

The vocabulary contains all words used

- a, dog, and, cat, frog

The vocabulary needs to be sorted

- a, and, cat, dog, frog

Example, continued

Document A: “A dog and a cat.”

Vector: (2,1,1,1,0)

a	and	cat	dog	frog
2	1	1	1	0

Document B: “A frog.”

Vector: (1,0,0,0,1)

a	and	cat	dog	frog
1	0	0	0	1

Queries

Queries can be represented as vectors in the same way as documents:

Dog = (0,0,0,1,0)

Frog = ( )

Dog and frog = ( )

Similarity measures

There are many different ways to measure how similar two documents are, or how similar a document is to a query

The cosine measure is a very common similarity measure

Using a similarity measure, a set of documents can be compared to a query and the most similar document returned

The cosine measure

For two vectors d and d’ the cosine similarity between d and d’ is given by:

$$\frac{d \times d'}{|d||d'|}$$

Here d X d’ is the vector product of d and d’, calculated by multiplying corresponding frequencies together

The cosine measure calculates the angle between the vectors in a high-dimensional virtual space

Example

Let d = (2,1,1,1,0) and d' = (0,0,0,1,0)

dXd' = 2X0 + 1X0 + 1X0 + 1X1 + 0X0=1

$|d| = \sqrt{(2^2+1^2+1^2+1^2+0^2)} = \sqrt{7}=2.646$

$|d'| = \sqrt{(0^2+0^2+0^2+1^2+0^2)} = \sqrt{1}=1$

Similarity = 1/(1 X 2.646) = 0.378

Let d = (1,0,0,0,1) and d' = (0,0,0,1,0)

Similarity =

Ranking documents

A user enters a query

The query is compared to all documents using a similarity measure

The user is shown the documents in decreasing order of similarity to the query term

Step 4:Vector Space Model Cosine Similarity

- Julie loves me more than Linda loves me
- Jane likes me more than Julie loves me

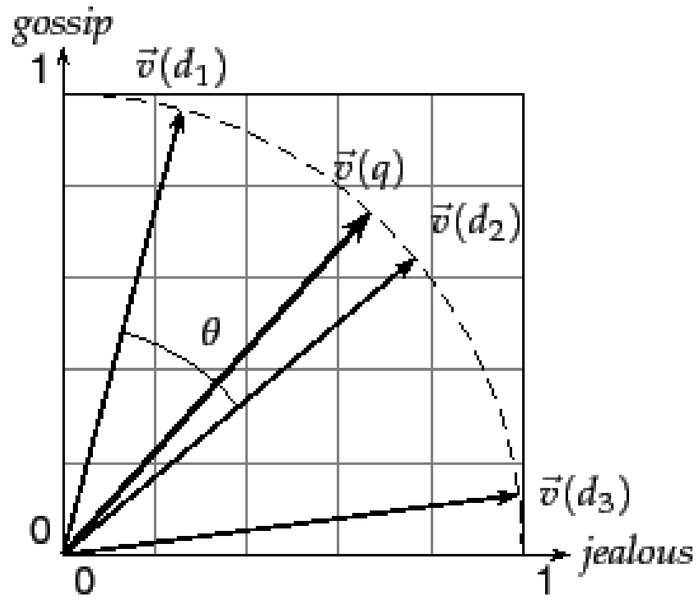
- To know how similar these texts are, purely in terms of word counts
  - (and ignoring word order).
- Begin by making a list of the words from both texts:

- me Julie loves Linda than more likes Jane

Step 4:Vector Space Model Cosine Similarity

- me 2 2
- Jane 0 1
- Julie 1 1
- Linda 1 0
- likes 0 1
- loves 2 1
- more 1 1
- than 1 1

Cosine Similarity



2 vectors are

- a: [2, 0, 1, 1, 0, 2, 1, 1]

- b: [2, 1, 1, 0, 1, 1, 1, 1]

Cosine of the angles between them

- The cosine of the angle between them is about 0.822.
- These vectors are 8-dimensional.
- A virtue of using cosine similarity is clearly that
  - it converts a question that is beyond human ability to visualise to one that can be.
- In this case this is an angle of about 35 degrees which is some 'distance' from zero or perfect agreement.

Cosine Similarity

$$\cos\theta = \frac{\mathbf{d_2} \cdot \mathbf{q}}{\|\mathbf{d_2}\| \, \|\mathbf{q}\|}$$