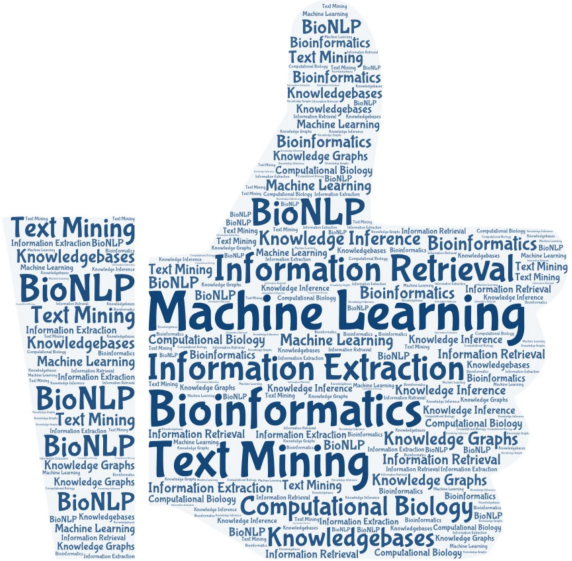
Level 4 / MSci NLP projects

- Projects allow you to go deeper and solve some cool NLP problems!
- Many NLP projects are listed for Level 4 students (including many of my own)
- Can self-propose an NLP project for Level 4 / MSci
- Good idea to think about this before September

Come chat or email me

Example projects



Coreference resolution for biomedical text: When a paper says “The drug was successful”, which drug is it talking about?

Extracting cancer knowledge with NLP: Identifying cancer mutations associated with drug resistance from research articles

Conversational search (with Sean Macavaney): Find documents that can help craft a reply in a conversational system (e.g., Siri, Alexa)

N.B. No biomedical knowledge is required for biomed projects

PhDs at Glasgow

- Join an international community of researchers
- Flexible working with a friendly work environment and fantastic compute resources
- Pursue your curiosity and become a world-expert on an NLP problem
- PhD students are funded with a modest stipend

Next deadline is 31st January

Come chat with me if interested in a PhD

Geometric similarity, Text Distributions, and Clustering

Jake Lever & Sean MacAvaney

University of Glasgow | School of Computing Science

Welcome!

Last time we covered:

- When do computers need to work with text?
- Overview of the course
- Basic text processing approaches (e.g., tokenisation, stemming)
- Set-based document similarity techniques

Today we will talk about:

- How to represent documents as vectors in a term space
- How to measure similarities between document vectors
- Tf-idf term weights
- A common distribution of text: Zipf's Law
- Document clustering techniques

Reminder: Labs

Two sessions:

- Tuesdays 9-12

Labs are not marked, but we **strongly** recommend doing them and attending the lab sessions. Text as Data is a practical course, and the best way to understand the content is by doing it.

Students who skipped the labs (or waited until the end of the term to do all of them) in previous years did worse on exams and coursework than those who did the labs.

Review: Set-based Similarity

Input documents:
X: “I like IRN-BRU”
Y: “Irn bru is bad”

Step 1: Pre-processing: tokenisation, stemming, case folding, sets:
X = {“bru”, “i”, “irn”, “like”}
Y = {“bad”, “bru”, “irn”, “is”}

Step 2: Compute similarity score (e.g., Overlap, Jaccard):
Jaccard:

$$\frac{|X \cap Y|}{|X \cup Y|} = \frac{|\{\text{“irn”, “bru”}\}|}{|\{\text{“bad”, “bru”, “i”, “irn”, “is”, “like”}\}|} = \frac{2}{6} = \frac{1}{3}$$

Think-Pair-Share: Set-based Similarity

$$\frac{|X \cap Y|}{|X \cup Y|} = \frac{|\{\text{“irn”, “bru”}\}|}{|\{\text{“bad”, “bru”, “i”, “irn”, “is”, “like”}\}|} = \frac{2}{6} = \frac{1}{3}$$

In your group, discuss the following and write your notes in the shared Padlet

Let’s say you want to find news articles that are similar to this one:

- What are some limitations of using set-based similarity for this task?

https://padlet.com/jakelever/tad2025_3

Source: https://www.bbc.co.uk/news/world-australia-60014059

Observation: Some words are more important than others

Ideally, I'd want to find articles about Novak Djokovic's Australian visa woes.

Important terms in this document:
- {Novak, Djokovic, Australia, Visa} (arguably others too)

I would want to find other documents where those terms are also important.

A simple heuristic for importance: Tokens that appear a lot in this document, but don't appear a lot in most documents.

BBC Sign in Home News Sport Weather iPlayer Sounds

NEWS

Home | Coronavirus | Climate | UK | World | Business | Politics | Tech | Science | Health | Family & Education

World | Africa | Asia | Australia | Europe | Latin America | Middle East | US & Canada

Novak Djokovic: Tennis star deported after losing Australia visa battle

Novak Djokovic has been deported from Australia after losing a last-ditch court bid to stay in the country.

Judges rejected a challenge by the unvaccinated tennis star after the government cancelled his visa on "health and good order" grounds.

Djokovic said he was "extremely disappointed" but accepted the ruling. He has left on a flight to Dubai.

It marks the end of a 10-day saga, in which the Serb fought to stay to defend his title in the Australian Open.

Djokovic's supporters fell silent outside the courtroom as the decision was announced on the eve of what would have been his opening match in the tournament. One fan told the BBC her summer would be "empty" without the 34-year-old playing at the Open.

Source: https://www.bbc.co.uk/news/world-australia-60014059

From Sets to Sparse Vectors

Treating a Set as a Vector

Step 1: Build a **vocabulary** from all the tokens in the corpus (which is a collection of documents).

Each token is assigned an integer ID.

Token ID	Token
0	a
1	bad
2	bru
3	i
4	irn
5	is
6	like
7	soda
8	taste
9	the

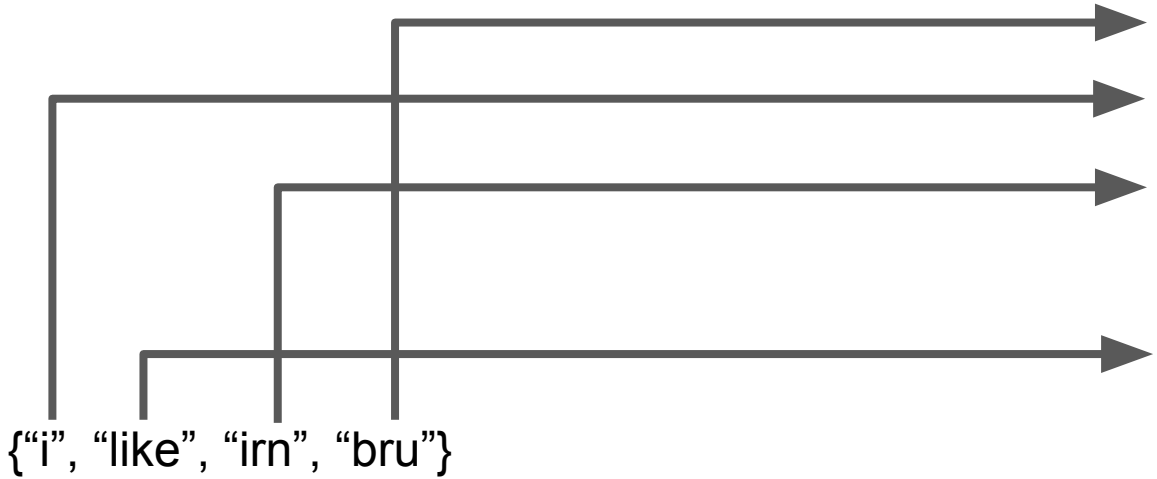
…

Treating a Set as a Vector

Step 2: Build a vector where the indices of the words in the set are all set to 1 (all others 0).

This is called “boolean” or “one-hot” encoding.

The length of the vector is the **vocabulary size**.



Present?	Token ID	Token
0	0	a
0	1	bad
1	**2**	**bru**
1	**3**	**i**
1	**4**	**irn**
0	5	is
1	**6**	**like**
0	7	soda
0	8	taste
0	9	the
	…	

Treating a Set as a Vector

In practice, most of the values in this vector will be 0.

vocabulary size >> document length
(Standard English: ~150K to 200K unique tokens.)

Since most values are 0, we call this data **sparse**.

Let's say we allocate an entire vector for this data (e.g., as an array):

{"i", "like", "irn", "bru"} =
`[0, 0, 1, 1, 1, 0, 1, 0, 0, 0, …]`

Fixed-length
(vocabulary size)

Drawback: Lots of wasted memory storing 0s.

Present?	Token ID	Token
0	0	a
0	1	bad
1	**2**	**bru**
1	**3**	**i**
1	**4**	**irn**
0	5	is
1	**6**	**like**
0	7	soda
0	8	taste
0	9	the

Treating a Set as a Vector

Often better: store as a sparse representation:

Sparse: Pairs of Token ID-value are stored (only non-zeros):

Variable-length (document length)

{“i”, “like”, “irn”, “bru”} = `[ (2, 1), (3, 1), (4, 1), (6, 1) ]`

Token ID

value (always 1, for now)

Sparse *Data* vs Sparse *Representation*

- Your data (e.g., document vector) can be either **sparse** or **dense**.
- You can choose to ***represent*** this data in various ways.

		Data	
		Dense (mostly non-zero)	Sparse (mostly zero)
Representation	Dense (fixed-length array)	✅ good idea	
	Sparse (pairs of IDs/values)		✅ good idea

UNK tokens

With Bag-of-Words, you must define your vocabulary

Problem: what happens when you see a new token?!

Solution: Use UNK (for unknown)

I like **sprite**

Token ID	Token
1	bru
2	i
3	irn
4	like
5	UNK

((2,1), (4,1), (5,1))

Dealing with punctuation

- Should punctuation be included in the vocab for BoW?
  - Depends on your applications (but often no)
  - Perhaps a document that includes “???” indicates some meaning
  - But “.” is likely not informative
- Some punctuation is included inside tokens (e.g. n’t)

Beyond One-Hot Encoding

Bag-of-Words Model - Term Frequency

Assumption: If a term occurs lots in a document it should imply something about what that document is about.

- This is a relaxation of the **binary occurrence assumption**.
- Recording the **term frequency** information provides more information of “aboutness”

BBC Sign in Home News Sport Weather iPlayer Sounds

NEWS

Home | Coronavirus | Climate | UK | World | Business | Politics | Tech | Science | Health | Family & Education

World | Africa | Asia | Australia | Europe | Latin America | Middle East | US & Canada

Novak Djokovic: Tennis star deported after losing Australia visa battle

Novak Djokovic has been deported from Australia after losing a last-ditch court bid to stay in the country.

Judges rejected a challenge by the unvaccinated tennis star after the government cancelled his visa on "health and good order" grounds.

Djokovic said he was "extremely disappointed" but accepted the ruling. He has left on a flight to Dubai.

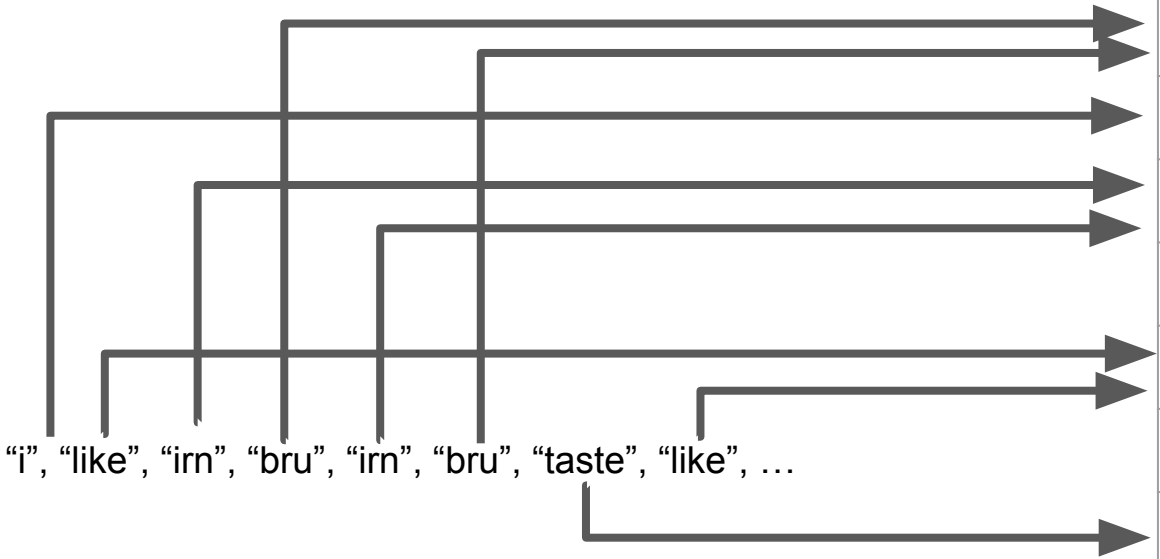
It marks the end of a 10-day saga, in which the Serb fought to stay to defend his title in the Australian Open.

Djokovic's supporters fell silent outside the courtroom as the decision was announced on the eve of what would have been his opening match in the tournament. One fan told the BBC her summer would be "empty" without the 34-year-old playing at the Open.

Source: https://www.bbc.co.uk/news/world-australia-60014059

Bag-of-Words Model - Term Frequency

Count the number of occurrences that a term appears in a document (Term Frequency, TF)



Count (TF)	Token ID	Token
0	0	a
0	1	bad
2	**2**	**bru**
1	**3**	**i**
2	**4**	**irn**
0	5	is
2	**6**	**like**
0	7	soda
1	**8**	**taste**
0	9	the
		…

Bag-of-Words Model - Term Frequency

In memory, this can be:

Dense Rep.: `[0, 0, 2, 1, 2, 0, 2, 0, 1, 0, …]`

Sparse Rep.: `[ (2, 2), (3, 1), (4, 2), (6, 2), (8, 1) ]`

Token ID count

Think-Pair-Share: Bag-of-Words Representation

Compute the **Bag-of-Words representation of the document text** below using the vocabulary on right (assume tokenizing on whitespace, lowercasing, porter stemming and ignoring punctuation):

"Fluffy cashmere blankets are very fluffy!"

Token	ID
a	0
along	1
and	2
are	3
blanket	4
buy	5
cashmer	6
come	7
edinburgh	8
fluffy	9
gener	10
herself	11
hi	12
hous	13
lot	14
me	15

Token	ID
mom	16
mum	17
my	18
of	19
other	20
scarf	21
scarv	22
scottish	23
sell	24
so	25
stuff	26
tartan	27
to	28
want	29
with	30
UNK	31

Think-Pair-Share: Bag-of-Words Representation

"Fluffy cashmere blankets are very fluffy!"

- Raw sequence: [9, 6, 4, 3, 31, 9]
- Dense BoW: [0,0,0,1,1,0,1,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1]
- Sparse BoW: ((3, 1), (4, 1), (6, 1), (9, 2), (31,1))

Token	ID
a	0
along	1
and	2
are	3
blanket	4
buy	5
cashmer	6
come	7
edinburgh	8
fluffy	9
gener	10
herself	11
hi	12
hous	13
lot	14
me	15

Token	ID
mom	16
mum	17
my	18
of	19
other	20
scarf	21
scarv	22
scottish	23
sell	24
so	25
stuff	26
tartan	27
to	28
want	29
with	30
UNK	31

Problem with Term Frequency

Even though important tokens are usually common in a document, **common words are not necessarily important**.

Token	Frequency
the	14
to	4
in	4
djokovic	4
a	4
on	3
his	3
after	3
would	2
was	2
visa	2
tennis	2
start	2
open	2
of	2
novak	2
he	2
has	2
day	2

BBC Sign in Home News Sport Weather iPlayer Sounds

NEWS

Home | Coronavirus | Climate | UK | World | Business | Politics | Tech | Science | Health | Family & Education

World | Africa | Asia | Australia | Europe | Latin America | Middle East | US & Canada

Novak Djokovic: Tennis star deported after losing Australia visa battle

Novak Djokovic has been deported from Australia after losing a last-ditch court bid to stay in the country.

Judges rejected a challenge by the unvaccinated tennis star after the government cancelled his visa on "health and good order" grounds.

Djokovic said he was "extremely disappointed" but accepted the ruling. He has left on a flight to Dubai.

It marks the end of a 10-day saga, in which the Serb fought to stay to defend his title in the Australian Open.

Djokovic's supporters fell silent outside the courtroom as the decision was announced on the eve of what would have been his opening match in the tournament. One fan told the BBC her summer would be "empty" without the 34-year-old playing at the Open.

Source: https://www.bbc.co.uk/news/world-australia-60014059

Problem with Term Frequency

Articles/pronouns (e.g., I, a, the, it, you, your)

Essentially all documents in English will use such words

Remember: We can define a list of **stopwords** that should be removed

Words that are frequent but do not convey much information. E.g.:

"BBC" in a corpus of news articles

"Research" in a corpus of scientific papers

(more on the *distribution* of words later in this lecture)

Co-relations between words. E.g.:

novak and djokovic occur much more frequently together than separately

Magnitudes of term frequencies. E.g.,:

Is the article about "djokovic" (4) twice as much as "visa" (2)?

Token	Frequency
the	14
to	4
in	4
djokovic	4
a	4
on	3
his	3
after	3
would	2
was	2
visa	2
tennis	2
start	2
open	2
of	2
novak	2
he	2
has	2
day	2

Problem 1: Raw Term frequency (tf)

- The term frequency ($tf_{t,d}$) of term *t* in document *d* is defined as the number of times that *t* occurs in *d*.

- **Raw tf has a problem:**
  - It implies that a document with 10 occurrences of a term is 10 times “more about” the term than a document with 1 occurrence

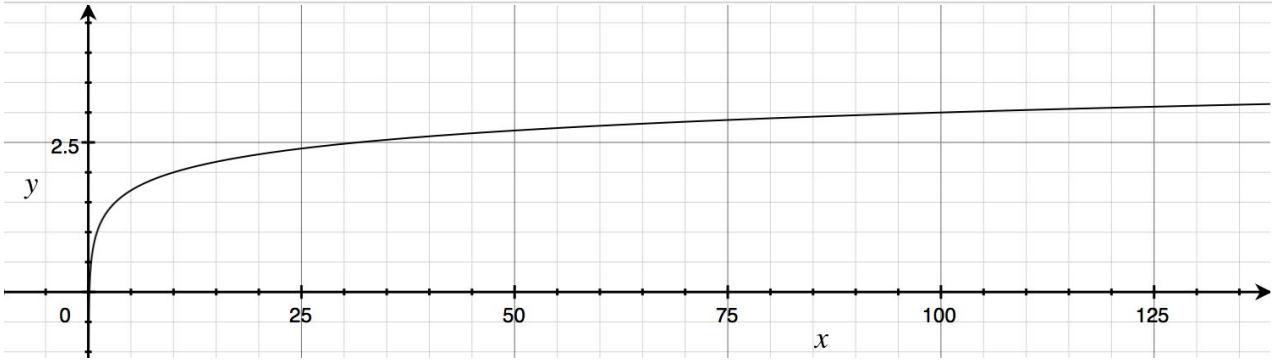
- **Aboutness does not increase linearly with term frequency.**

Solution: Sublinear TF scaling

- The **log** frequency weight of term *t* in *d* is

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

- What impact does this have on very large term frequencies?



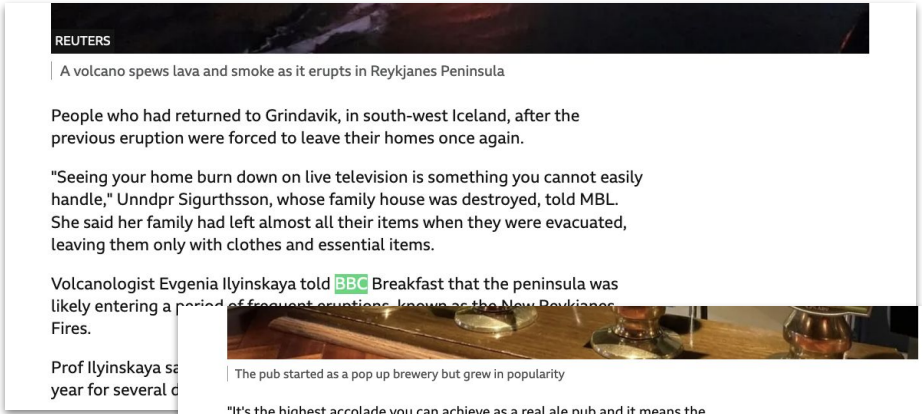
- Example: 0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 1000 → 4

Problem 2 – How ‘useful’ is each term?

Words that occur in many documents do not provide much new “information”

“BBC” in a corpus of news articles

“Research” in a corpus of scientific papers



REUTERS

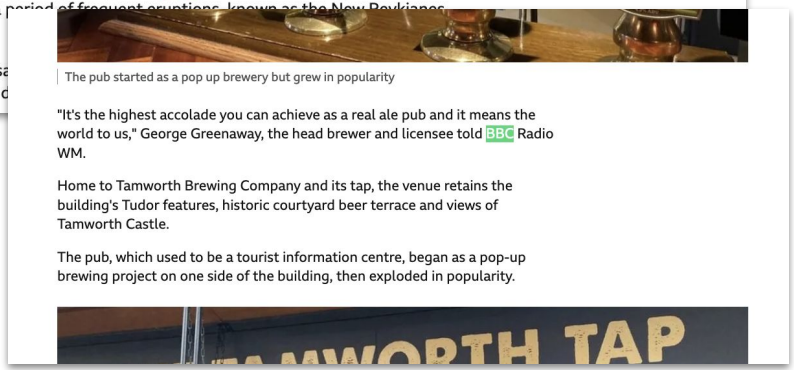
A volcano spews lava and smoke as it erupts in Reykjanes Peninsula

People who had returned to Grindavik, in south-west Iceland, after the previous eruption were forced to leave their homes once again.

"Seeing your home burn down on live television is something you cannot easily handle," Unndpr Sigurthsson, whose family house was destroyed, told MBL. She said her family had left almost all their items when they were evacuated, leaving them only with clothes and essential items.

Volcanologist Evgenia Ilyinskaya told BBC Breakfast that the peninsula was likely entering a period of frequent eruptions, known as the New Reykjanes Fires.

Prof Ilyinskaya sa
year for several d



The pub started as a pop up brewery but grew in popularity

"It's the highest accolade you can achieve as a real ale pub and it means the world to us," George Greenaway, the head brewer and licensee told BBC Radio WM.

Home to Tamworth Brewing Company and its tap, the venue retains the building's Tudor features, historic courtyard beer terrace and views of Tamworth Castle.

The pub, which used to be a tourist information centre, began as a pop-up brewing project on one side of the building, then exploded in popularity.

particular Aberdeenshire, the Highlands, Orkney and Shetland Islands.

Northern Ireland will also be hit, including counties Tyrone and Londonderry.

- **How to drive in snow and icy weather**
- **The best way to de-ice a car and other winter tips**

Scotland's snow and ice warning will extend on Tuesday to northern parts of England, the West Midlands, and Wales - including Durham, Northumberland, Cheshire, Merseyside, Wrexham, and Staffordshire.

These areas may see travel delays, power cuts and potential injuries from slipping and falling on icy roads and pavements.

BBC Weath
coldest spe
significant

"Bitterly co
feeding ple
Northerly w

- **Billions on offer for Stormont, but what for?**

Last week, Prime Minister Rishi Sunak said the public sector pay disputes in Northern Ireland **could be resolved "rapidly"** if the Stormont executive was restored.

He said "significant progress" had been made in talks with the DUP and there was "now a very good basis" to revive the power-sharing institutions.

Speaking ahead of Monday's talks on BBC **Radio Foyle's North West Today**, SDLP leader Colum Eastwood said everyone was "now waiting on Jeffrey [Donaldson, DUP leader]".

He said people across Northern Ireland are becoming increasingly frustrated by the impasse.

"We have had two years now of no government, before that we had three and a half years of no government," he said.

Problem 2 – How ‘useful’ is each term?

Idea: Measure how many documents a term appears in to predict how useful it is.

Common terms (the, of, etc.) are not very helpful

Also: Rare terms (misspellings, etc.) are sometimes also not informative and sometimes removed entirely (pruned).

This is called Document Frequency:

How many documents in the collection contains a term?

Problem 2 – How ‘useful’ is each term?

But wait! Terms that occur in more documents are actually **less** useful than rarer ones.

A higher Document Frequency indicates a term is **less** important

Therefore, we want the inverse of Document Frequency:

A higher Inverse Document Frequency (IDF) indicates a term is **more** important

$$idf_t = \log(\frac{N}{df_t})$$

Notes:

We usually use log (N/dft) instead of N/dft to non-linearly “dampen” the effect of IDF.

Sometimes we don’t specify a base for the log.

In examples, we tend to use base 10 for convenience, but 2 is more common in practice.

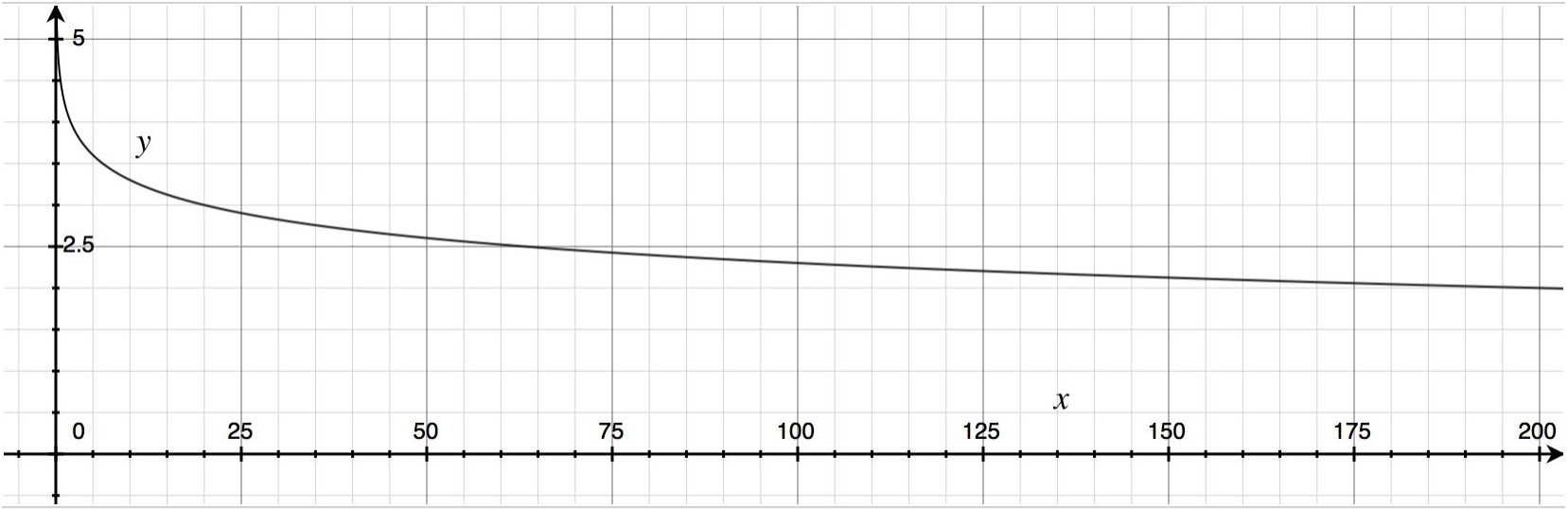
idf example, suppose N = 1 million

term	df_t	idf_t
calpurnia	1	6
animal	100	5
sunday	1,000	4
fly	10,000	3
under	100,000	2
the	1,000,000	1

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

There is one idf value for each term t in a collection.

idf, graphically



Summary on Term Frequency, Document Frequency, Inverse DF

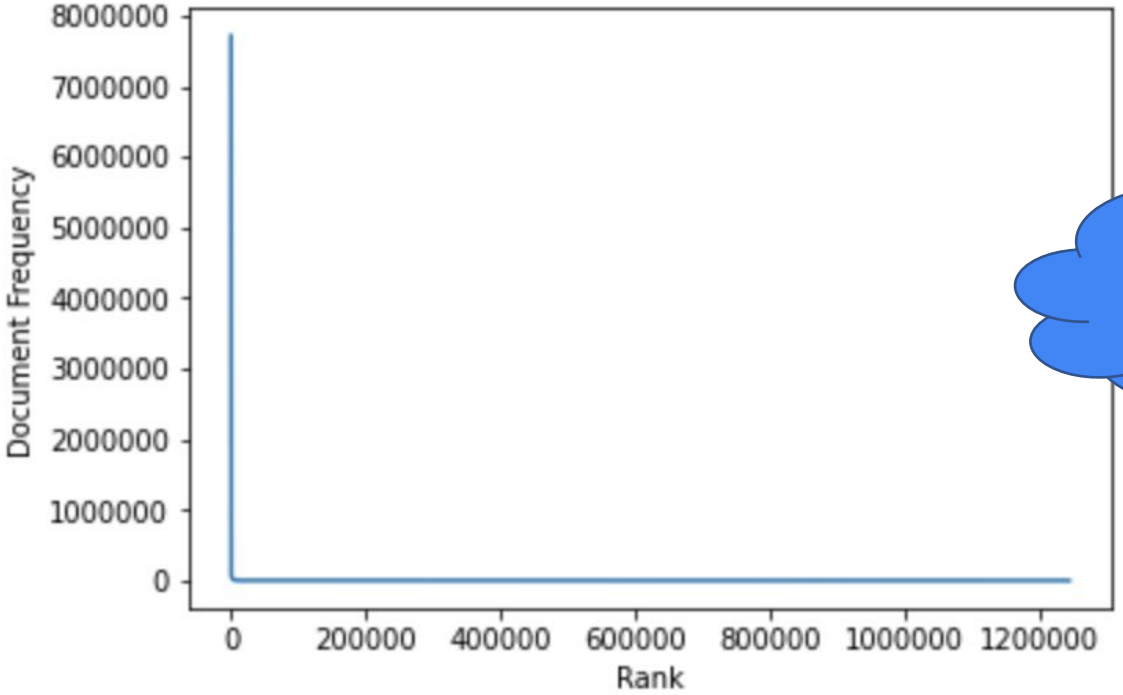
	Description	**High Values**	**Low Values**
Term Frequency (TF)	How many times the term appears in a specific document	A document might be about this term	A document probably isn't about this term
Document Frequency (DF)	How many documents the term appears in an entire corpus	This term is probably uninformative	This term is probably informative
Inverse Document Frequency (IDF)	Inverse of Document Frequency	This term is probably informative	This term is probably uninformative

Let’s rank tokens by document frequency!

Rank	Token	DF
1	the	7,714,694
2	of	6562777
3	and	6432856
4	a	5919793
5	to	5847920
6	in	5321030
…		
1,242,320	tukana	1

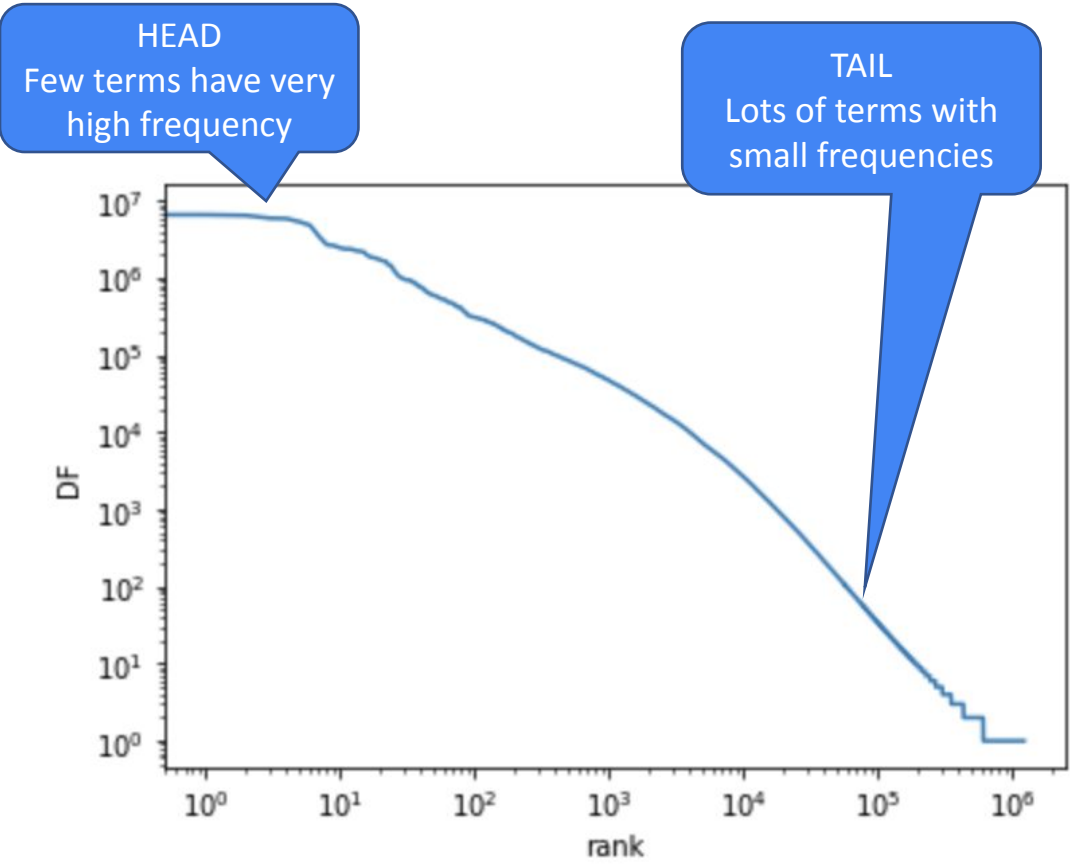
Over MS MARCO, a small corpus with ~9M documents and ~500M tokens

Let’s plot rank versus document frequency



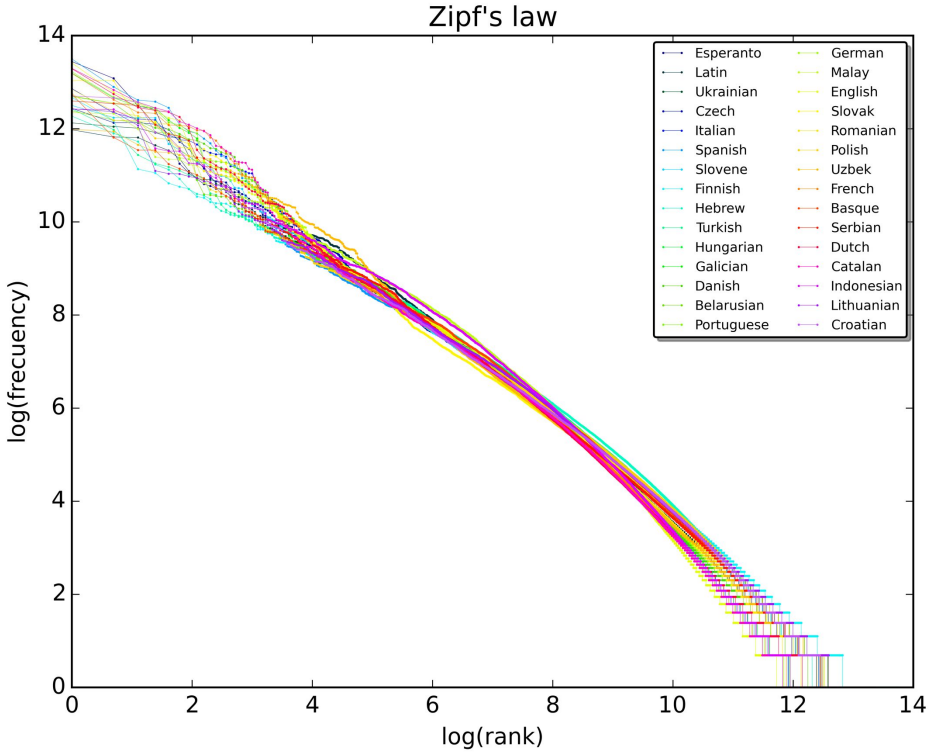
Difficult to interpret like this…

Let's plot, on logarithmic scales



- The most frequent terms are not very useful (but there are only a few of them)
  - E.g. "the", "a"
  - Or "home", "page", "copyright"
- There are lots of infrequent terms
  - But they can be useful!
- This is Zipf's law – it shows us why IDF is useful
  - And why we apply a logarithm to df!

Zipf's law



A statistical property of language.

This can be approximated as a straight-line in log-scale

Zipf's law is an example of a "Power Law"

- Many power law distributions occur in nature

tf-idf weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight.

$$w_{t,d} = (1 + log(tf_{t,d})) * log(\frac{N}{df_t})$$

- A strong weighting scheme for text similarity
  - Note: the "-" in tf-idf is a hyphen, not a minus sign!
  - Alternative names: tf.idf, tf x idf
- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection

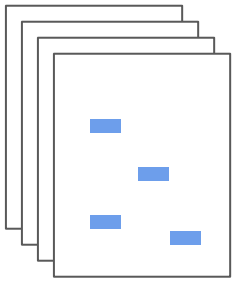
We often use tf.idf weights instead of term frequency counts

tf-idf weighting has many variants

Term frequency		Document frequency	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1
l (logarithm)	$1 + \log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$
a (augmented)	$0.5 + \frac{0.5 \times \mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N - \mathrm{df}_t}{\mathrm{df}_t}\}$
b (boolean)	$\begin{cases} 1 & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$		
L (log ave)	$\frac{1 + \log(\mathrm{tf}_{t,d})}{1 + \log(\mathrm{ave}_{t \in d}(\mathrm{tf}_{t,d}))}$		

More advanced options

Document length: Normalise to deal with longer versus shorter documents (discussed in IR course)



Replace document frequency: Use the times a word appears in many documents (collection frequency) instead of the number of documents

Probabilistic interpretation: Use the likelihood of words appearing (discussed later)

Summary

- How representative a term is of a document is **not linearly** correlated with its term frequency
- We can examine the **collection** as a whole for information about how ***informative*** each term is.
- The key outcome is TF.IDF: **Term Frequency** x **Inverse Document Frequency**.

Break!

I like **sprite**

Token ID	Token
1	bru
2	i
3	irn
4	like
5	UNK

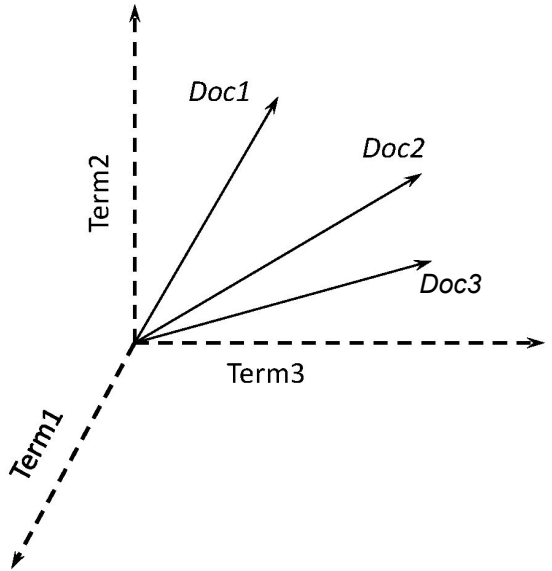
((2,1), (4,1), (5,1))

Up next: treating document vectors as geometric vectors

Geometric Similarity

Vector Space Model

- We can plot the vectors for each document



Assumption: Documents that are "close together" in vector space "talk about" the same things, i.e. are similar

- 3D pictures are useful, but can be misleading for high-dimensional space

Formalizing vector space proximity

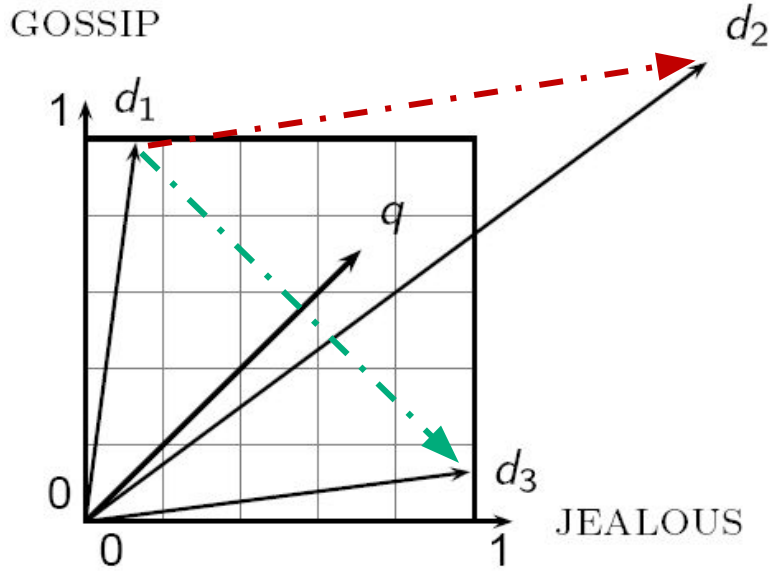
Idea: measure the distance between the endpoints of the vectors.

- This is called “Euclidean distance”

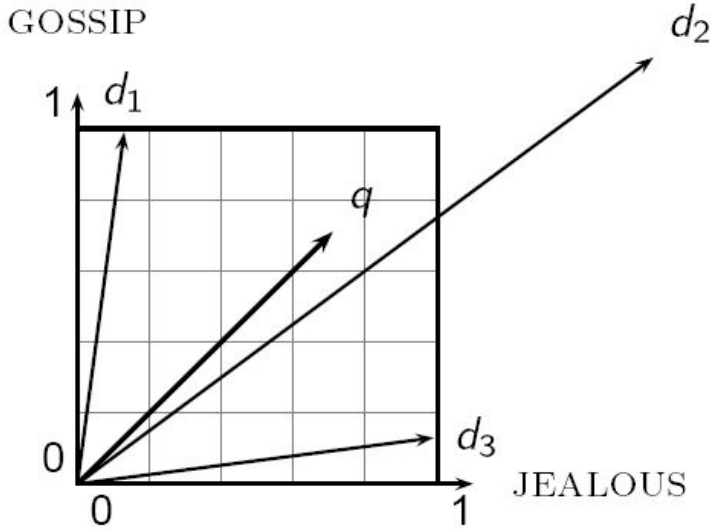
$$||\overrightarrow{D_1} - \overrightarrow{D_2}||$$

Problem: the distance is large for vectors of different lengths.

The Euclidean distance between D_1 and D_2 is larger than the distance between D_1 and D_3, even though the distribution of terms in D_1 is more similar to D_3 than to D_2.

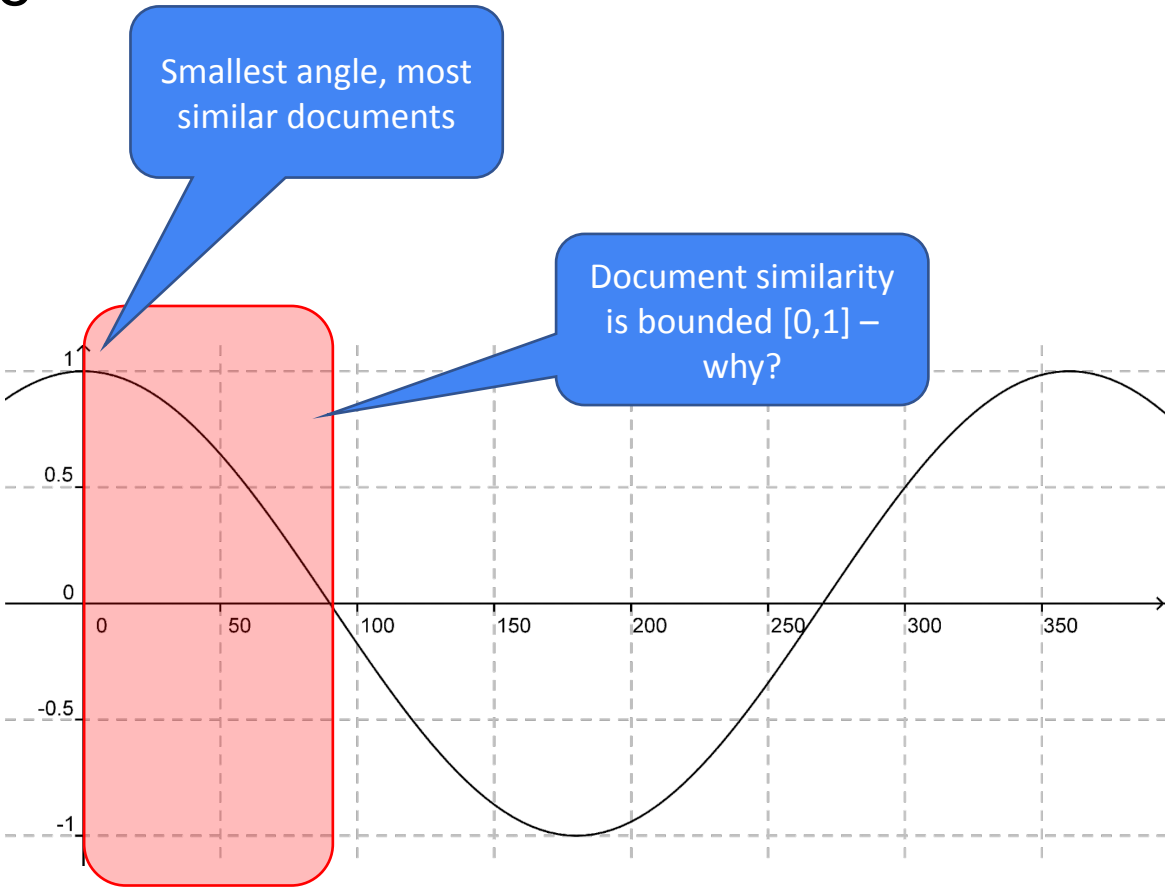


Idea: Use the angle between vectors



- Use the angle (instead of the distance) between two vectors
- Benefit:
  - Vector magnitude is ignored so long/short documents can be similar
- But:
  - We want high score for similar documents
  - Identical documents would have angle=0
  - We need a transform!

Cosine Curve



Vector Space Similarity

- Similarity is the distance between points representing units of text
  - *Similarity* measure more common than a distance or *dissimilarity* measure
  - e.g. Cosine correlation

Recall from geometry:

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta$$

$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \cos\theta$$

$$|\vec{v}| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

$$Cosine(D_1, D_2) = \frac{D_1 \cdot D_2}{|D_1||D_2|}$$

Dot product of the two vectors

Product of magnitudes

Dot product reminder

- We can do the dot product of two term vectors for D1, D2.
- The dot product of two vectors is a scalar.

$$dot - product(D_1, D_2) = \overrightarrow{D_1} * \overrightarrow{D_2} = \sum_{i=1}^{|V|} v_i w_i = v_1 * w_1 + v_2 * w_2 + \ ... + v_{|V|} * w_{|V|}$$

- v_i is the count for word i in D1
- w_i is the count for word i in D2

In python: Numpy dot-product example:

D1 = np.array([1,2,3])
D2 = np.array([2,3,4])
np.dot(D1, D2) = 20

Exercise: Cosine Similarity

- Consider two documents D_1 and D_2 represented with a bag-of-words representation
  - D_1 = (1,0,3) and D_2 = (4,2,1)
- Task: Calculate their cosine similarity

$$Cosine(D_1, D_2) = \frac{D_1 \cdot D_2}{|D_1||D_2|}$$

Solution: Cosine Similarity

- Consider two documents D_1 and D_2 represented with a bag-of-words representation
  - D_1 = (1,0,3) and D_2 = (4,2,1)
- Task: Calculate their cosine similarity

$$Cosine(D_1, D_2) = \frac{D_1 \cdot D_2}{|D_1||D_2|}$$

Solution:

$$Cosine(D_1, D_2) = \frac{(1x4 + 0x2 + 3x1)}{\sqrt{(1^2 + 0^2 + 3^2)}\sqrt{(4^2 + 2^2 + 1^2)}}$$

$$= \frac{4 + 3}{\sqrt{10}\sqrt{21}} = \frac{7}{\sqrt{210}}$$

$$\approx \frac{7}{14.5} \approx 0.5$$

Summary

Euclidean distance is often not ideal for measuring the similarity between vectors

We instead often measure the **angle** between vectors

To treat the angle into a similarity score (high value = highly similar), we calculate the **cosine** of the angle

Clustering: Motivation

Objectives

Clustering

"How can I group this set of items by similarity?"

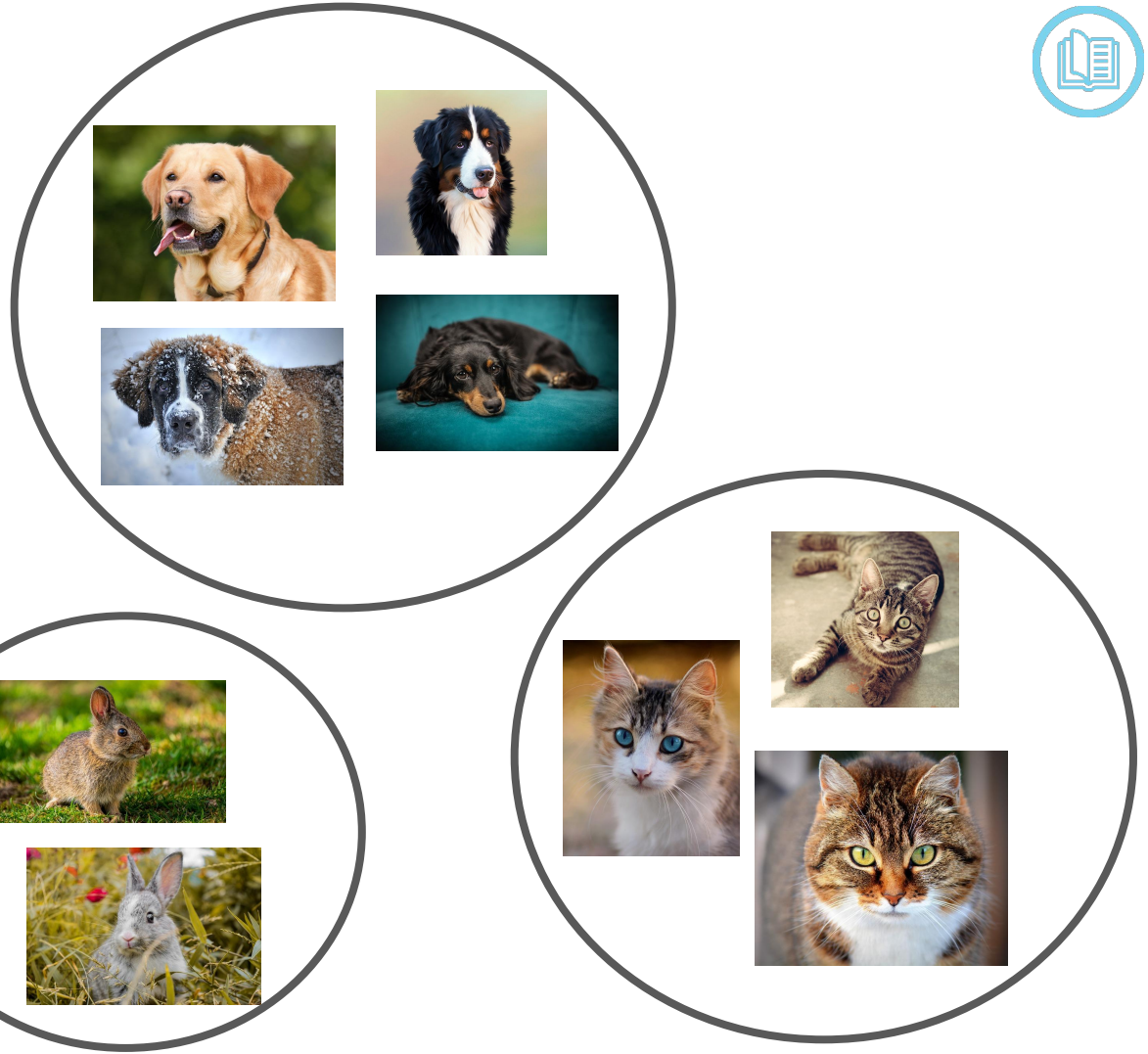
- A classical problem in pattern recognition, statistics, and machine learning

- When applied to documents or objects, they enable a wide range of tasks in text analysis applications

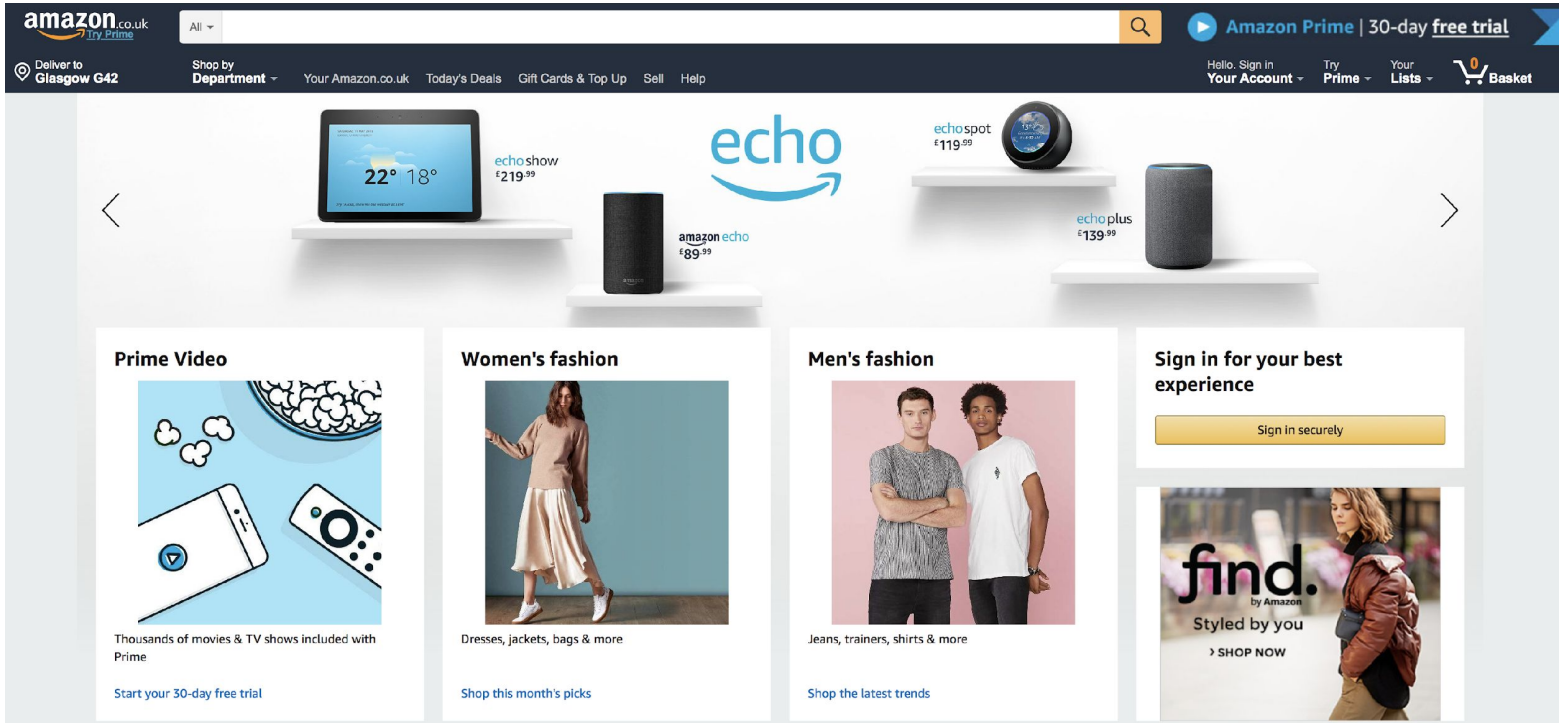
Clustering

“How can I group this set of items by similarity?”

- A classical problem in pattern recognition, statistics, and machine learning
- Enable a wide range of tasks in text analysis applications



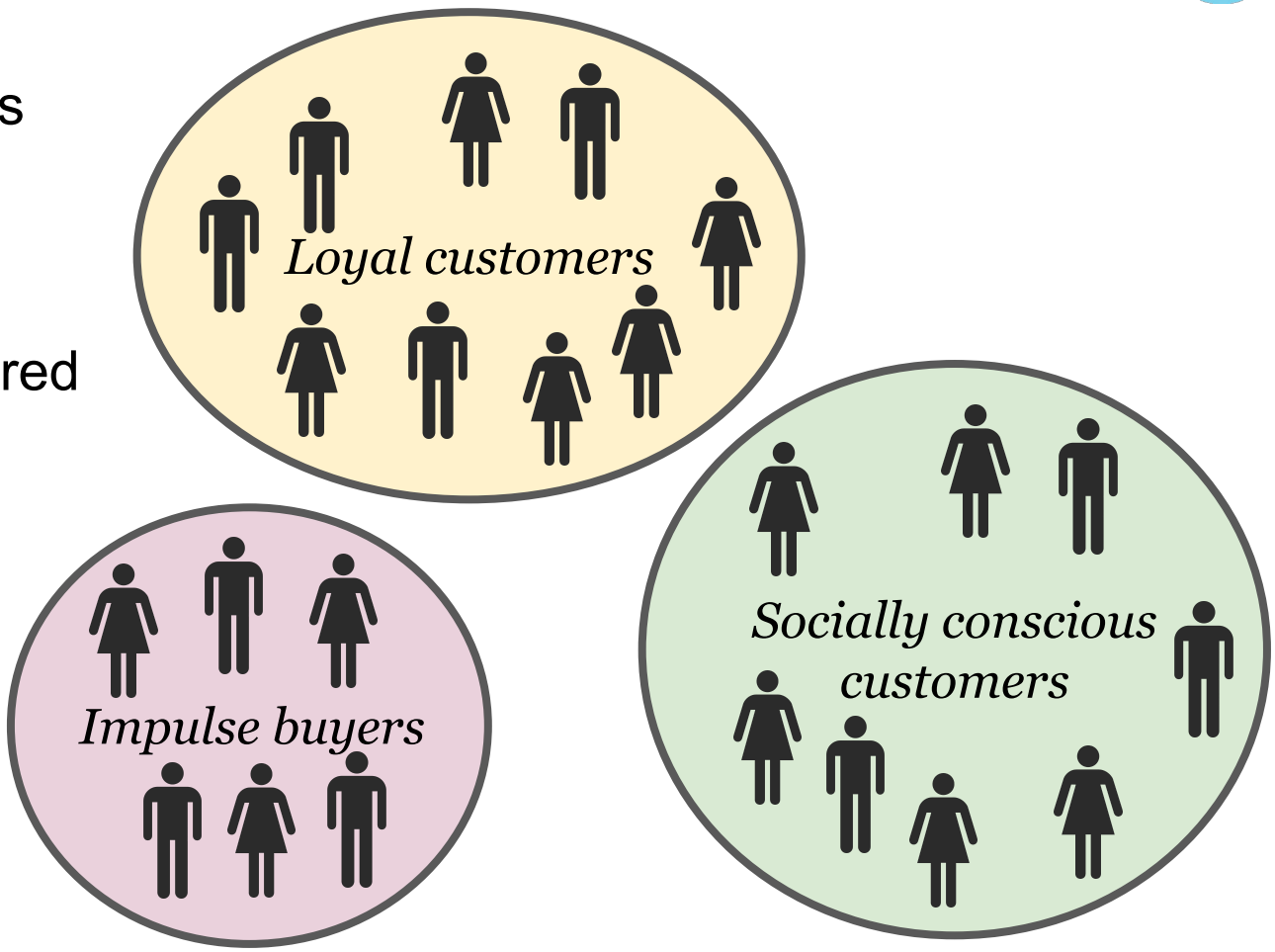
Amazon uses clustering throughout their website



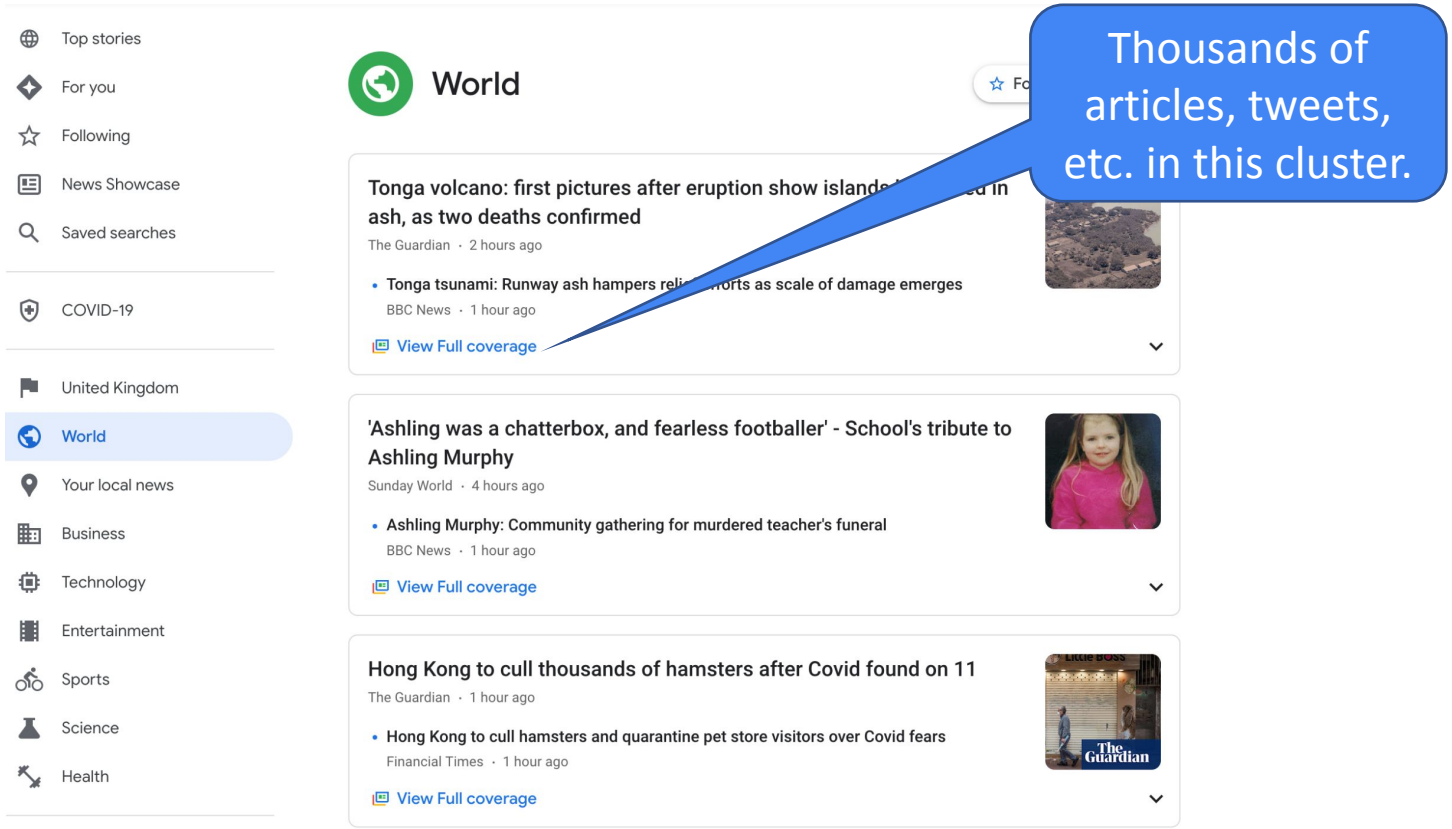
Grouping products based on buying patterns or product features

Application: Customer Segmentation

- Cluster customers by purchasing history
- Target specific clusters with tailored marketing



Application of clustering: Grouping news articles on the same topic



Can we group our Reddit posts using clustering?

Our Reddit posts can be represented a N-dimensional vectors using a one-hot vector or bag-of-words approach. N is the number of tokens in the vocabulary.

Can we use clustering to find clusters of posts? What posts would be in the same cluster as our Irn Bru post?

Anyone tried Irn Bru?

It's a Scottish drink and it's banned some countries and I was wondering if anyone here has tried it. It has quite a unique taste and it's not something you'd forget quickly. You either love it or hate it I think.

29 Comments Award Share Save Hide Report 100% Upvoted

[0, 0, 1, 0, 1, 0, 1, 0 …]

Clustering: Background

Purpose of Clustering

- Discover the underlying structure of data by grouping similar objects together

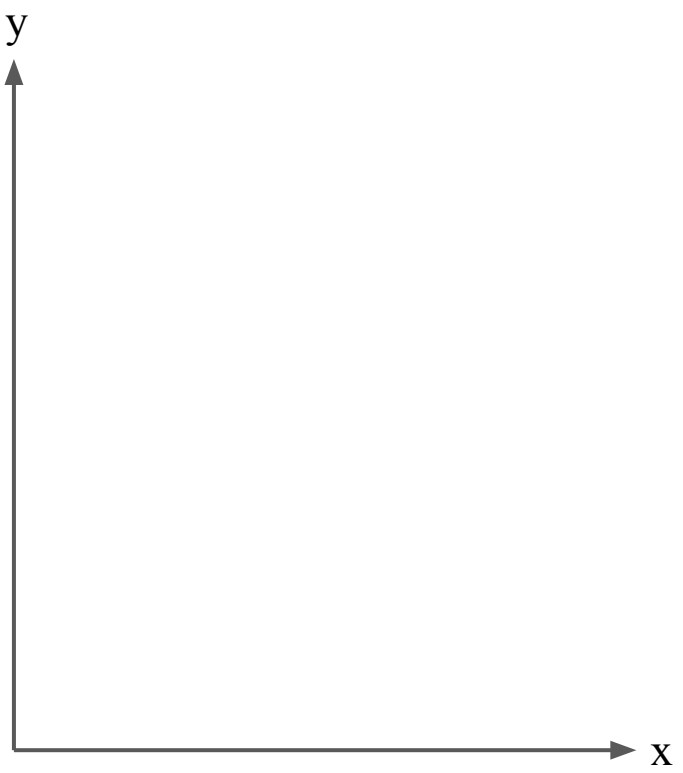
- In TaD, the “objects” are typically text documents
  - Cluster documents by the similarity of the terms they contain
  - Documents within the same cluster are assumed to be similar

- Later: we can also cluster terms
  - E.g. terms that often co-occur in the same documents are grouped together

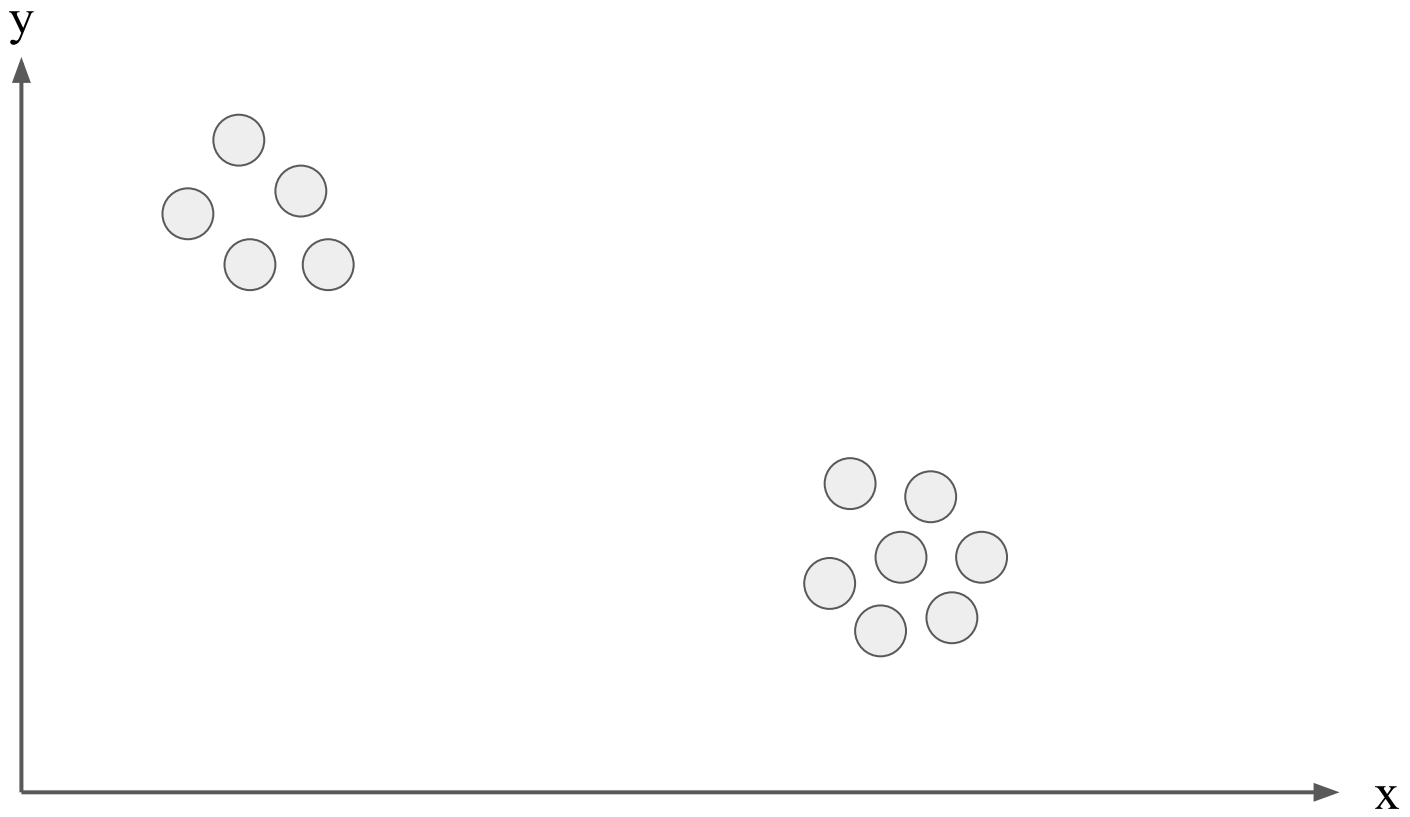
Let’s do some clustering

Documents (like our Reddit posts) are represented by typically VERY high dimensional vectors.

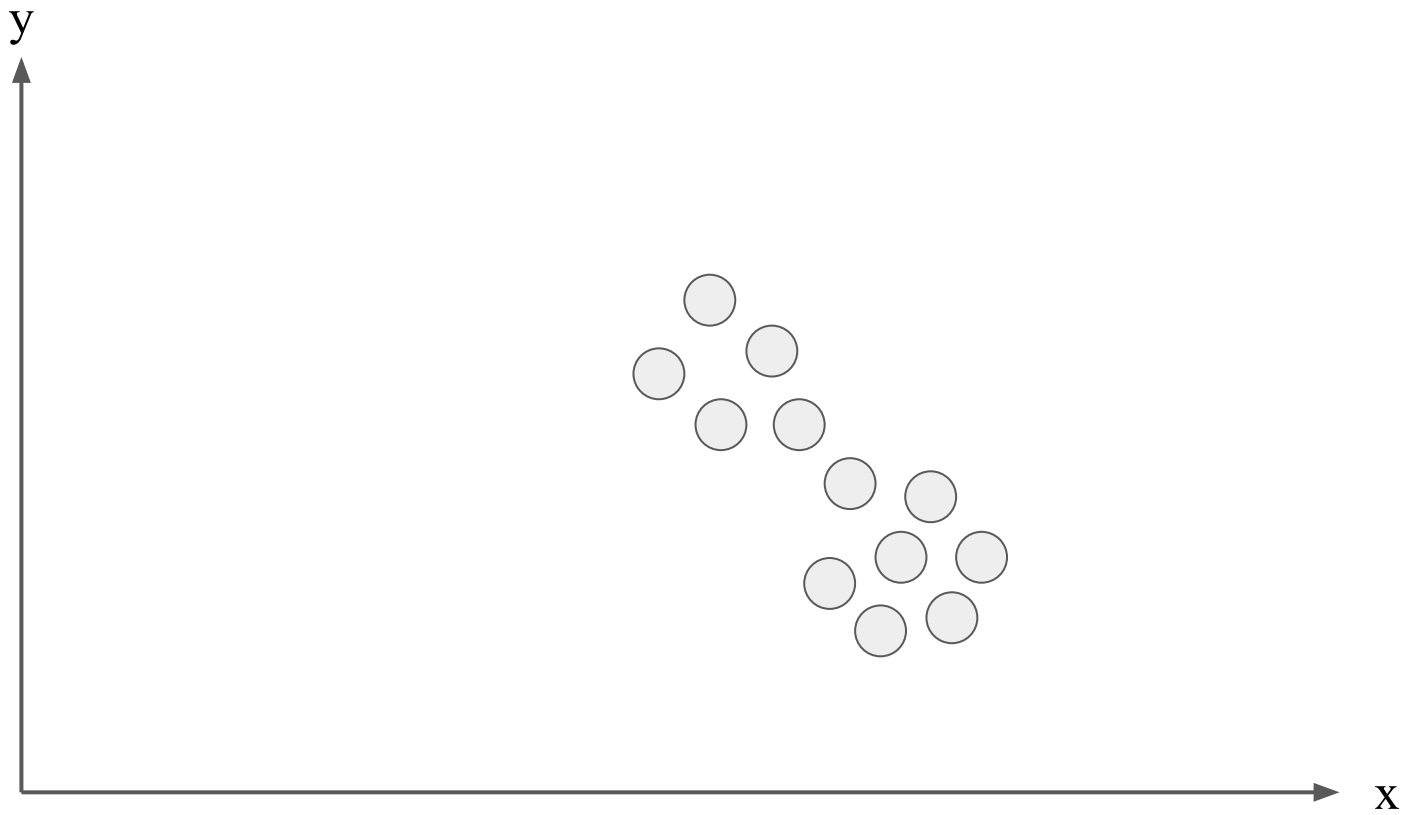
To show clustering, we’re going to work in two dimensions (x and y)



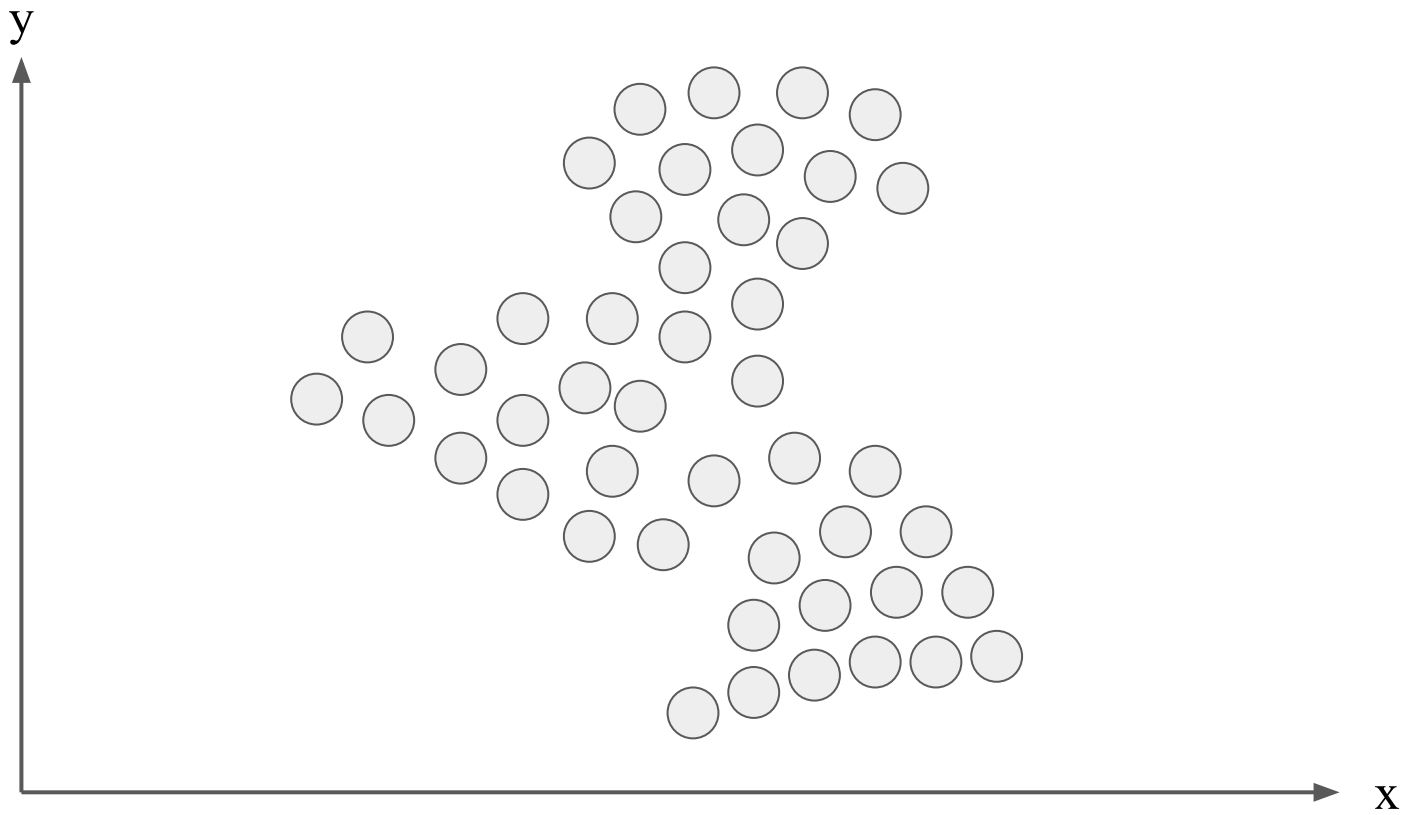
How many clusters?



How many clusters?



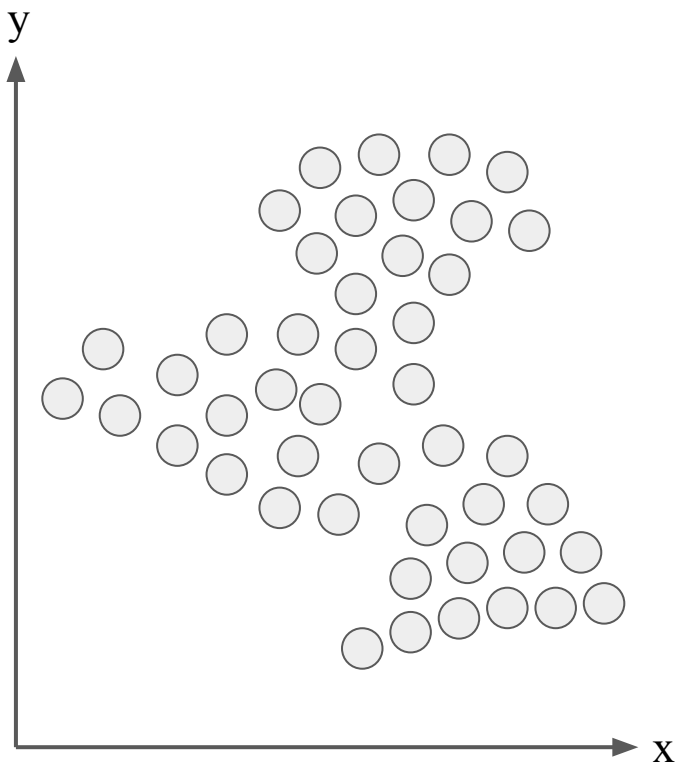
How many clusters?



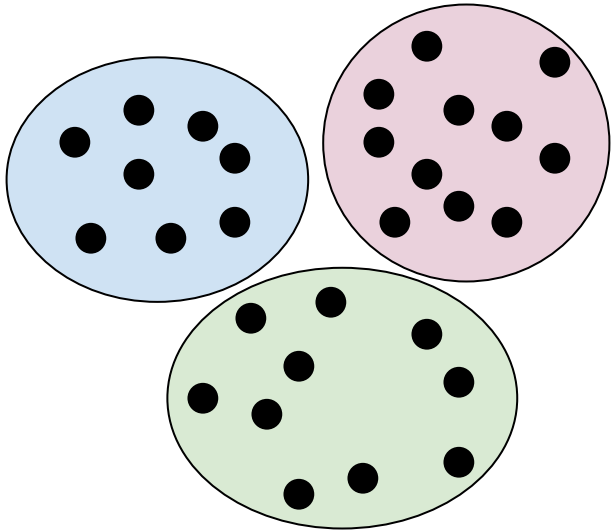
How many clusters should there be?

Often, there is no definitive “right” number of clusters.

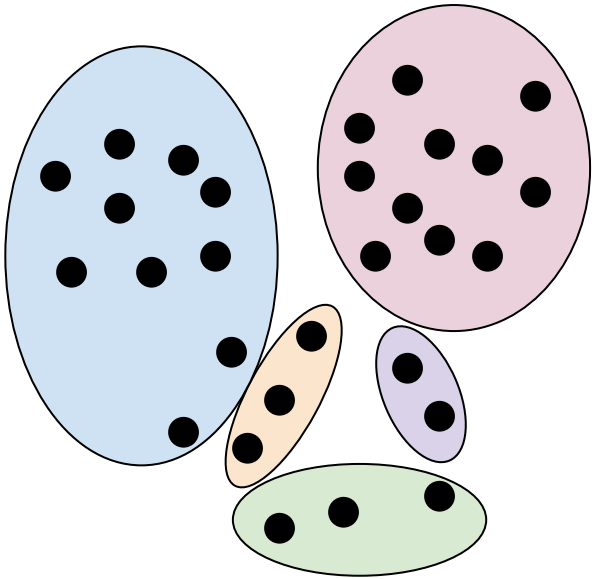
We’ll come back to a method



How many clusters?



Number of clusters and the actual clusters will vary depending on the **clustering algorithm** used



Clustering compared to other ML methods

- Clustering is (typically) an unsupervised learning task.
- Differently from (supervised) classification, the clusters (classes) are not pre-defined
  - No labelled training data
  - Clusters are discovered by the algorithm

The Clustering Process

1. Derive a document representation
   – Typically, vectors of weighted terms

2. Measure similarity between documents

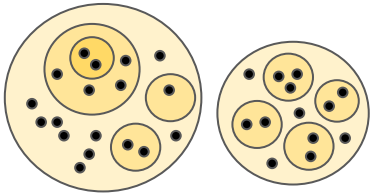
3. Apply a clustering method

4. Check the validity/quality of the clustering

Breather

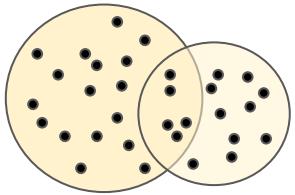
Clustering: Methods

Key Properties of Clustering Algorithms



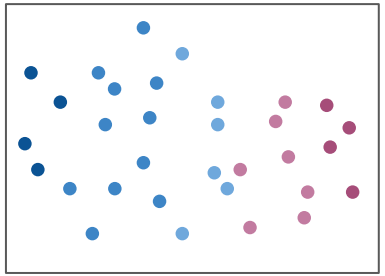
Partitioning criteria

- Single level vs. multi-level hierarchical partitioning
- (often, multi-level hierarchical partitioning is desirable)



Separation of clusters

- Exclusive (e.g., one object belongs to only one cluster) vs. overlapping (e.g., one object may belong to more than one cluster)

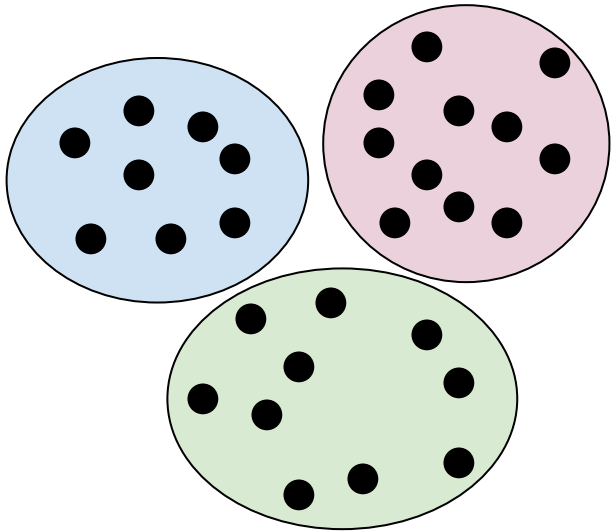


Hard versus fuzzy

- In fuzzy clustering, an object belongs to every cluster with some weight between 0 and 1
- Probabilistic clustering has similar characteristics
- Weights must sum to 1

Partitioning Clustering

- Partitions a set of *N* documents into *K* disjoint clusters
- Find the partition that optimises a specific criterion
- Typically, a function of within-cluster similarity and between-cluster distance



K-Means Clustering

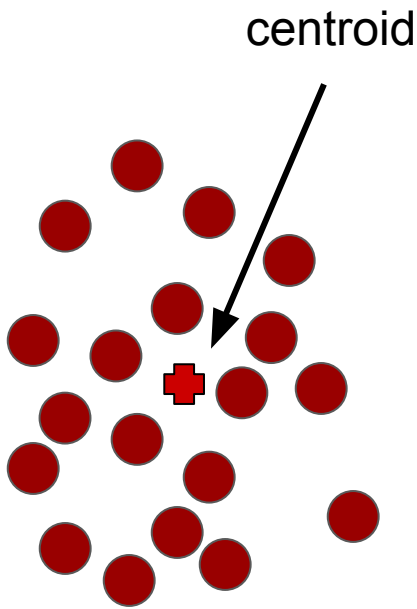
One of the most popular clustering algorithms.

Input:

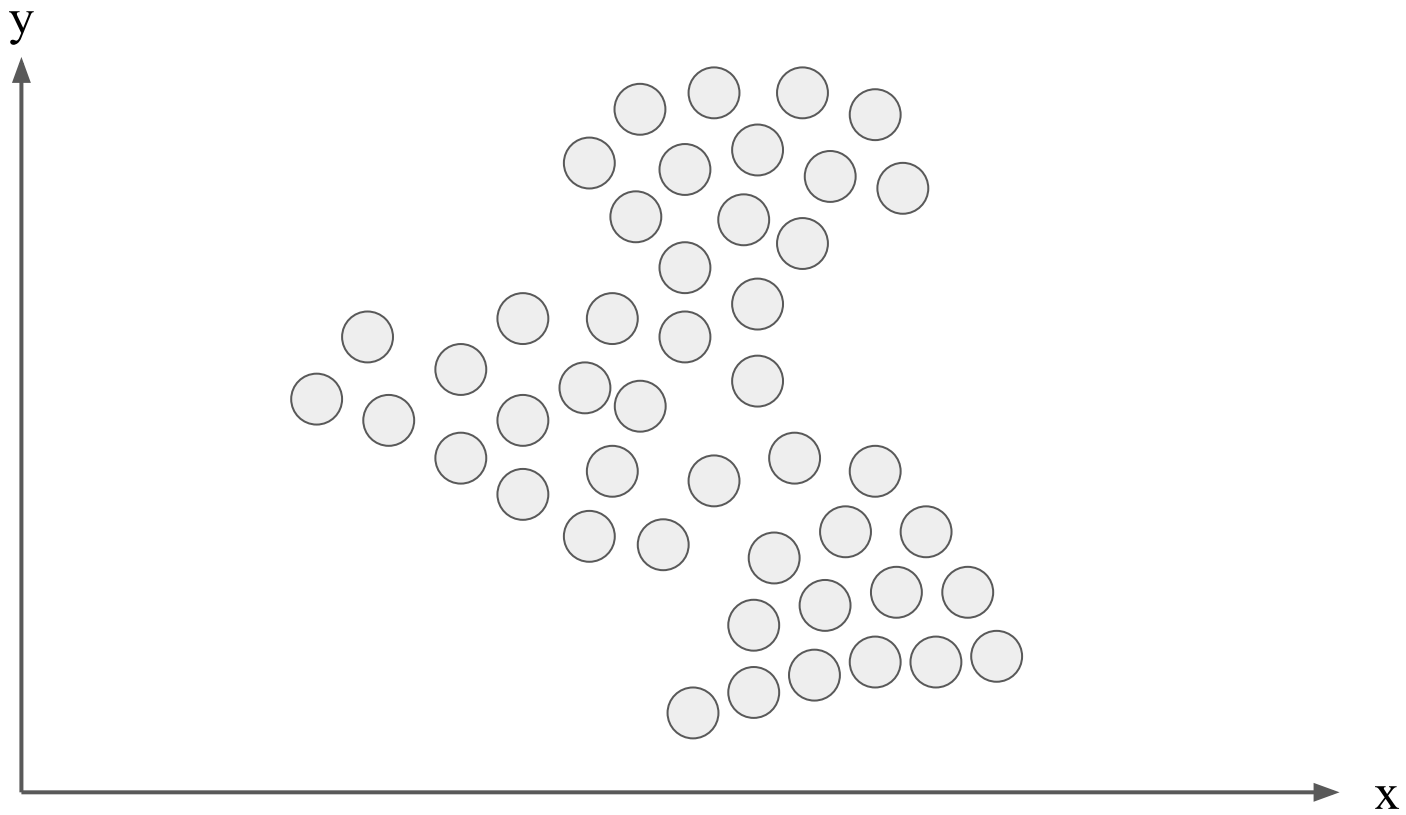
- Dataset (e.g. documents represented by vectors)
- k: number of clusters (which you must decide beforehand)

Output:

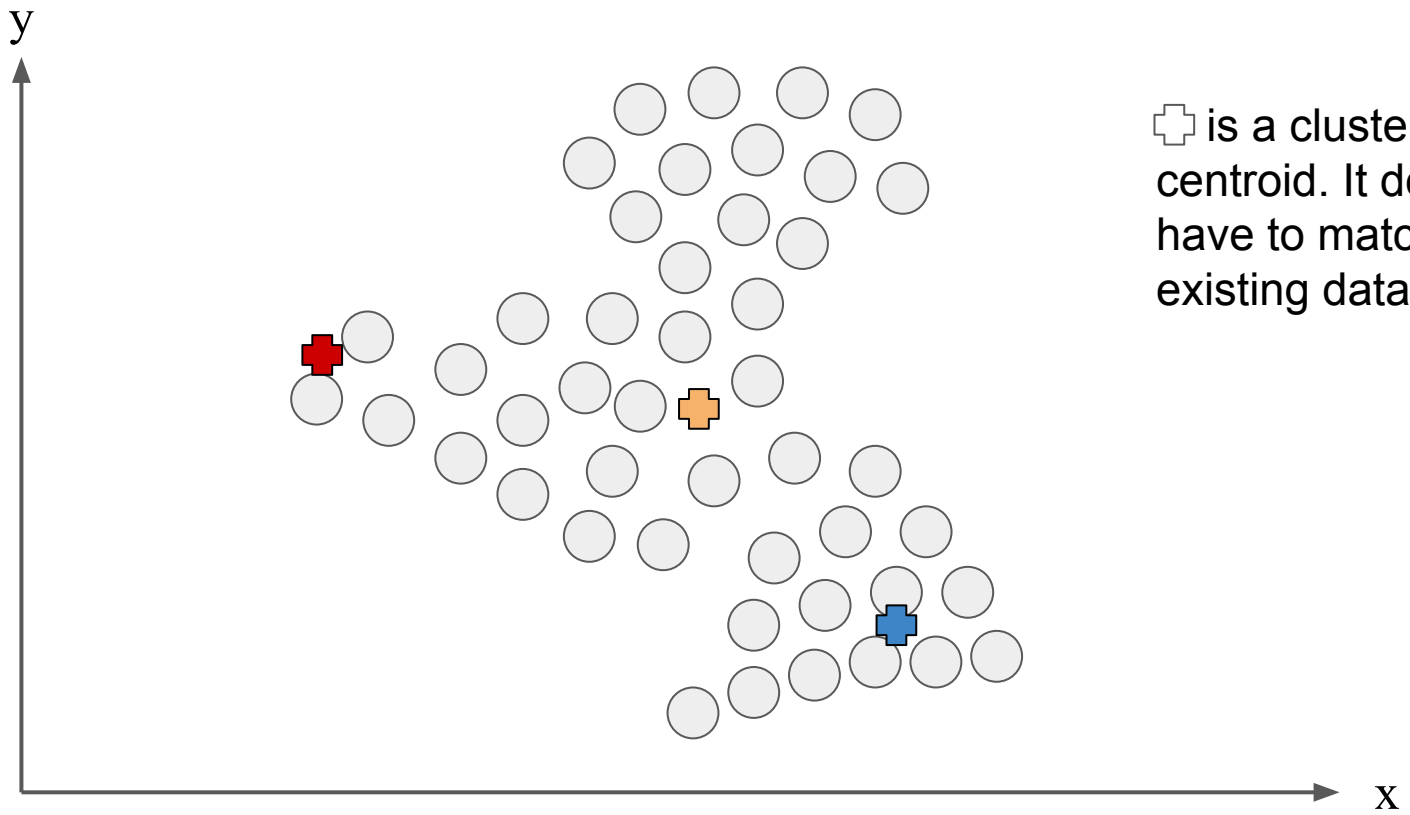
- Dataset **partitioned** into k clusters
- Each cluster represented by its centre point (known as the centroid).



Let’s apply K-means to this odd dataset. We’ll use k=3

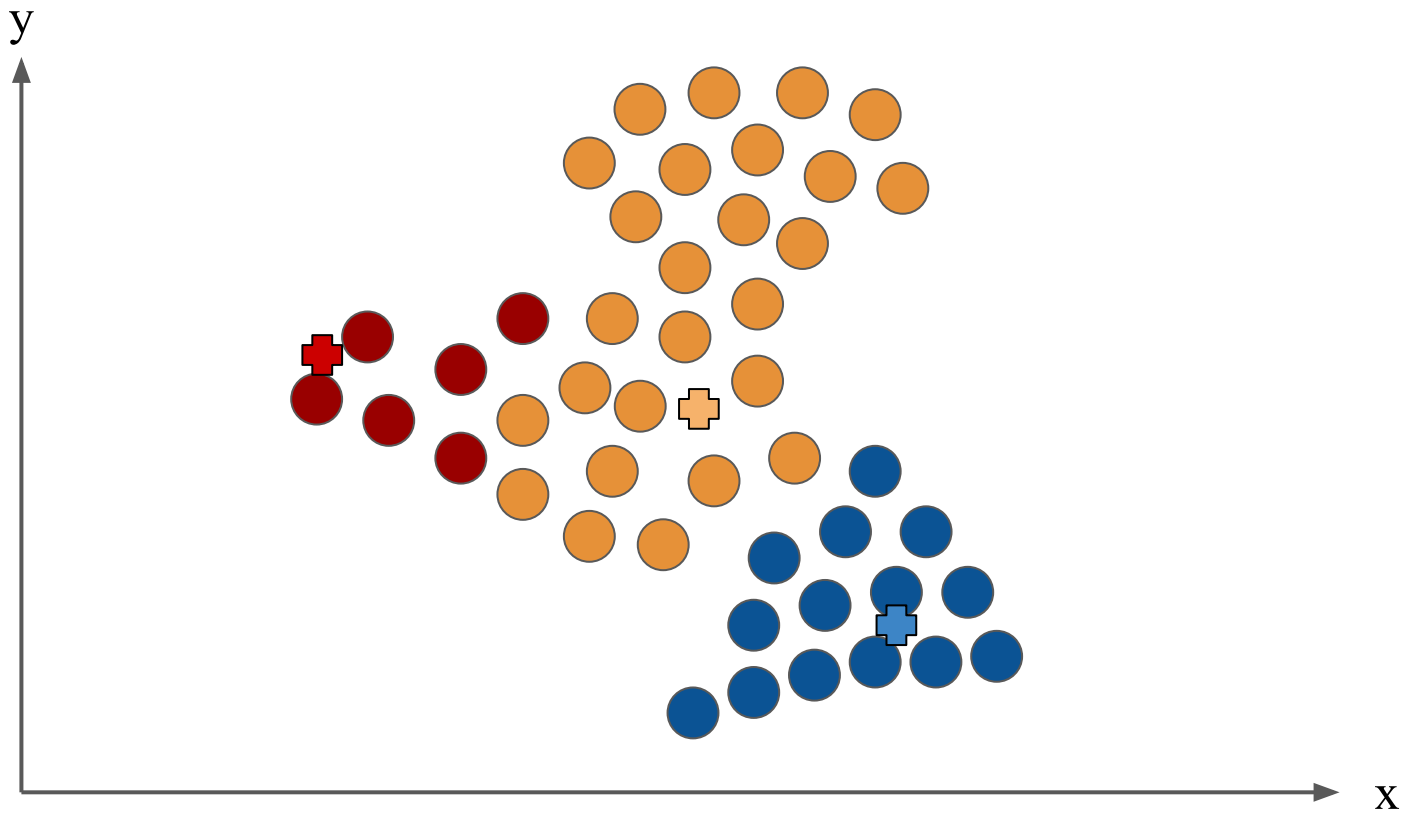


Step 0: Pick k random cluster centres (known as centroids)

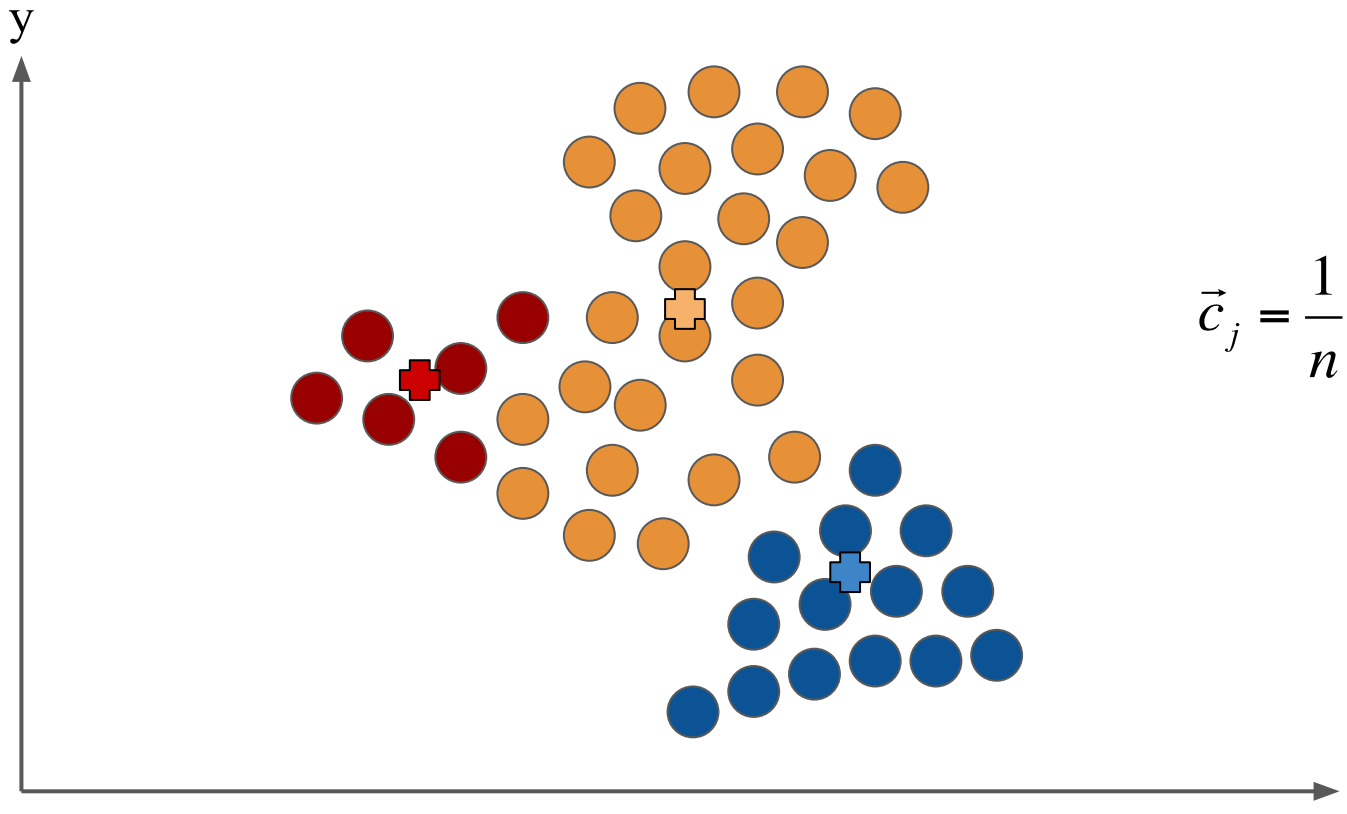


is a cluster centroid. It does not have to match an existing data point

Step 1: Assign each data point to its closest centroid

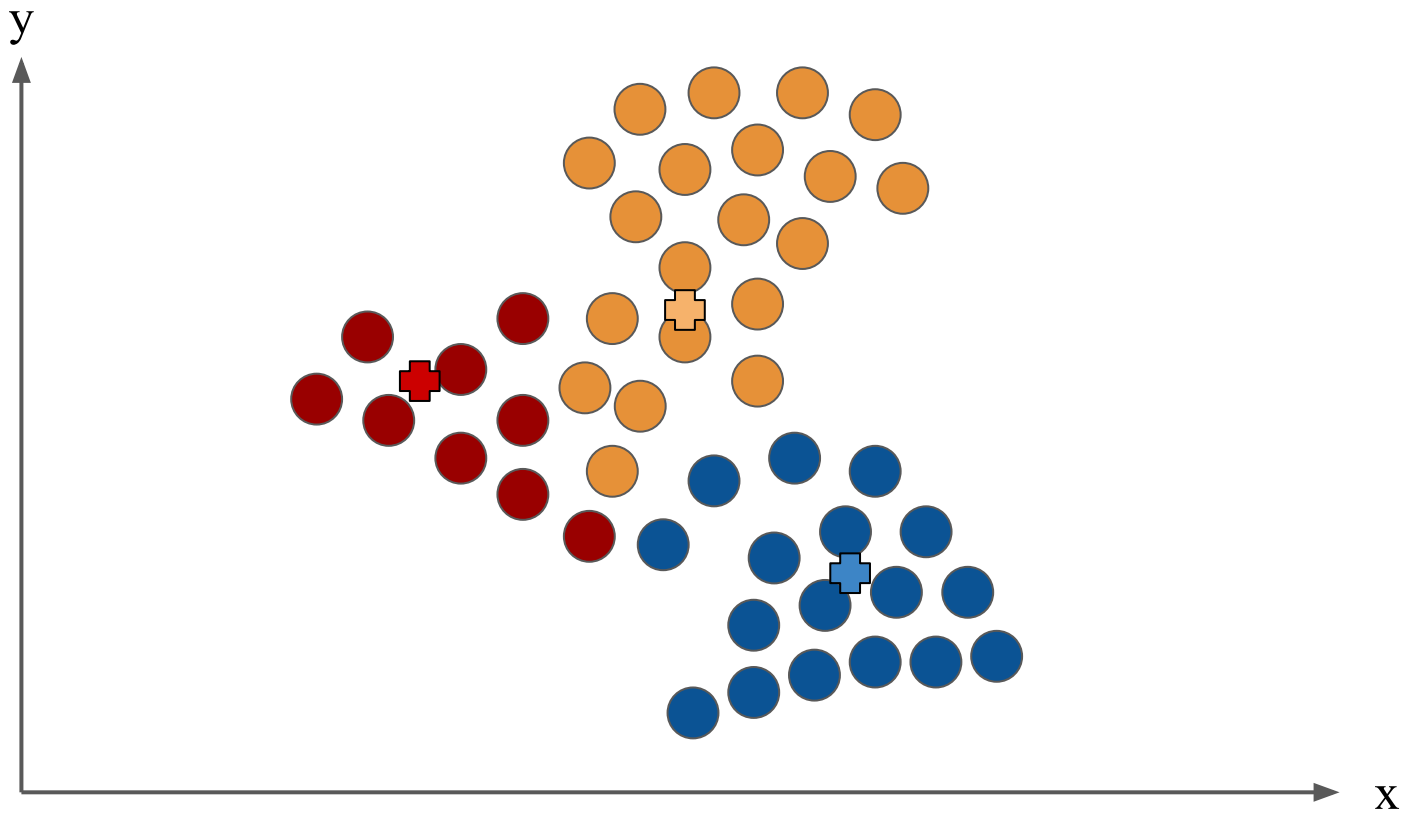


Step 2: Recalculate the centroids (as the average of assigned points)

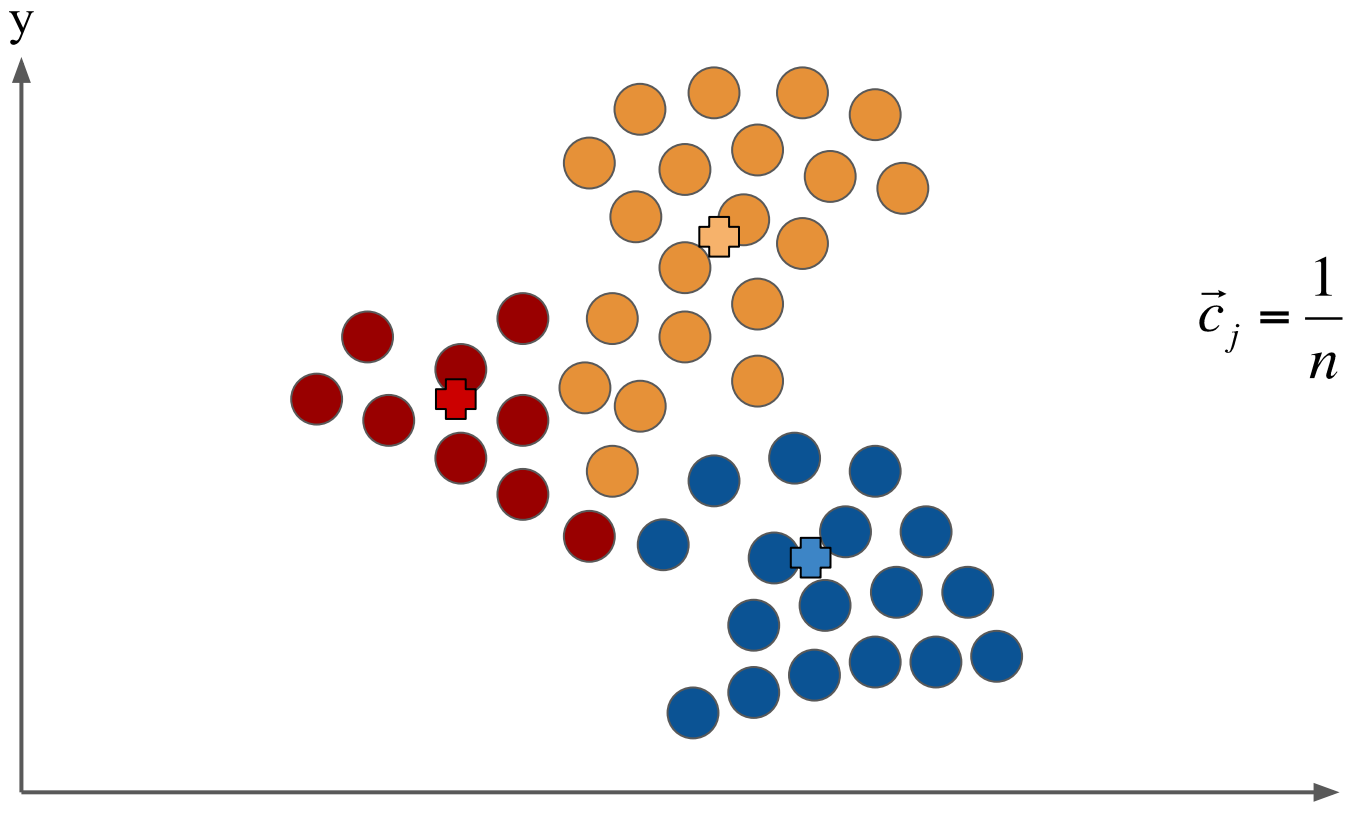


$$\vec{c}_j = \frac{1}{n}\sum \vec{d}_i$$

Repeat Steps 1 and 2 until convergence (1: Assign clusters)

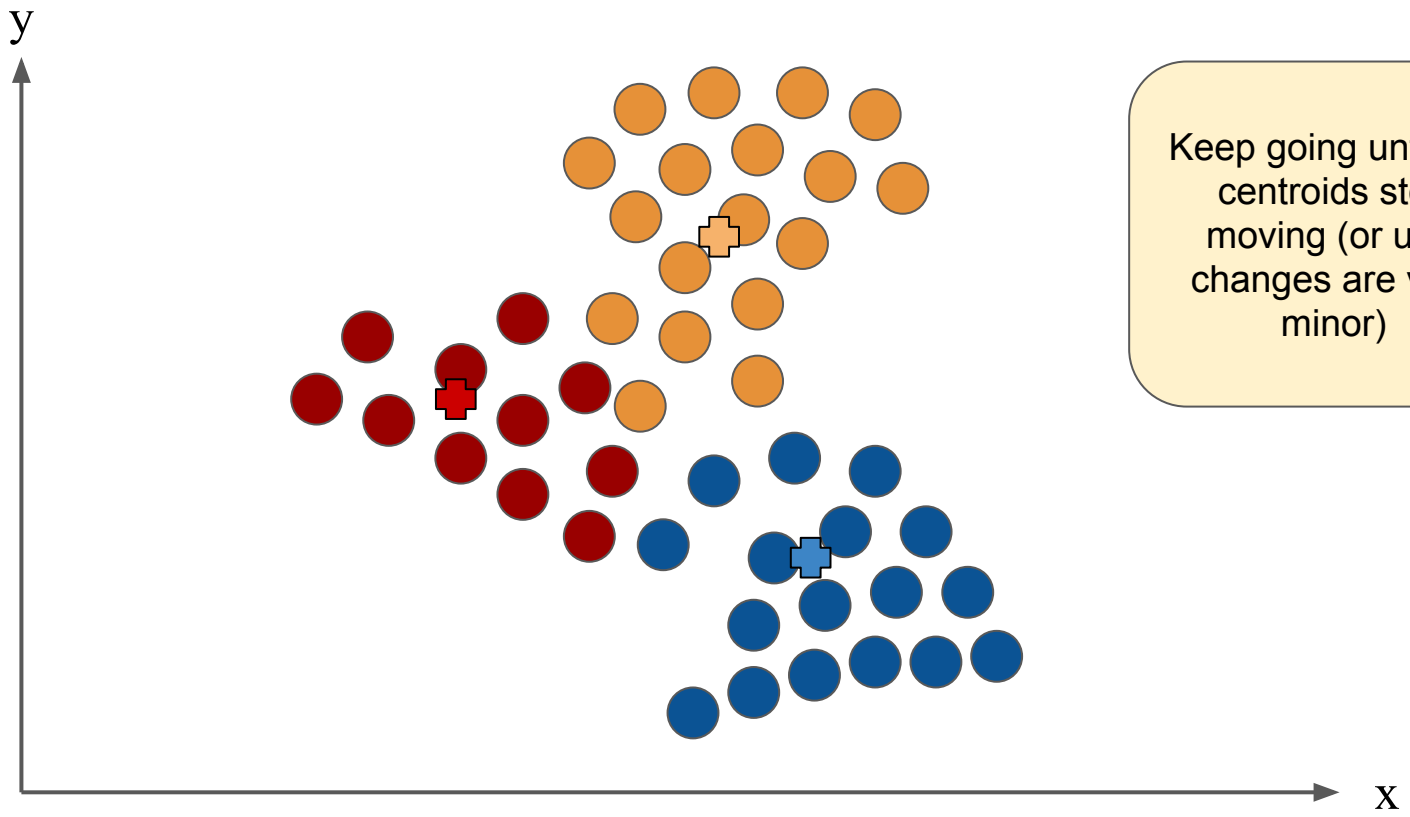


Repeat Steps 1 and 2 until convergence (2: Recalculate centroids)



$$\vec{c}_j = \frac{1}{n}\sum \vec{d}_i$$

Repeat Steps 1 and 2 until convergence (1: Assign again)



Keep going until the centroids stop moving (or until changes are very minor)

Summary of K-Means Clustering Algorithm Steps

- Step 0: Select k centroids
  - Could randomly select k data points from data set
- Step 1: Assign each data point to its nearest centroid
- Step 2: Recalculate the centroids (as the average of assigned points)

Then repeat Steps 1 and 2 until convergence

The Good and not so Good of K-Means

- Good:
  - Tends to converge quickly
- Not so good:
  - Sensitive to choice of initial seeds (may be local minima)
  - But: there are some clever methods for picking initial points

Complexity: O(N*K) (N= #Documents, K=#clusters)

Choosing the initial centroid points for Step 0

This is a whole area of research

Some classic approaches:

- Forgy partition:
  - Randomly select k data points from the dataset and use them as your initial centroids
- Random Partition:
  - Assign each point randomly to a cluster and calculate the centroids from each cluster

Online Demo

https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

Selecting k is tricky!

K-means requires you to decide on the k and will give very different clusters for different values of k.

Too small k

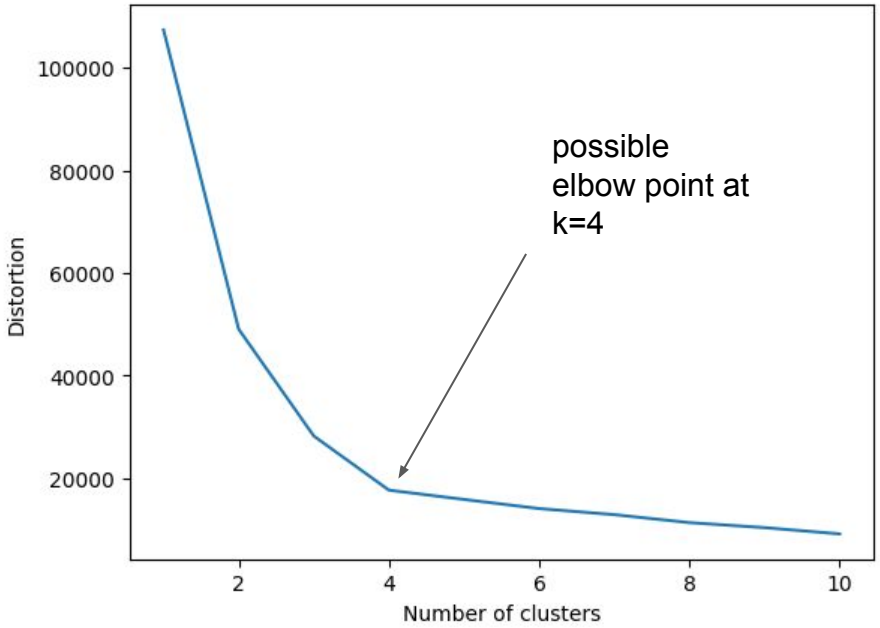
- Important clusters are merged losing information

Too big k

- Make clusters with few data points. Too fine grained

With 2D and even 3D data, we can visualize and decide k visually. We need a general approach to choosing k.

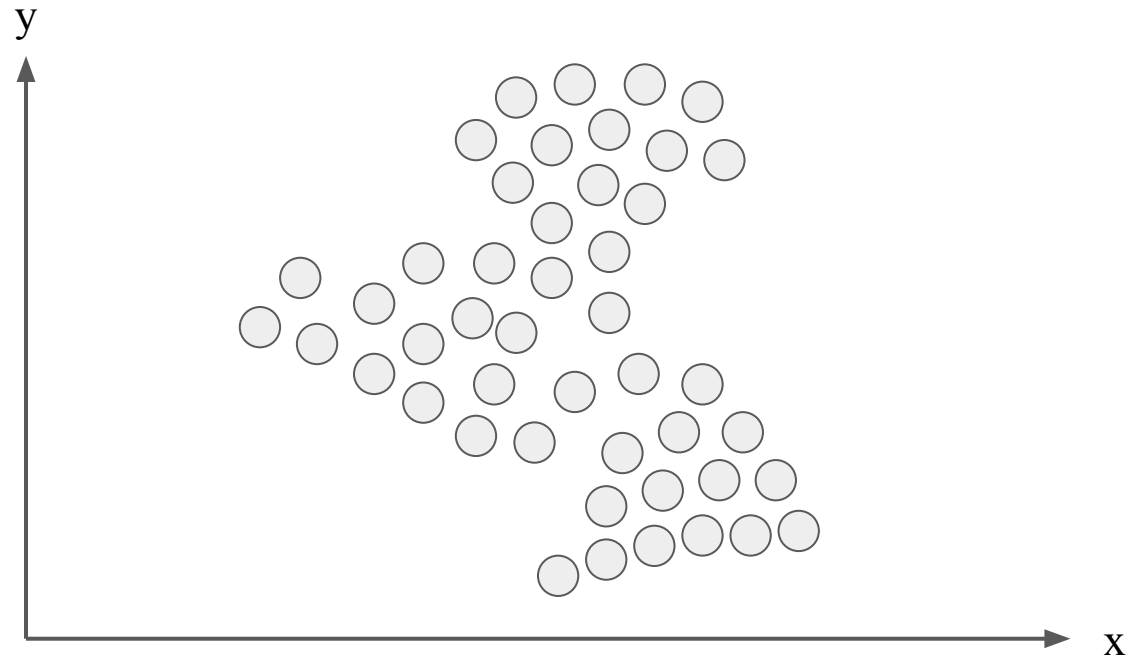
The Elbow method of picking k



- Compute distortion = sum of squared distance of each point to its centroid
- Graph distortion against the number of clusters (k)
- Look for a flattening of the curve (the elbow point) - gives good choice of k
  - little gained with a value of k larger than this
- But there's some arguments against it: https://arxiv.org/abs/2212.12189

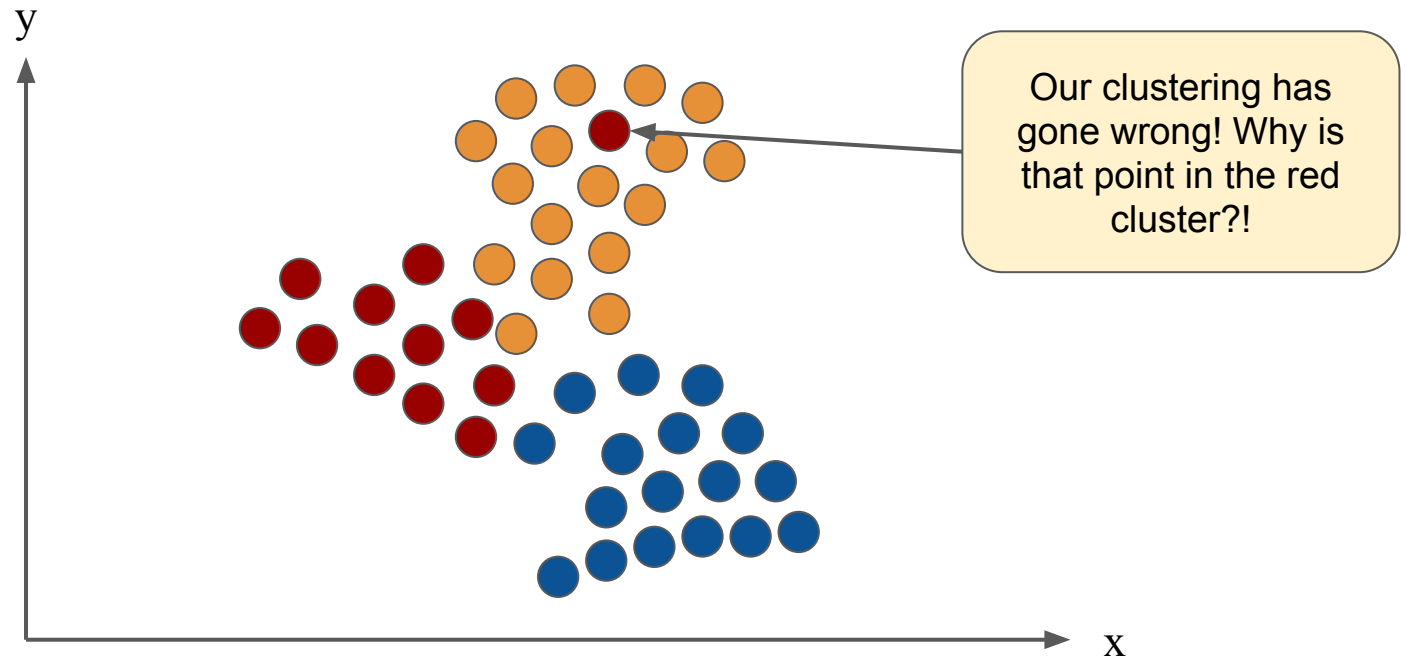
Measuring clustering

There is no definitive measure for clustering. The **silhouette coefficient** does provide one metric to judge if individual data points have been put in the right cluster. Let's look at a bad clustering:



Measuring clustering

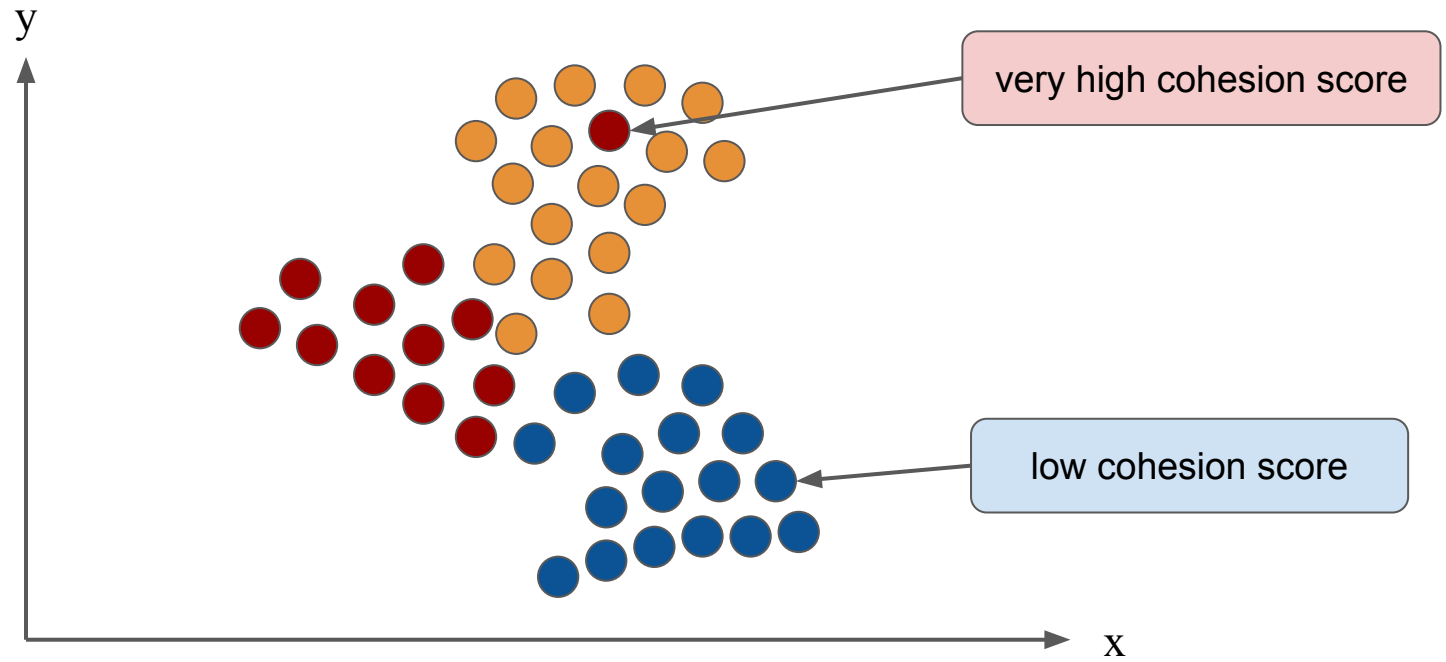
There is no definitive measure for clustering. The **silhouette coefficient** does provide one metric to judge if individual data points have been put in the right cluster. Let's look at a bad clustering:



Measuring clustering

Calculating silhouette coefficient for a point i uses:

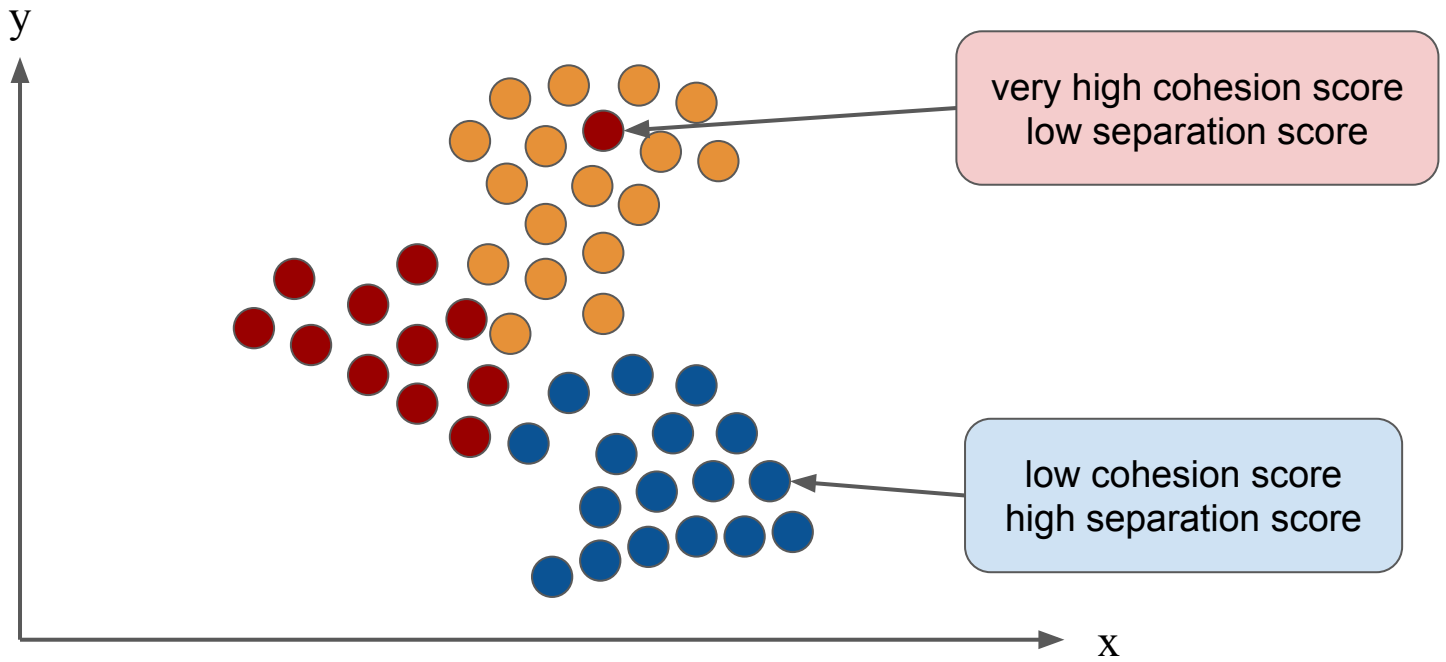
- **cohesion score:** average distance of i to the objects in the same cluster (you want a small cohesion score)



Measuring clustering

Calculating silhouette coefficient for a point i uses:

- **cohesion score:** average distance of i to the objects in the same cluster
- **separation score:** min (average distance of i to objects in another cluster)



Calculating silhouette coefficient

For an individual object, i

- Cohesion (a) = average distance of i to the objects in the same cluster
- Separation (b) = min (average distance of i to objects in another cluster)

Silhouette:

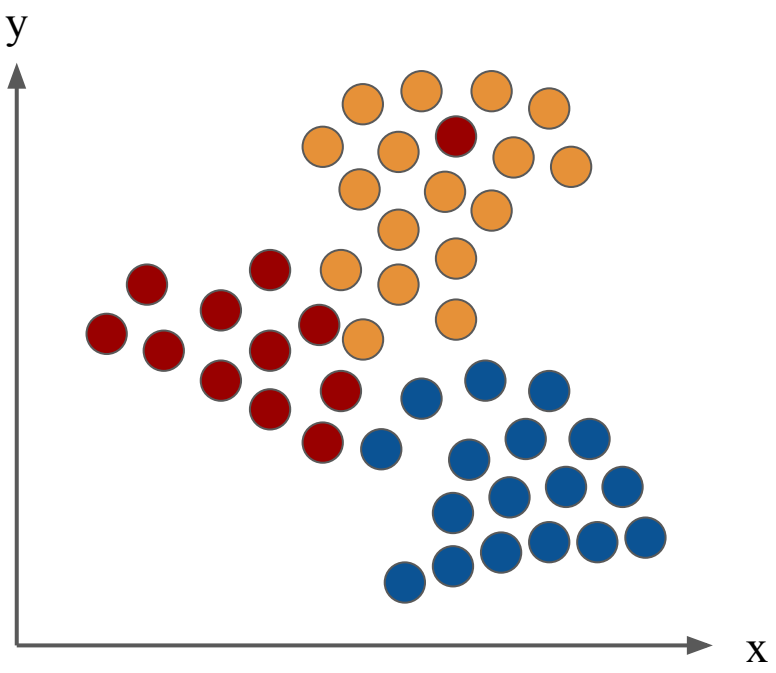
$s(i) = 1 - a/b$ if $a < b$

$s(i) = 0$ if $a = b$

$s(i) = b/a - 1$ if $a > b$

Silhouette ranges from −1 to +1.

A silhouette close to 1 implies an object is in the correct cluster, while a silhouette close to −1 implies the object is in the wrong cluster.



Web-scale K-Means

Problem: Running similarity over all items in the dataset is slow, especially for BIG datasets.

Idea: Instead of comparing against all objects, compare against a sample

Solution: Mini-batch K-Means uses "mini-batches" to reduce computation time.

- Mini-batches are random samples of the input data.
- Dramatically reduce the convergence time
- Only slightly worse than running full algorithm

Algorithm 1 Mini-batch k-Means.

Given: k, mini-batch size b, iterations t, data set X
Initialize each $\mathbf{c} \in C$ with an $\mathbf{x}$ picked randomly from X
$\mathbf{v} \leftarrow 0$
for $i = 1$ to t **do**
  $M \leftarrow b$ examples picked randomly from X
  for $\mathbf{x} \in M$ **do**
    $\mathbf{d}[\mathbf{x}] \leftarrow f(C, \mathbf{x})$ // Cache the center nearest to $\mathbf{x}$
  end for
  for $\mathbf{x} \in M$ **do**
    $\mathbf{c} \leftarrow \mathbf{d}[\mathbf{x}]$ // Get cached center for this $\mathbf{x}$
    $\mathbf{v}[\mathbf{c}] \leftarrow \mathbf{v}[\mathbf{c}] + 1$ // Update per-center counts
    $\eta \leftarrow \frac{1}{\mathbf{v}[\mathbf{c}]}$ // Get per-center learning rate
    $\mathbf{c} \leftarrow (1 - \eta)\mathbf{c} + \eta\mathbf{x}$ // Take gradient step
  end for
end for

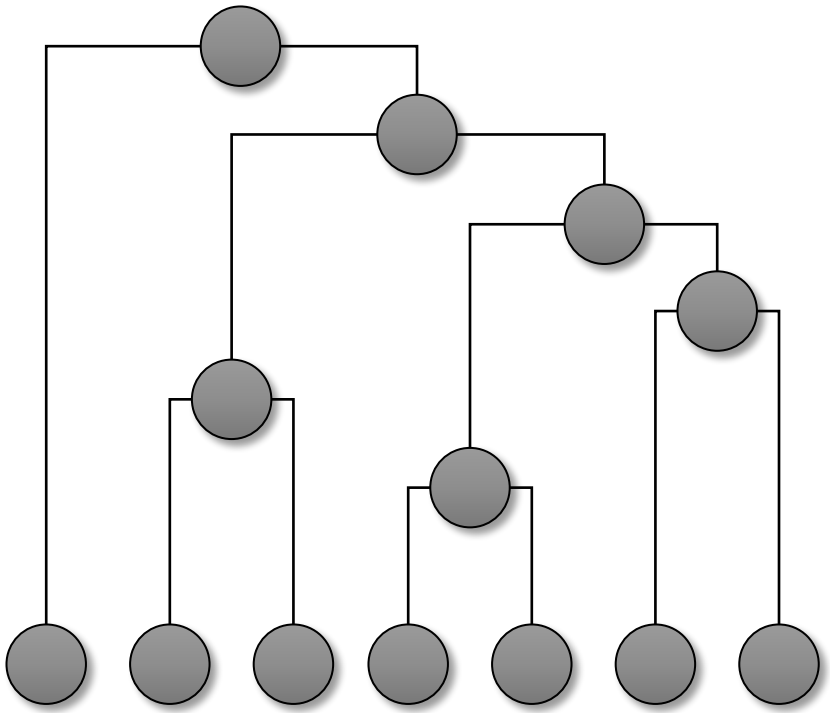
Sculley, D. (2010, April). Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web* (pp. 1177-1178).

K-means summary

- K-means is the usually choice for an all-purpose clustering algorithm
  - Will always converge (given enough time)
  - However, may converge to a local minimum
- K-means is fast and efficient
  - Minibatch modifications
  - Search-based retrieval using centroid as a query
- How do we improve the quality of a k-means clustering?
  - Pick seeds that are sufficiently far apart (kmeans++)
  - Run k-means multiple times with different seeds
  - Try different values of k, select the best value
- K-Means produces clusters that are:
  - Flat (not hierarchical)
  - Non-overlapping (disjoint)
  - Hard assignments

Other types of clustering: Hierarchical clustering

- Results in a tree-like representation with clusters of highly similar documents nested within clusters of less similar objects
- Can be agglomerative (bottom-up) or divisive (top-down)

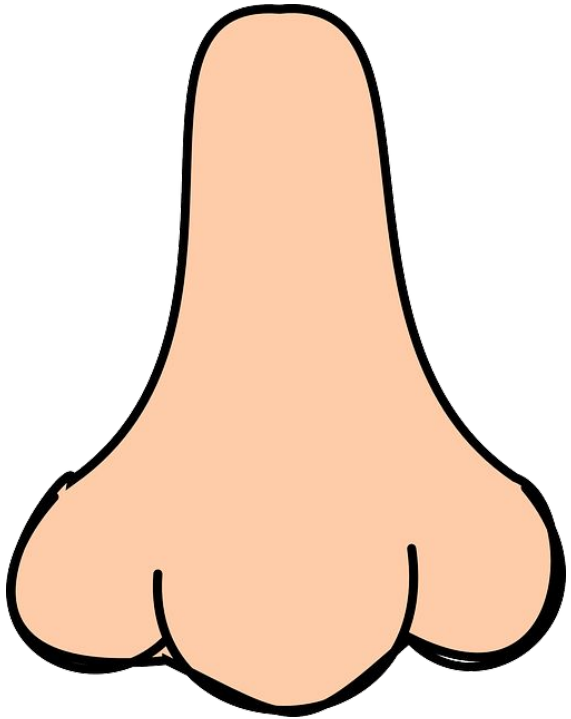


Comparing K-Means vs. Hierarchical

- K-Means is more efficient than Hierarchical
  - Implementation is $O(KN)$ instead of $O(N^3)$ for Hierarchical
- K-Means usually produces clusters of similar quality to Hierarchical
- K-Means is a usual choice for an all-purpose clustering algorithm for a wide range of tasks
  - K needs to be defined upfront! (main drawback)

Cluster Evaluation: Look at your data

- Do a smell test
- Basic checks for oddities
  - e.g. 90% of data in one cluster
- Look at a few examples in each cluster
- (this isn’t exactly systematic)



Cluster Evaluation: Intrinsic

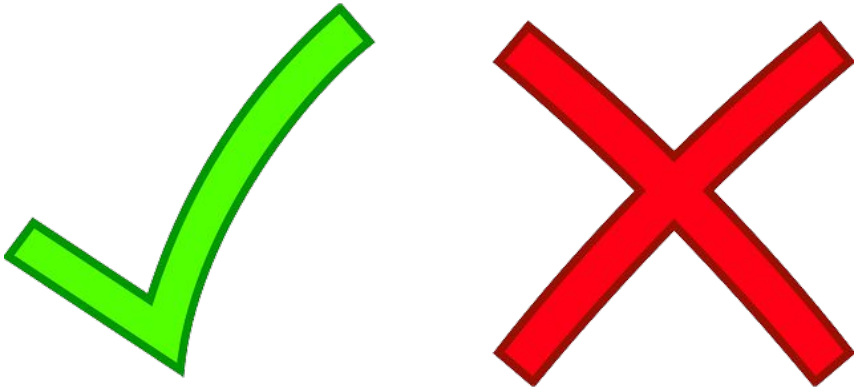
Internally consistent

- Cohesion – Items in a cluster should be similar to one another
- Separation – Clusters should separate objects from one another
- Silhouette Coefficient combines cohesion and separation

Externally consistent (manual human judgments)

- Judge whether pairs of objects are similar enough to be in the same group
- Purity: Measure the extent to which clusters contain a single class. (requires human labeled classes)

Cluster Evaluation: Extrinsic



Are you using the clustering as part of a larger task that can be measured?

- cluster membership could be an input for another system (e.g. a classifier)

Does this clustering improve the performance of the downstream task?

(could compare against random clustering or other methods)

Summary

- Clustering is typically unsupervised
- Very valuable for finding patterns & groupings in a dataset
- Different methods can give different clustering
- K-means clustering is a popular approach
  - Partitions each sample into distinct clusters
  - Need to pick k first
- We don’t have to cluster documents – any objects could be clustered
- Evaluating clustering is hard!
  - There is generally no right answer

Next Lab: Geometric similarity

1. The next lab will involve vectorizing and calculating cosine-similarity of documents.
2. Solution for last lab is up!

colab

spaCy