Tutorial 1

SC4021/CE4034/CZ4034

Q1

- Consider these documents:

Doc1	breakthrough drug for schizophrenia
Doc2	new schizophrenia drug
Doc3	new approach for treatment of schizophrenia
Doc4	new hopes for schizophrenia patients

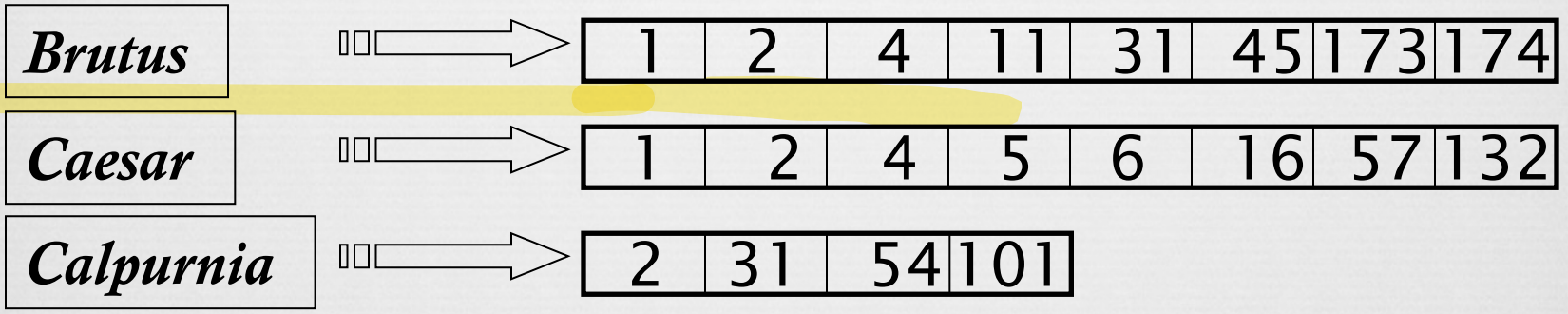
a) Draw the term-document incidence matrix for this document collection

b) Draw the inverted index representation for the collection

Example 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

Example inverted index



Do try this at home

A1.a

Doc1	breakthrough drug for schizophrenia
Doc2	new schizophrenia drug
Doc3	new approach for treatment of schizophrenia
Doc4	new hopes for schizophrenia patients

	Doc1	Doc2	Doc3	Doc4

A1.a

	Doc1	Doc2	Doc3	Doc4
approach	0	0	1	0
breakthrough	1	0	0	0
drug	1	1	0	0
for	1	0	1	1
hope	0	0	0	1
new	0	1	1	1
of	0	0	1	0
patient	0	0	0	1
schizophrenia	1	1	1	1
treatment	0	0	1	0

A1.b

approach =>
breakthrough =>
drug =>
for =>
hope =>
new =>
of =>
patient =>
schizophrenia =>
treatment =>

	Doc1	Doc2	Doc3	Doc4
approach	0	0	1	0
breakthrough	1	0	0	0
drug	1	1	0	0
for	1	0	1	1
hope	0	0	0	1
new	0	1	1	1
of	0	0	1	0
patient	0	0	0	1
schizophrenia	1	1	1	1
treatment	0	0	1	0

A1.b

approach => 3
breakthrough => 1
drug => 1, 2
for => 1, 3, 4
hope => 4
new => 2, 3, 4
of => 3
patient => 4
schizophrenia => 1, 2, 3, 4
treatment => 3

	Doc1	Doc2	Doc3	Doc4
approach	0	0	1	0
breakthrough	1	0	0	0
drug	1	1	0	0
for	1	0	1	1
hope	0	0	0	1
new	0	1	1	1
of	0	0	1	0
patient	0	0	0	1
schizophrenia	1	1	1	1
treatment	0	0	1	0

Q2

- For the document collection shown in Q1, what are the returned results for these two queries:

a) schizophrenia AND drug

b) for AND NOT (drug OR approach)

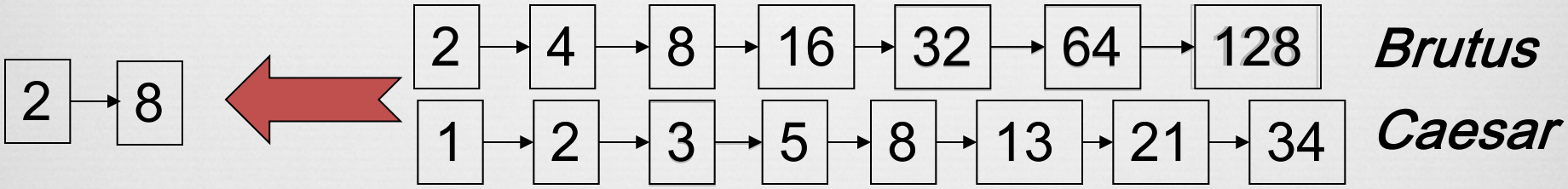
Incidence vectors

- 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

The merge

- Walk through the two postings simultaneously, in time linear in the total number of postings entries



If the list lengths are *x* and *y*, the merge takes O(*x*+*y*) operations

Crucial: postings sorted by docID

Do try this at home

A2.a

a) schizophrenia AND drug

1111 AND 1100 = 1100

b) for AND NOT
(drug OR approach)

1100 OR 0010 = 1110

NOT 1110 = 0001

1011 AND 0001 = 0001

	Doc1	Doc2	Doc3	Doc4
approach	0	0	1	0
breakthrough	1	0	0	0
drug	1	1	0	0
for	1	0	1	1
hope	0	0	0	1
new	0	1	1	1
of	0	0	1	0
patient	0	0	0	1
schizophrenia	1	1	1	1
treatment	0	0	1	0

A2.b

approach	=> 3
breakthrough	=> 1
drug	=> 1, 2
for	=> 1, 3, 4
hope	=> 4
new	=> 2, 3, 4
of	=> 3
patient	=> 4
schizophrenia	=> 1, 2, 3, 4
treatment	=> 3

a) schizophrenia AND drug

merge (1,2,3,4) and (1,2)

b) for AND NOT (drug OR approach)

merge (1,2) or (3)

merge (1,3,4) and not (1,2,3)

Q3

- The table below gives the sizes of postings lists for tokens. a, b, c, d, e and f

Term	a	b	c	d	e	f
Postings size	174	350	637	9066	950	252

a) Recommend a query processing order for the Boolean query: (a OR d) AND (b OR e) AND (c OR f)

b) Estimate the minimum and maximum possible number of results for the query: (c OR e) AND (NOT a)

Do try this at home

A3.a

Term	a	b	c	d	e	f
Postings size	174	350	637	9066	950	252

(c OR f) $\simeq$ 637 + 252 = 889

(b OR e) $\simeq$ 350 + 950 = 1300

(a OR d) $\simeq$ 174 + 9066 = 9240

(c OR f) AND (b OR e) $\simeq$ 889

(c OR f) AND (b OR e) AND (a OR d) $\simeq$ 889

A3.b

Term	a	b	c	d	e	f
Postings size	174	350	637	9066	950	252

(c OR e) : 950

(NOT a): 8892
(9066-174 = 8892)

(c OR e) AND (NOT a): 776
(950-174=776)

Min: 776 (best case)

(c OR e) : 637+950= 1587

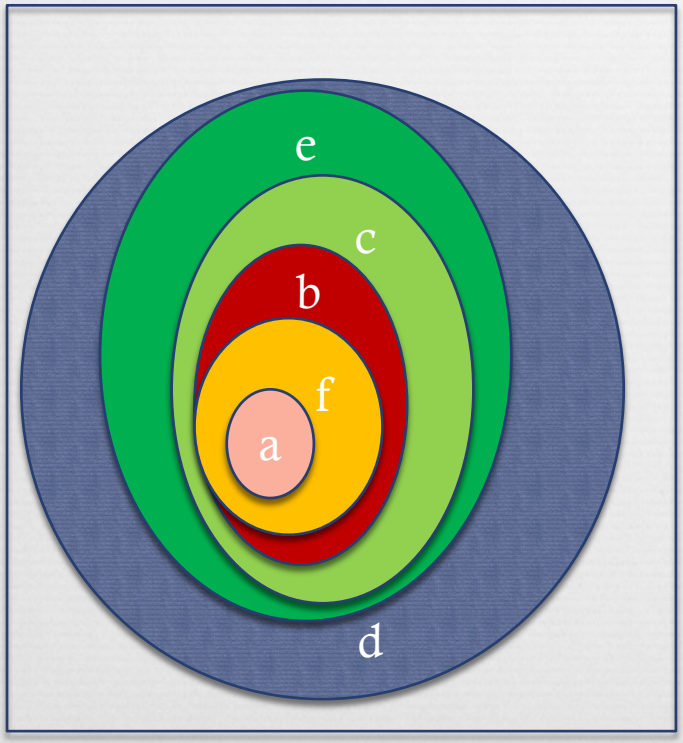
NOT a: 11250
(9066+950+252+637+350) = 11250)

(c OR e) AND (NOT a): 1587

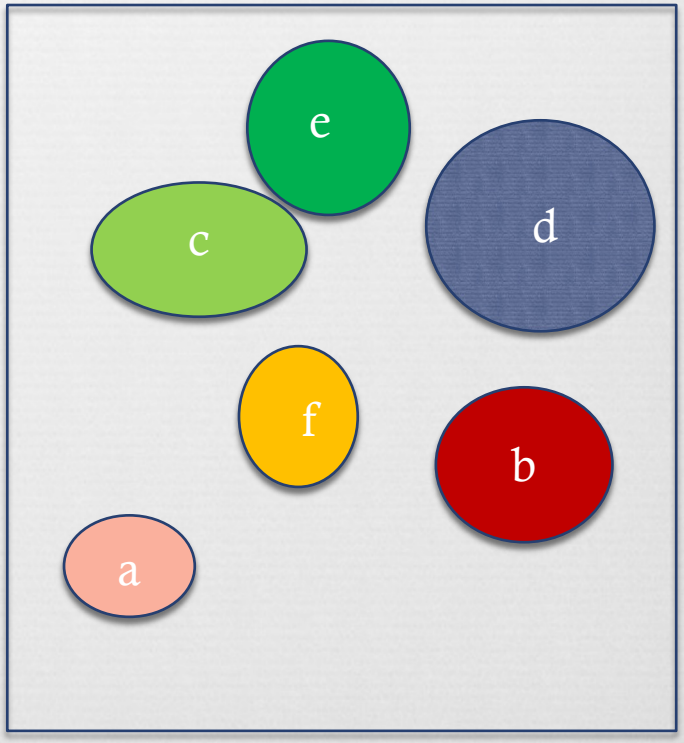
Max: 1587 (worst case)

A3.b

Best case



Worst case



Q4

- For a conjunctive query (e.g., t1 AND t2 AND t3), is processing postings lists in order of size guaranteed to be optimal? Explain why it is or give an example where it isn't.

Do try this at home

A4

- Consider the query “t1 AND t2 AND t3”
  - t1 = 100, t2 = 105, t3 = 110
  - t1 AND t2 = 100
  - t1 AND t3 = 0
- The ordering t1, t2, t3 requires 100+105+100+110=415 steps through the postings lists
- The ordering t1, t3, t2 requires 100+110+0+0=210 steps through the postings lists

Q5

- For the following Porter stemmer rule group:
  - sses → ss (e.g., caresses → caress)
  - ies → i (e.g., ponies → poni)
  - ss → ss (e.g., caress → caress)
  - s → (e.g., cats → cat)

a) What is the purpose of including an identity rule such as SS → SS?

Q5

b) Applying just this rule group, what will the following words be stemmed to?
   - circus
   - canaries
   - boss

c) What rule should be added to stem *pony*, considering the stemming of *ponies*?

   ponies ➔ poni

Do try this at home

A5

a) Otherwise, “s -> (null)” will change “caress -> cares”

b) circu, canari, boss

c) y -> i

Tutorial 2

SC4021/CE4034/CZ4034

Q1

Consider the three words "fly", "flier" and "lier"

a) List the 3-grams for each word

b) Compute the **Jaccard** coefficient between the word 'lie' and each of the three words. Which word could be the suggested spell-corrected word for the query 'lie'?

c) Compute the edit distance between 'lie' and each of the three words by using **Levenshtein** distance algorithm. Which word could be the suggested spell-corrected word for the query 'lie'?

Jaccard coefficient (J)

- A commonly-used measure of overlap between sets.
- Let X and Y be two sets; then

$$J = |X \cap Y| \; / \; |X \cup Y|$$

- Jaccard coefficient is always between 0 and 1

Levenshtein distance

- Levenshtein distance: is a dynamic programming a algorithm that perform tabular computation to detect the minimum edit distance.
- D(n,m) is computed using a **bottom-up** approach:
  - We compute the edit distance on a smaller sub-string
  - use previous solution to compute the overall edit distance
  - i.e., compute D(i,j) for all i (0<i<n) and j (0<j<m)

Levenshtein distance

- First, we need to align the string:

*	M	E	*	N	T	I	O	N
E	X	E	C	U	T	I	O	N

- Then we compute the insertion, substitution and deletion needed

*	M	E	*	N	T	I	O	N
E	X	E	C	U	T	I	O	N
i	*s*		*i*	*s*				

Levenshtein distance

- Initialization: Distance between string N and M to null

$$D(i,0) = i$$
$$D(0,j) = j$$

- Recurrent relation:
  - for each i = 1..N
    - for each j = 1..M

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 & \textit{delete cost} \\ D(i,j-1) + 1 & \textit{insert cost} \\ D(i-1,j-1) + \begin{cases} 1 \text{ if } N(i) \neq M(j) \\ 0 \text{ if } N(i) == M(j) \end{cases} & \textit{substitution cost} \end{cases}$$

Do try this at home

A1.a

- fly: fly
- flier: fli, lie, ier
- lier: lie, ier
- lie: lie

A1.b

Jaccard coefficient = $|X \cap Y| / |X \cup Y|$

lie & fly:

$\{lie\} \cap \{fly\} = \{\}$ $\{lie\} \cup \{fly\} = \{lie, fly\}$

$J = |\{\}| / |\{lie, fly\}| = 0$

lie & flier:

$\{lie\} \cap \{fli, lie, ier\} = \{lie\}$ $\{lie\} \cup \{fli, lie, ier\} = \{fli, lie, ier\}$

$J = 1/3 = 0.33$

lie & lier:

$J = |\{lie\} \cap \{lie, ier\}| / |\{lie, ier\}| = 1/2 = 0.5$

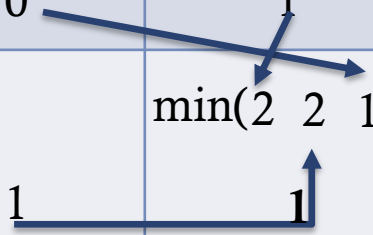
A1.b

- lie & fly: J = 0 D = 1
- lie & flier: J = 0.33 D = 0.66
- lie & lier: J = 0.5 D = 0.5

- 0.5 is the smallest edit distance, so ‘lier’ is more likely to be a spell-corrected word

A1.c

	#	f	l	y
#	0	1	2	3
l	1	min(2 2 1) **1**	min(3,2,1) **1**	min(4,2,3) **2**
i	2	min(2,3,2) **2**	min(2,3,2) **2**	min(3,3,2) **2**
e	3	min(3,4,3) **3**	min(3,4,3) **3**	min(3,4,3) **3**



A1.c

	#	l	i	e	r
#	0	1	2	3	4
l	1	2,2,**0**	3,**1**,2	4,**2**,3	5,**3**,4
i	2	**1**,3,2	2,2,**0**	3,**1**,2	4,**2**,3
e	3	**2**,4,3	**1**,3,2	2,2,**0**	3,**1**,2

A1.c

	#	f	l	i	e	r
#	0	1	2	3	4	5
l	1	2,2,**1**	3,2,**1**	4,**2**,3	5,**3**,4	6,**4**,5
i	2	**2**,3,**2**	**2**,3,**2**	3,3,**1**	4,**2**,3	5,**3**,4
e	3	**3**,4,**3**	**3**,4,**3**	**2**,4,3	3,3,**1**	4,**2**,3

Q2

a) Given the word "cat", compute all its possible right rotations for a permuterm query.

b) Generate 5 wild-card queries that can retrieve the word "cat"

c) Can you think of an English term that matches the permuterm query **er$fi***, but does not satisfy the query fi*mo*er?

Permuterm query

- Permuterm queries are needed to handle wild-cart queries with * in the middle of the term: **co*tion**
- Generate all the possible right rotation of a term and index all the permutation.

i.e. For term ***hello,*** index it under:
hello$, ello$h, llohe, lohel, o$hell where $ is a special symbol

- To execute a query:
  1. Rotate the term to the right until * appear at the end
  2. Look in the index if there are some match for the rotated query
  3. Retrieve the normal form of the rotated query

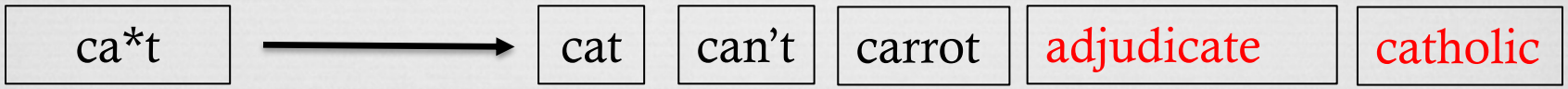
Do try this at home

A2.a

1. cat$
2. at$c
3. t$ca

A2.b

1. c*t
2. ca*t
3. *at
4. *t
5. ca*



A2.c

- Satisfy: **er$fi* => fi*er$**

Not satisfy: **fi*mo*er**

1. Fisher
2. Fiancer
3. Fiber

Q3

Term	Doc1	Doc2	Doc3
angels fools	1	0	0
angels rush	1	0	1
angels fear	0	0	1
fools rush	1	0	0
fear fools	0	1	0
fear to	0	1	1
where angels	1	0	1
to tread	1	0	0
fear in	0	1	1
rush in	1	0	1

a) Which are the biword boolean queries generated by the following phrase query?
   1. fools rush in
   2. where angels rush in
   3. angels fear to tread

b) Which are (if any) the documents retrieved?

Biword Index

- Index every consecutive pair of terms in the text as a phrase
- Es. *Friends, Romans, Countrymen* would generate the biwords:
  1. *friends romans*
  2. *romans countrymen*
- Longer phrase queries can be broken into the Boolean query on biwords:
- Es. *stanford university palo alto*
- *stanford university* AND *university palo* AND *palo alto*

Do try this at home

A3.a

- “fools rush in” => fools rush AND rush in
- “where angels rush in” => where angels AND angels rush AND rush in
- “angels fear to tread” => angels fear AND fear to AND to tread

A3.b

- “fools rush in” = doc1 → is this always true?
- “where angels rush in” = doc1, doc3
- “angels fear to tread” = null

Q4

term	doc1	doc2	doc3
angels	#36, 174, 252, 651$		#15, 123, 412$
fools	#1, 17, 74, 222$	#8, 78, 108, 458$	
fear		#13, 43, 113, 433$	#18, 328, 528$
in	#3, 37, 76, 444, 851$	#10, 20, 110, 470, 500$	#5, 17, 25, 195$
rush	#2, 66, 194, 321, 702$		#4, 16, 404$
to	#47, 86, 234, 999$	#14, 24, 774, 944$	#19, 319, 599, 709$
tread	#57, 94, 333$		
where	#67, 124, 393, 1001$	#11, 41, 101, 421, 431$;	#14, 36, 736$

- Which document(s), if any, meet each of the following phrase queries, based on the over mentioned positional index?
  (a) "fools rush in"
  (b) "where angels rush in"
  (c) "angels fear to tread"

Positional index

- Extract inverted index entries for each distinct term: ***to, be, or, not.***
- Merge their *doc:position* lists to enumerate all positions with "***to be or not to be***".
  - ***to****:*
    - *2*:1,17,74,222,551; *4*:8,16,190,429,433; *7*:13,23,191; ...
  - ***be****:*
    - *1*:17,19; *4*:17,191,291,430,434; *5*:14,19,101; ...
- Same general method for proximity searches

Do try this at home

A4

- “fools rush in” => doc1

Fools #1, 17, 74, 222$ **rush** #2, 66, 194, 321, 702$
in #3, 37, 76, 444, 851$

- “where angels rush in” => doc3

Where #14, 36, 736$ **angels** #15, 123, 412$ **rush** #4, 16, 404$
in #5, 17, 25, 195$

~~Doc1~~;No positional merge available

Where #67, 124, 393, 1001$ **angels** #36, 174, 252, 651$
rush #2, 66, 194, 321, 702$ **in** #3, 37, 76, 444, 851$

Q5

- We have a three-word text query 'Enjoy a beer' . Below is a table showing the term counts and document term size of each 4 documents. With the information provided, please recommend the top 3 documents which should be returned to the normalized text query.

	enjoy	**a**	**beer**	**size**
Doc1	2	10	5	400
Doc2	3	35	8	500
Doc3	5	40	3	600
Doc4	10	10	6	750

Do try this at home

A5

	enjoy	a	beer	size
Doc1	0.005		0.0125	1
Doc2	0.006		0.016	1
Doc3	0.0083		0.005	1
Doc4	0.0133		0.008	1

Stop words removal: 'a'

Doc1: 0.005 * 0.0125 = 0.0000625 3

Doc2: 0.006 * 0.016 = 0.000096 2

Doc3: 0.0083 * 0.005 =0.0000415 4

Doc4: 0.0133 * 0.008 = 0.000106 1

A5

The returned order of documents:

1. Document 4
2. Document 2
3. Document 1

Tutorial 3

SC4021/CE4034/CZ4034

Q1

- Consider the table of term frequencies for 3 documents denoted Doc1, Doc2, Doc3 below. Compute the tf-idf weights for the terms *car, auto, insurance, best*, for each document, using the idf values from the table below.

term	Doc1	Doc2	Doc3	idf
car	22	4	24	1.65
auto	3	33	0	2.08
insurance	0	33	29	1.62
best	14	0	17	1.5

tf-idf weighting

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

$$w_{t,d} = (1 + \log \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

- Best known weighting scheme in information retrieval
  - 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

Do try this at home

A1

$$w_{t,d} = (1 + \log_{10} tf_{t,d}) \times \log_{10}(N / df_t)$$

tf	Doc1	Doc2	Doc3	idf
car	22	4	24	1.65
auto	3	33	0	2.08
insurance	0	33	29	1.62
best	14	0	17	1.5

1+log tf	Doc1	Doc2	Doc3
car	2.34	1.60	2.38
auto	1.48	2.52	0
insurance	0	2.52	2.46
best	2.15	0	2.23

w	Doc1	Doc2	Doc3
car	3.86	2.64	3.93
auto	3.08	5.24	0
insurance	0	4.08	3.99
best	3.23	0	3.35

term	Doc1	Doc2	Doc3	idf(t)
car	22	4	24	1.65
auto	3	33	0	2.08
insurance	0	33	29	1.62
best	14	0	17	1.5

Q2

ddd.qqq

- Refer to the tf and idf values for four terms and three documents from Q1. Compute the two top scoring documents on the query *best car insurance* for each of the following weighing schemes: (i) nnn.atc; (ii) ntc.atc.

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times \mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

tf-idf example: lnc.ltc

Document: *car insurance auto insurance*
Query: *best car insurance*

Term	Query						Document				Prod
	tf	tf-log	df	idf	tf-idf	norm	tf	tf-log	tf-idf	norm	
auto	0	0	5000	2.3	0	0	1	1	1	0.52	0
best	1	1	50000	1.3	1.3	0.34	0	0	0	0	0
car	1	1	10000	2.0	2.0	0.52	1	1	1	0.52	0.27
insurance	1	1	1000	3.0	3.0	0.78	2	1.3	1.3	0.68	0.53

Doc length $= \sqrt{1^2 + 0^2 + 1^2 + 1.3^2} \approx 1.92$

Score = 0+0+0.27+0.53 = 0.8

Do try this at home

A2

- Find document vectors: (i) nnn (ii) ntc

nnn(doc)	Doc1	Doc2	Doc3
car	22*1*1=22	4*1*1=4	24*1*1=24
auto	3*1*1=3	33*1*1=33	0*1*1=0
insurance	0*1*1=0	33*1*1=33	29*1*1=29
best	14*1*1=14	0*1*1=0	17*1*1=17

ntc(doc)	Doc1	Doc2	Doc3
car	(22*1.65=36.3)/42.4=0.86	(4*1.65=6.6)/87.3=0.08	(24*1.65=39.6)/66.5=0.60
auto	(3*2.08=6.24)/42.4=0.15	(33*2.08=68.64)/87.3=0.79	0*2.08=0
insurance	0*1.62=0	(33*1.62=53.46)/87.3=0.61	(29*1.62=46.98)/66.5=0.71
best	(14*1.5=21)/42.4=0.5	0*1.5=0	(17*1.5=25.5)/66.5=0.38
Doc length	42.4 ($\sqrt{36.3^2+6.24^2+0+21^2}$)	87.3	66.5

A2

- Find the vector for query *best car insurance*: (i,ii) atc

atc(Query)	tf	a	t	at	atc
car	1	0.5+0.5*1/1=1	1.65	1.65	0.60
auto	0	0	0	0	0
insurance	1	0.5+0.5*1/1=1	1.62	1.62	0.59
best	1	0.5+0.5*1/1=1	1.5	1.5	0.54

max(tf)=1
length=2.76

nnn.atc	Doc1	Doc2	Doc3
car	22*0.60=13.2	4*0.60=2.4	24*0.60=14.4
auto	3*0=0	33*0=0	0*0=0
insurance	0*0.59=0	33*0.59=19.47	29*0.59=17.11
best	14*0.54=7.56	0*0.54=0	17*0.54=9.18
SUM	20.76	21.87	40.69

A2

- (ii) ntc.atc

ntc(doc)	Doc1	Doc2	Doc3
car	0.86	0.08	0.60
auto	0.15	0.79	0
insurance	0	0.61	0.71
best	0.5	0	0.38

atc(Query)	atc
car	0.60
auto	0
insurance	0.59
best	0.54

ntc.atc	Doc1	Doc2	Doc3
car	0.86*0.60=0.52	0.08*0.60=0.05	0.60*0.60=0.36
auto	0.15*0=0	0.79*0 =0	0*0=0
insurance	0*0.59=0	0.61*0.59=0.36	0.71*0.59=0.42
best	0.5*0.54=0.27	0*0.54=0	0.38*0.54=0.21
SUM	0.79	0.41	0.99

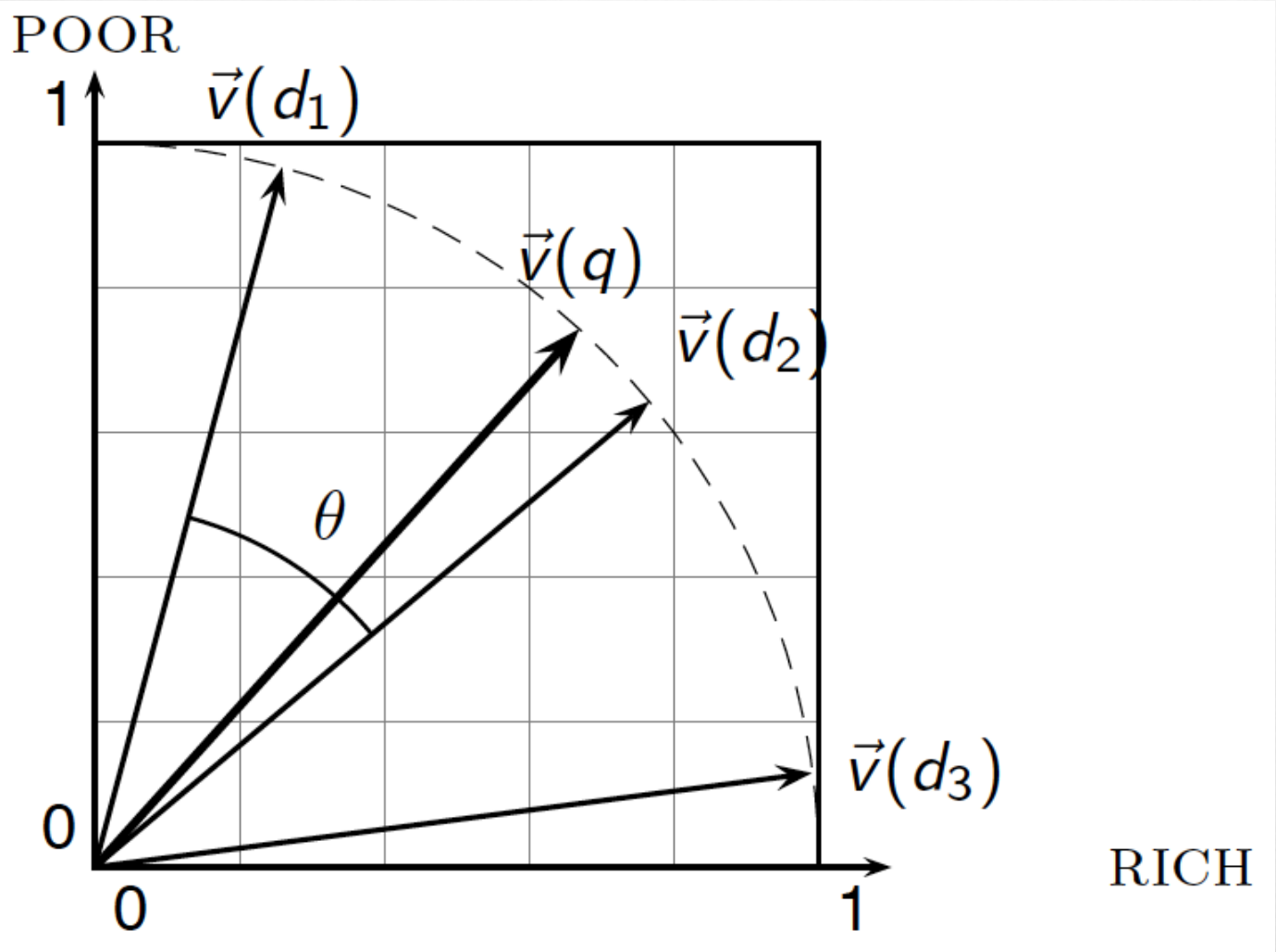
Q3

	Antony and Cleopatra	Julius Caesar	The Tempest
Antony	157	73	0
Brutus	28	157	0
Caesar	232	227	0
Calpurnia	0	10	0
Cleopatra	23	0	37
Mercy	0	10	15
Worser	2	0	1

a) Compute the cosine similarity and the Euclidean distance between the documents and the query: **"caesar mercy brutus"** based on the term-document count matrix above.
b) How does the Euclidean distance change if we normalize the vectors?

NB: Compute the vector space using tf-idf formula of Q1 (both for query and documents)

Cosine similarity illustrated

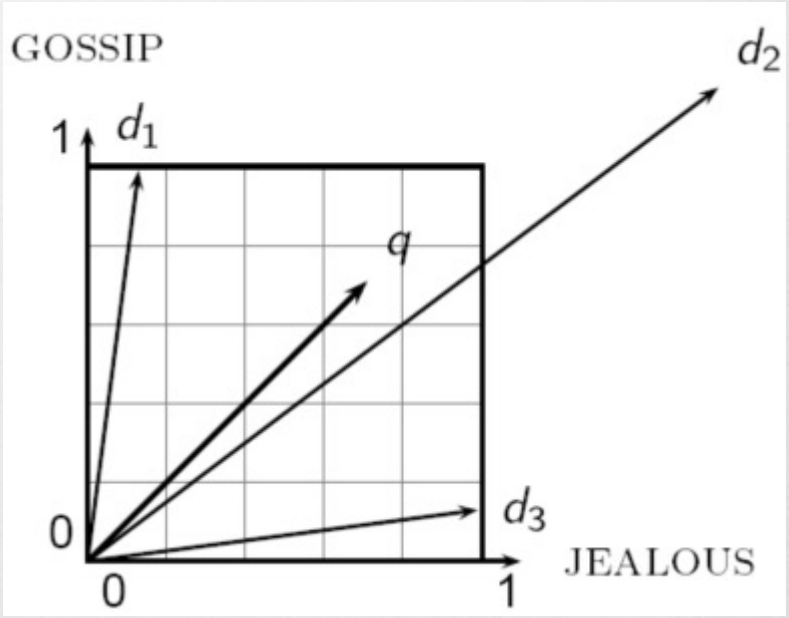


Euclidean distance

- Euclidean distance: the distance between points (x_1,y_1) and (x_2,y_2) is given by:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- Unfortunately, this distance is biased by the length of the vectors. So is not able to detect the correct terms distribution



Cosine for normalized vectors

- For length-normalized vectors, cosine similarity is simply the dot product (or scalar product):

$$\cos(\vec{q},\vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

for q, d length-normalized. $\vec{v}' = \frac{\vec{v}}{|\vec{v}|}$

Do try this at home

A3

First of all, we have to compute the vector space, form the term-frequency matrix, using the formula:

$$w_{t,d} = (1 + \log_{10} \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

- Our collection is composed by 3 documents: {Antony and Cleopatra, Julius Caesar and The Tempest} so N = 3
- The document frequency is computed based on the term-frequency matrix:
  - if the cell has value different form 0, it means that the given word appears in the referred document (df+1)
  - if the cell has 0 term-frequency value, it means that the document doesn't contains the given word (df+0)

With this information it is possible to compute N/df and, as consequence, the idf values for our words.

Ex. Word = **Brutus** df = **2** {Antony and Cleopatra, Julius Caesar} idf = log(3/2) = 0.18

A3

- Compute the vector space $w_{t,d} = (1 + \log tf_{t,d}) \times \log_{10}(N / df_t)$

For documents: Each document is a vector of 7 dimensions, where each dimension is the tfidf value of a term.
For the query: The same of the above applies.

	Antony and Cleopatra	Julius Caesar	The Tempest	Query
Antony	0.56	0.50	0	0
Brutus	0.43	0.56	0	0.18
Caesar	0.59	0.59	0	0.18
Calpurnia	0	0.95	0	0
Cleopatra	0.42	0	0.45	0
Mercy	0	0.35	0.38	0.18
Worser	0.23	0	0.18	0

Vector for 'Antony and Cleopatra': V1 = [0.56, 0.43, 0.59, 0, 0.42, 0, 0.23]
Vector for 'Julius Caesar': V2 = [0.50, 0.56, 0.59, 0.95, 0, 0.35, 0]
Vector for 'The Tempest': V3 = [0, 0, 0, 0, 0.45, 0.38, 0.18]
Vector for the query: Vq = [0, 0.18, 0.18, 0, 0, 0.18, 0]

A3

Euclidean distance:

In general for n-dimensional space, the distance is:

$$eq(d,q) = \sqrt{(d_1 - q_1)^2 + (d_2 - q_2)^2 + \cdots + (d_n - q_n)^2}$$

Cosine similarity:

$$cos\_sim(d,q) = \frac{\sum_{i=1}^{n} q_i d_i}{\sum_{i=1}^{n} \sqrt{q_i^2} \sqrt{\sum_{i=1}^{n} d_i^2}}$$

	Antony and Cleopatra	Julius Caesar	The Tempest
Cosine similarity	0.57 (V1&Vq) 	0.62 (V2&Vq) 	0.35 (V3&Vq) 
Euclidean distance	0.90 (V1&Vq) 	1.22 (V2&Vq) 	0.58 (V3&Vq) 

Normalization	
n (none)	1
c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$

A3

Normalization: By default, we are using cosine normalization aka L2 norm

Normalized values				
	Antony and Cleopatra	**Julius Caesar**	**The Tempest**	**Query**
Antony	0.54	0.36	0	0
Brutus	0.41	0.40	0	0.58
Caesar	0.57	0.42	0	0.58
Calpurnia	0	0.68	0	0
Cleopatra	0.40	0	0.73	0
Mercy	0	0.25	0.62	0.58
Worser	0.22	0	0.29	0
Euclidean distance normalized	0.93 (Normalized_V1 & Normalized_Vq)	0.87 (Normalized_V2 & Normalized_Vq)	1.14 (Normalized_V3 & Normalized_Vq)	

Tutorial 4

SC4021/CE4034/CZ4034

Q1

- Consider again the data of Tutorial 3 (Q2) with *ntc.atc* for the query-dependent scoring.
- Suppose that we were given static quality scores of 0.3 and 0.6 for Doc1 and Doc2, respectively.
- Determine what ranges of static quality score for Doc3 result in it being the first, second or third result for the query *best car insurance*.

A2

- (ii) ntc.atc

ntc(doc)	Doc1	Doc2	Doc3
car	0.86	0.08	0.60
auto	0.15	0.79	0
insurance	0	0.61	0.71
best	0.5	0	0.38

atc(Query)	atc
car	0.60
auto	0
insurance	0.59
best	0.54

ntc.atc	Doc1	Doc2	Doc3
car	0.86*0.60=0.52	0.08*0.60=0.05	0.60*0.60=0.36
auto	0.15*0=0	0.79*0 =0	0*0=0
insurance	0*0.59=0	0.61*0.59=0.36	0.71*0.59=0.42
best	0.5*0.54=0.27	0*0.54=0	0.38*0.54=0.21
SUM	0.79	0.41	0.99

Static quality scores

- We want top-ranking documents to be both *relevant* and *authoritative*
- *Relevance* is being modeled by cosine scores
- *Authority* is typically a query-independent property of a document
- Examples of authority signals
  - Wikipedia among websites
  - Articles in certain newspapers
  - A paper with many citations
  - Many diggs, Y!buzzes or del.icio.us marks
  - (Pagerank)

Net score

- Consider a simple total score combining cosine relevance and authority
- net-score(q,d) = $g(d)$ + cosine(q,d)
  - Can use some other linear combination than an equal weighting
  - Indeed, any function of the two "signals" of user happiness – more later
- Now we seek the top K docs by net score

Do try this at home

A1

	Doc1	Doc2	Doc3
ntc.atc	0.79	0.41	0.99
g(d)	0.3	0.6	x
sum	1.09	1.01	$0.99 + x$

- 1st place: $0.99 + x > 1.09$
  - $x > 0.1$
- 2nd place: $1.09 > 0.99 + x > 1.01$
  - $0.1 > x > 0.02$
- 3rd place: $0.92 + x < 1.01$
  - $0 < x < 0.02$

Q2

- Let the static quality scores for Doc1, Doc2 and Doc3 of Tutorial 3 be 0.25, 0.5 and 1, respectively. Sketch the postings for impact ordering (champion list) when each postings list is ordered by the sum of the *ntc* values from Tutorial 3 and the static quality scores (net score).

A2 from Tutorial 3

- Find document vectors: (i) nnn (ii) ntc

nnn(doc)	Doc1	Doc2	Doc3
car	22*1*1=22	4*1*1=4	24*1*1=24
auto	3*1*1=3	33*1*1=33	0*1*1=0
insurance	0*1*1=0	33*1*1=33	29*1*1=29
best	14*1*1=14	0*1*1=0	17*1*1=17

ntc(doc)	Doc1	Doc2	Doc3
car	(22*1.65=36.3)/42.4=0.86	(4*1.65=6.6)/87.3=0.08	(24*1.65=39.6)/66.5=0.60
auto	(3*2.08=6.24)/42.4=0.15	(33*2.08=68.64)/87.3=0.79	0*2.08=0
insurance	0*1.62=0	(33*1.62=53.46)/87.3=0.61	(29*1.62=46.98)/66.5=0.71
best	(14*1.5=21)/42.4=0.5	0*1.5=0	(17*1.5=25.5)/66.5=0.38
Doc length	42.4 ($\sqrt{36.3^2+6.24^2+0+21^2}$)	87.3	66.5

Do try this at home

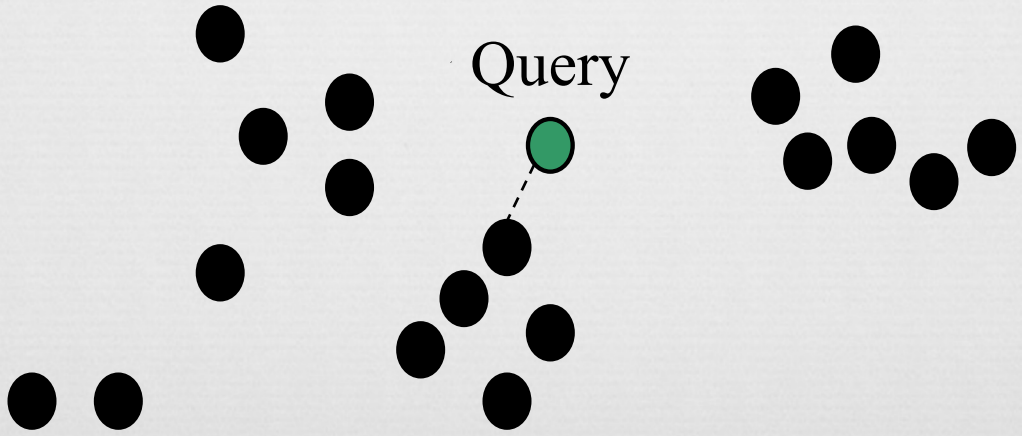
A2

ntc(doc)	Doc1	Doc2	Doc3
static quality score	0.25	0.5	1
car	0.86	0.08	0.60
auto	0.15	0.79	0
insurance	0	0.61	0.71
best	0.5	0	0.38

- Car: Doc3 (1.6), Doc1 (1.11), Doc2 (0.58)
- Auto: Doc2 (1.29), Doc3 (1), Doc1 (0.4)
- Insurance: Doc3 (1.71), Doc2 (1.11), Doc1 (0.25)
- Best: Doc3 (1.38), Doc1 (0.75), Doc2 (0.5)

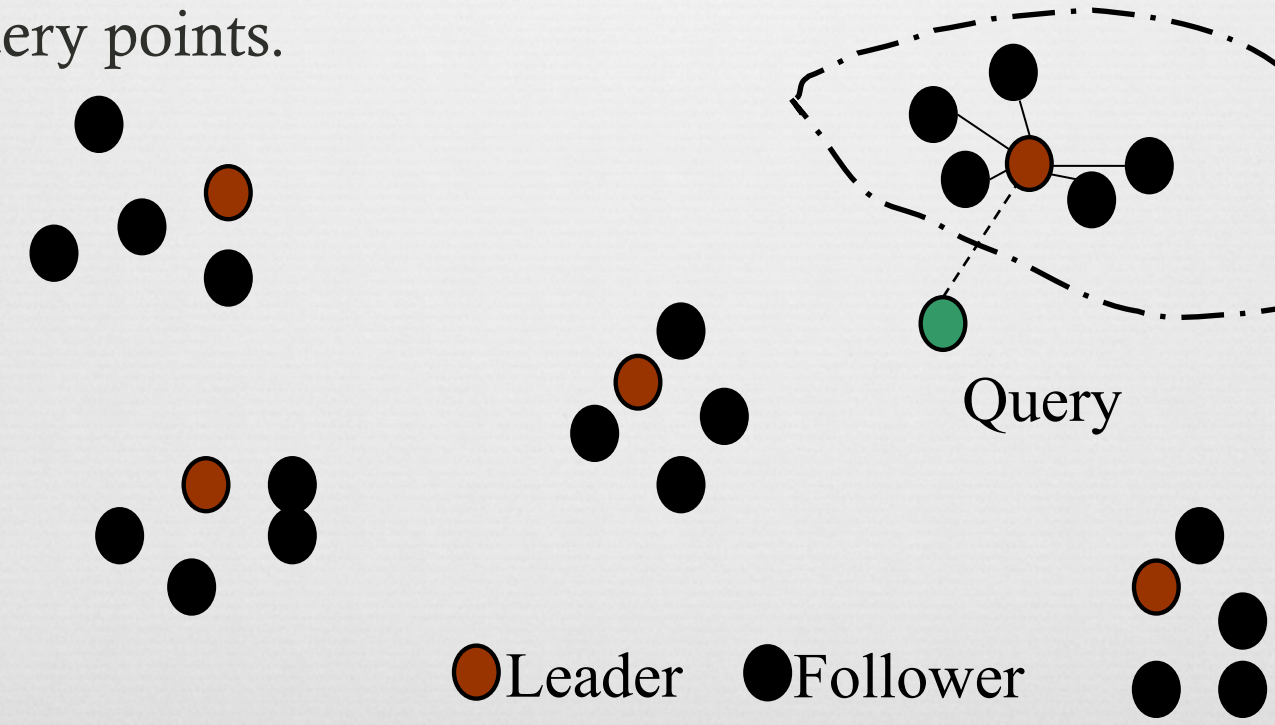
Q3

- The nearest-neighbor problem in the plane is the following:
  - given a set of N data points on the plane (i.e., 2-dimensional space), we preprocess them into some data structure such that, given a query point Q, we seek the point in N that is closest to Q in Euclidean distance.



Q3

- Clearly cluster pruning can be used as an approach to the nearest-neighbor problem in the plane, if we wished to avoid computing the distance from Q to every one of the query points.

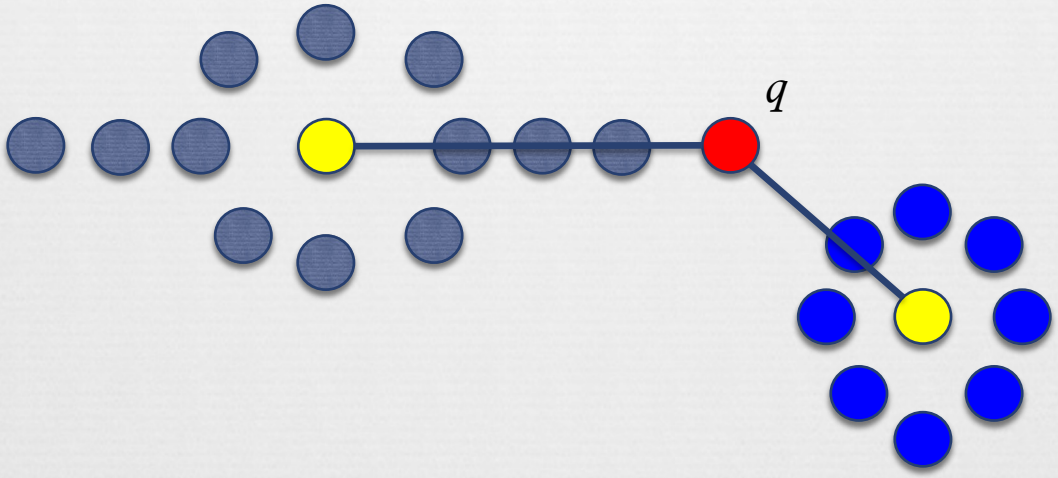


Q3

- Devise a simple example on the plane so that with two leaders, the answer returned by cluster pruning is incorrect (it is not the data point closest to q).

Do try this at home

A3



Key concepts (1st half)

- Boolean model
- Text pre-processing
- Tf-idf
- Vector space model

Tutorial 5

SC4021/CE4034/CZ4034

Q1

- Suppose you travel at speed P Km/h for distance x Km and speed R Km/h for another x Km. The total distance of the whole journey is $2x$ Km.
- Compute the average speed (which is the harmonic mean or F_1-measure of the values of P and R) for the whole journey of $2x$.
- Speed = distance / time

Do try this at home

A1

- Speed = distance / time

$$\frac{2x}{\frac{x}{P}+\frac{x}{R}}=\frac{2PR}{P+R}$$

Q2

- The balanced F measure (a.k.a. F1) is defined as the harmonic mean of precision and recall. What is the advantage of using the harmonic mean rather than "averaging" (using the arithmetic mean)?
- Hint:
  - P=0.1, R=0.9
  - P=0.5, R=0.5

Do try this at home

A2

- P=0.1, R=0.9
  - Arithmetic mean: (0.1+0.9)/2=0.5
  - Harmonic mean: 2*0.1*0.9/1=0.18
- P=0.5, R=0.5
  - Arithmetic mean: (0.5+0.5)/2=0.5
  - Harmonic mean: 2*0.5*0.5/1=0.5
- Very roughly speaking,
  - Arithmetic mean is closer to max
  - Harmonic mean is closer to min (more robust)

Q3

- Consider the following retrieval task where there are a total of 20 documents in the corpus and 8 documents are relevant to a query (the remaining 12 are irrelevant). The following left-to-right sequence denotes whether each successively retrieved document is relevant (1) or not (0):
- 00100 10000 11111 00001

Q3

- Using manual computation or a spreadsheet software or scripts/code (e.g. matlab),
  - Plot the Precision P versus N (number of documents retrieved) curve, clearly labeling P.
  - Plot the Recall R versus N curve on the same graph as part (a), clearly labeling R.
  - Plot the F1-Measure versus N curve on the same graph as part (a), clearly labeling F.
  - Plot the Arithmetic Mean (M) versus N curve on the same graph as part (a) clearly labeling M.

Unranked retrieval evaluation: Precision and Recall

- **Precision**: fraction of retrieved docs that are relevant = P(relevant|retrieved)
- **Recall**: fraction of relevant docs that are retrieved = P(retrieved|relevant)

	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn

- Precision P = tp/(tp + fp)
- Recall R = tp/(tp + fn)

F_1 & Arithmetic Mean

- F_1-Measure

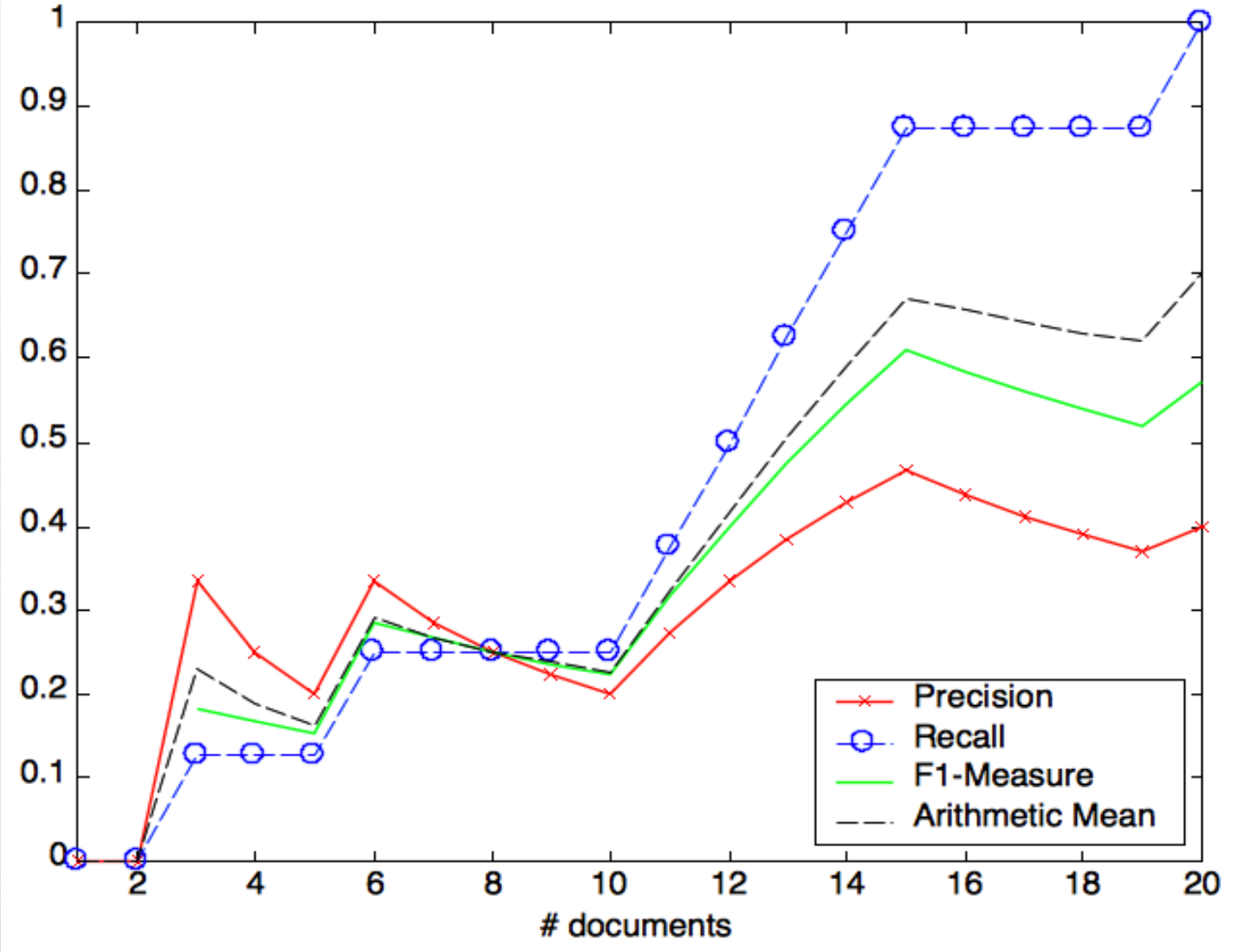
$$F_1 - measure = \frac{2PR}{P+R}$$

- Arithmetic Mean

$$AM = \frac{P+R}{2}$$

Do try this at home

A3: 00100 10000 11111 00001 (N_R=8)

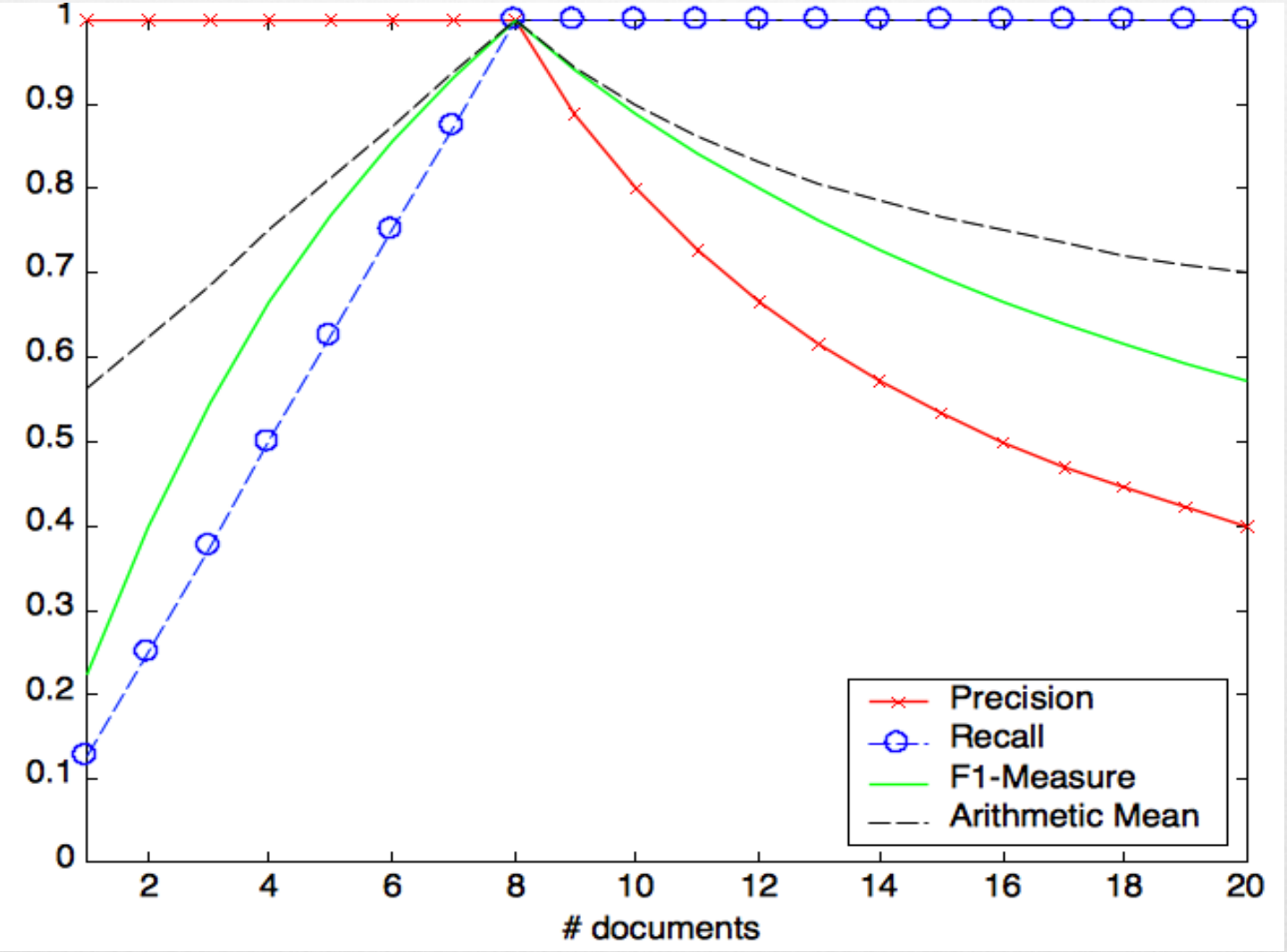


Q4

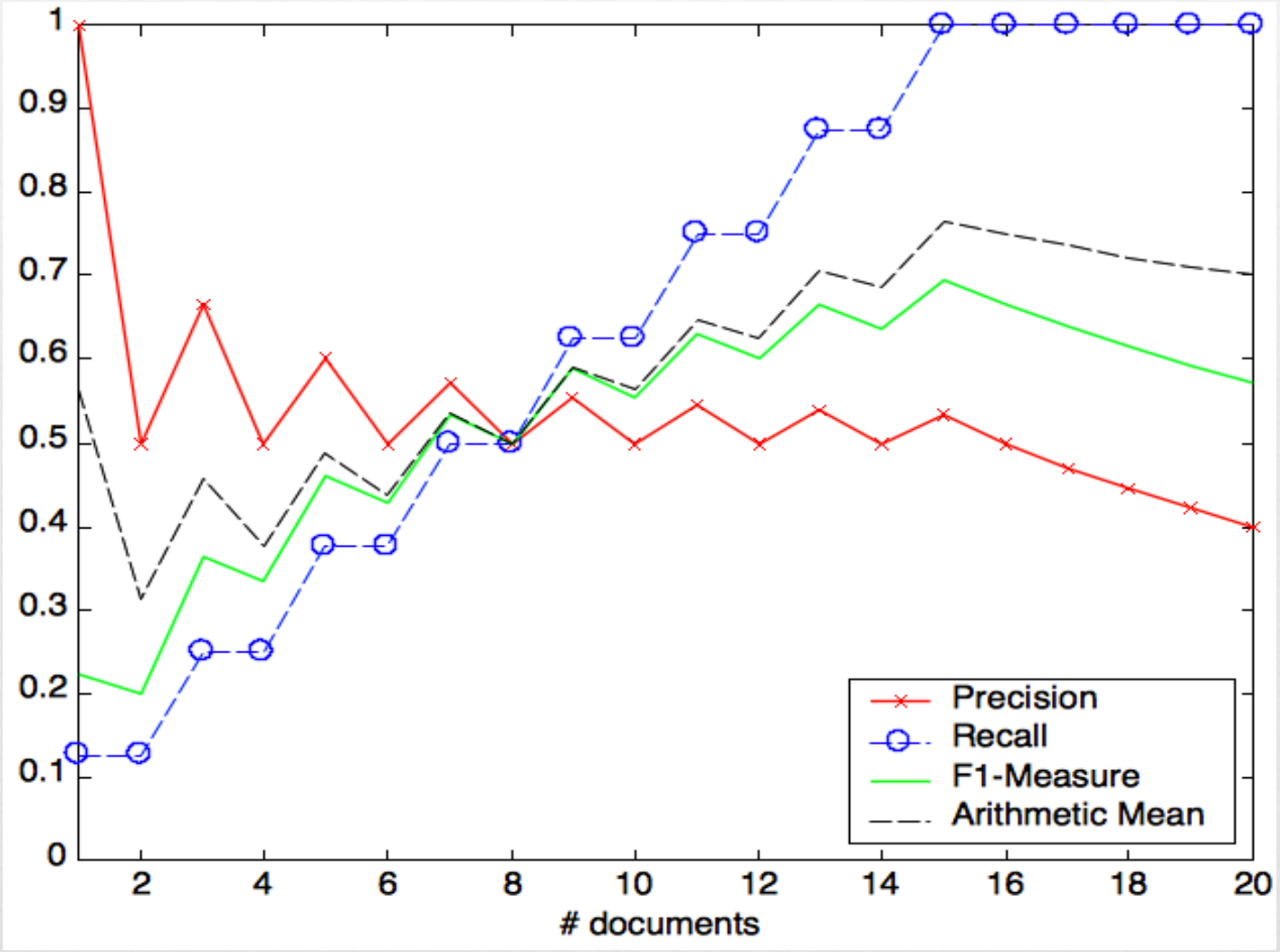
- Repeat Q3(a) to (d) for the following sequence of documents:
  a) 11111 11100 00000 00000
  b) 10101 01010 10101 00000
  c) 00000 00000 00111 11111
  d) 11111 11111 11111 00000
- For parts (a)-(c), assume $N_R=8$ relevant documents, and for part (d), assume $N_R=30$ relevant documents.

Do try this at home

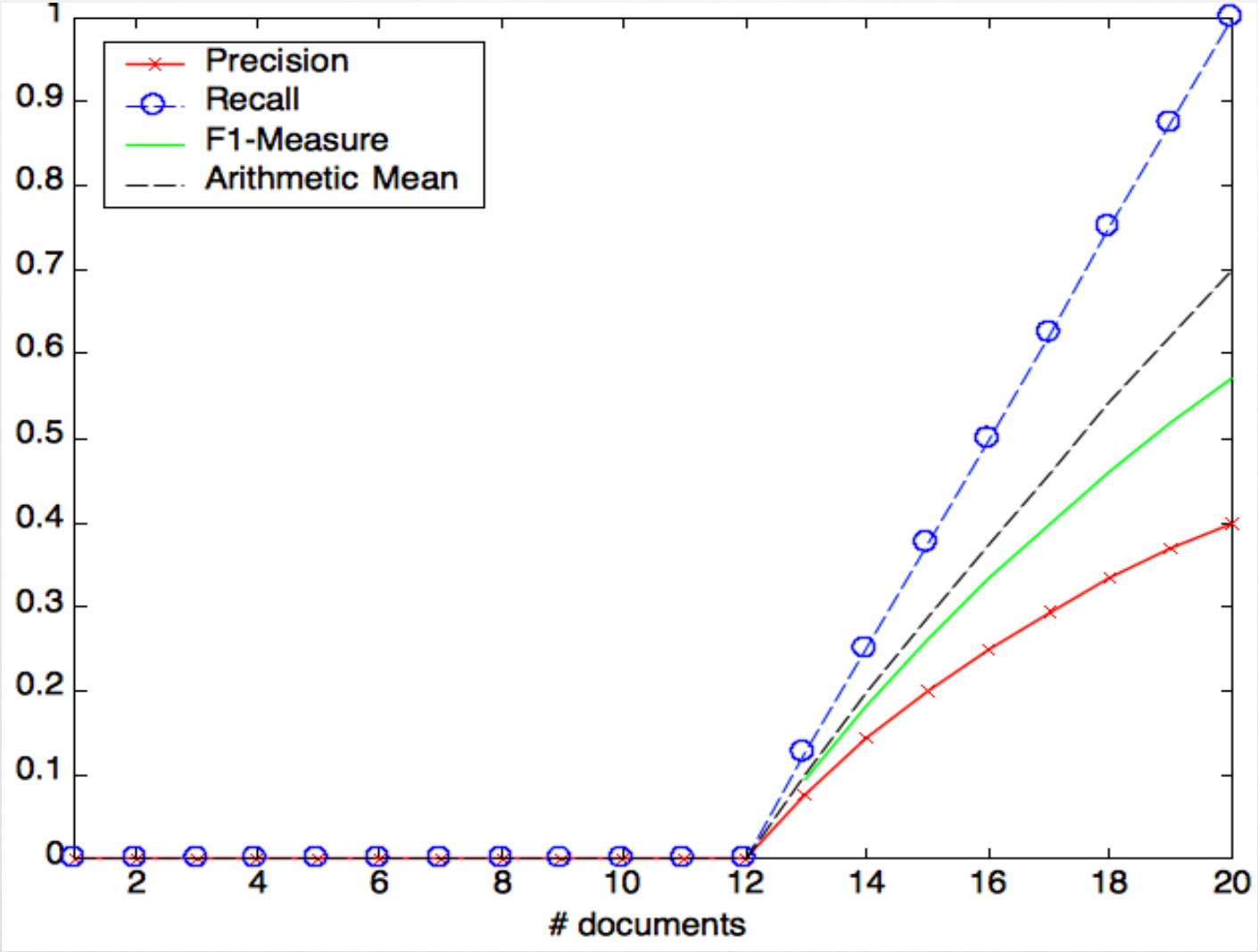
A4.a: 11111 11100 00000 00000 (N_R=8)



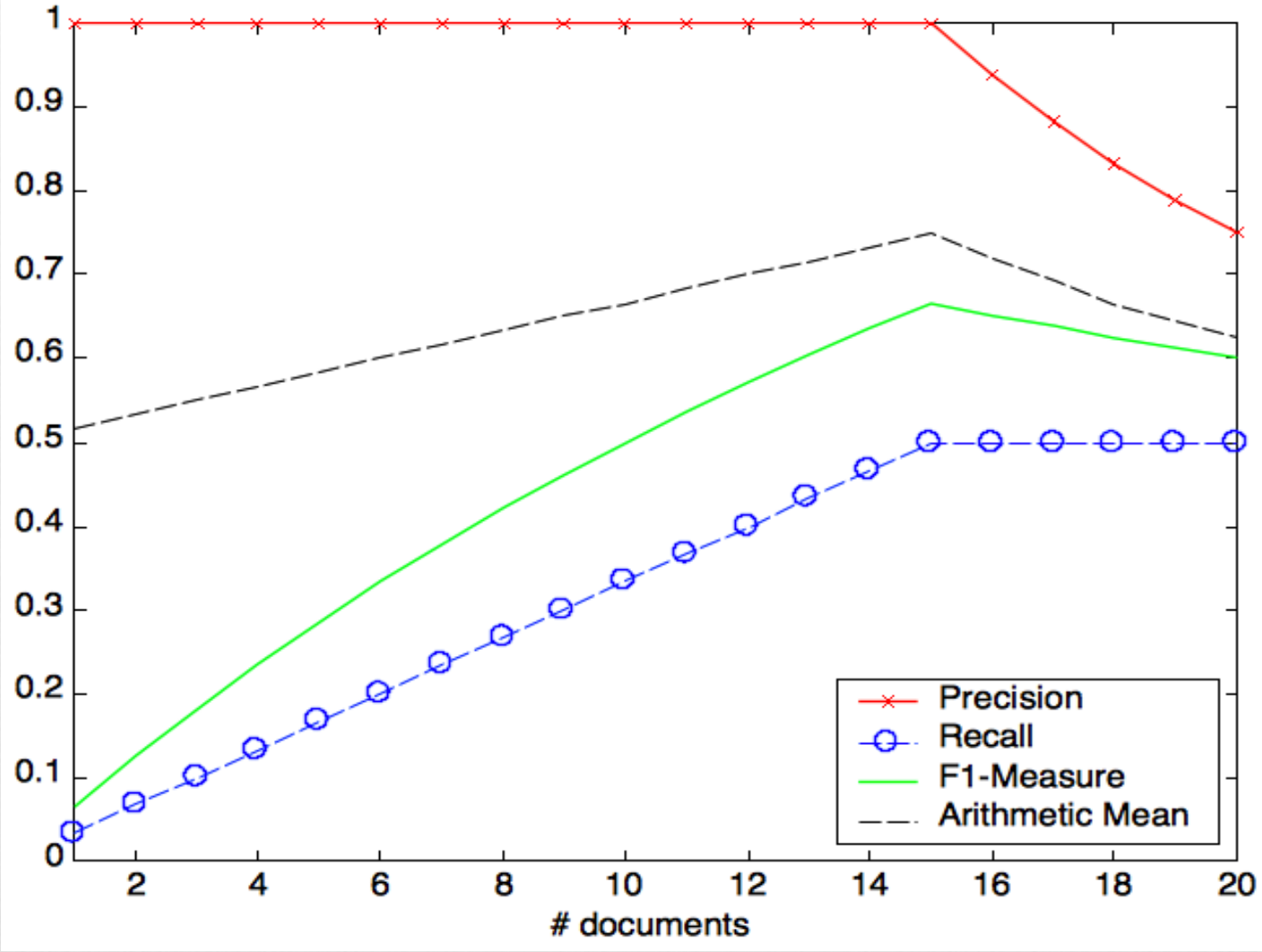
A4.b: 10101 01010 10101 00000 (N_R=8)



A4.c: 00000 00000 00111 11111 (N_R=8)



A4.d: 11111 11111 11111 00000 (N_R=30)

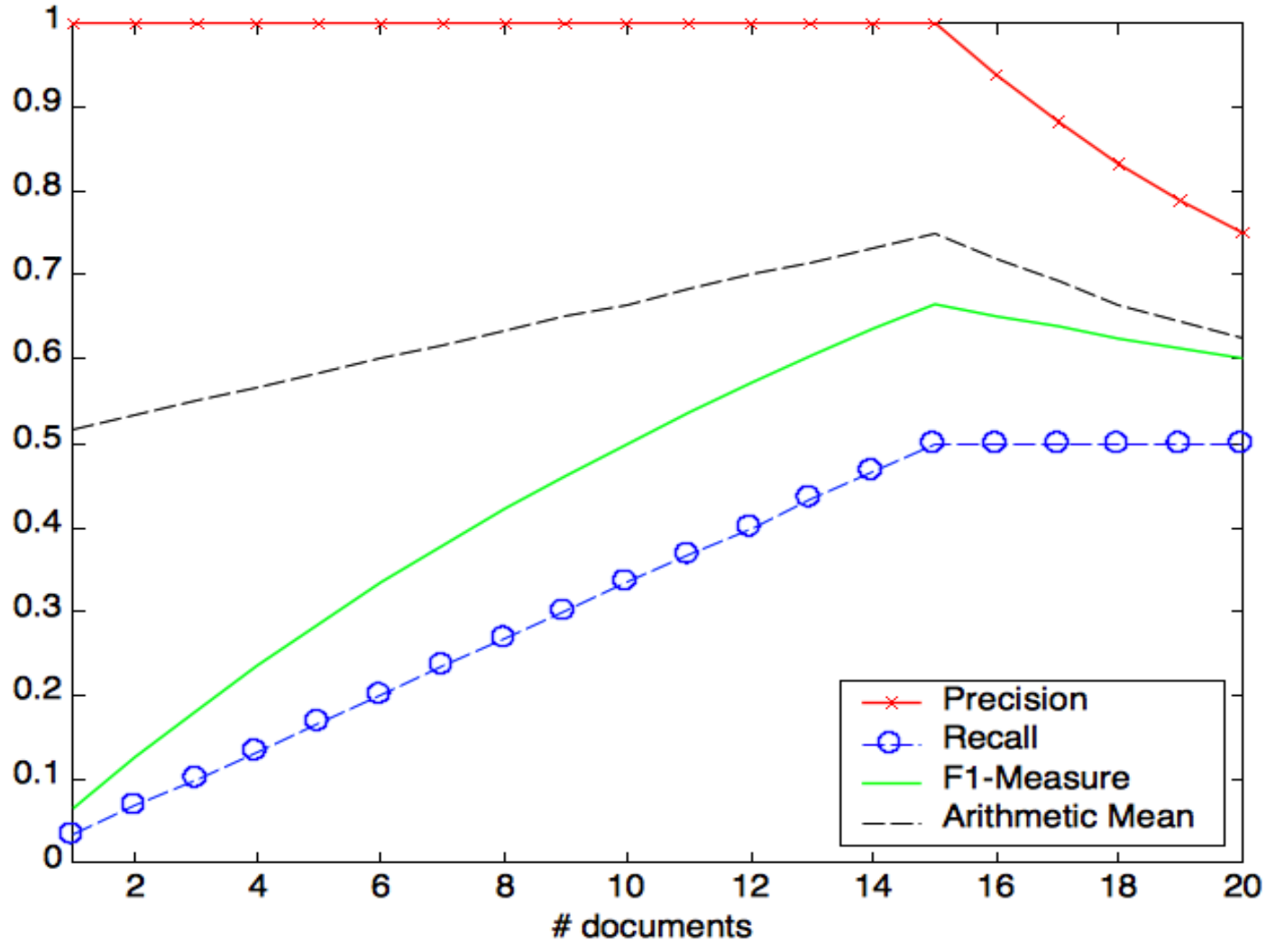


Q5

- Based on your plots in Q3 and Q4, state whether the following statements are true or false?
  a) The arithmetic mean curve is always sandwiched between the P and R curve.
  b) The harmonic mean (F1-measure) curve is always sandwiched between the P and R curve.
  c) The arithmetic mean is always larger or equal to the harmonic mean (F-Measure).
  d) The P versus N curve always starts from 1.
  e) The R versus N curve always starts from 0.
  f) The R versus N curve always ends at 1.
  g) The P and R curve will always intersect @ N = R0 (total # of relevant docs in corpus)

Do try this at home

A5.a: The arithmetic mean curve is always sandwiched between the P and R curve

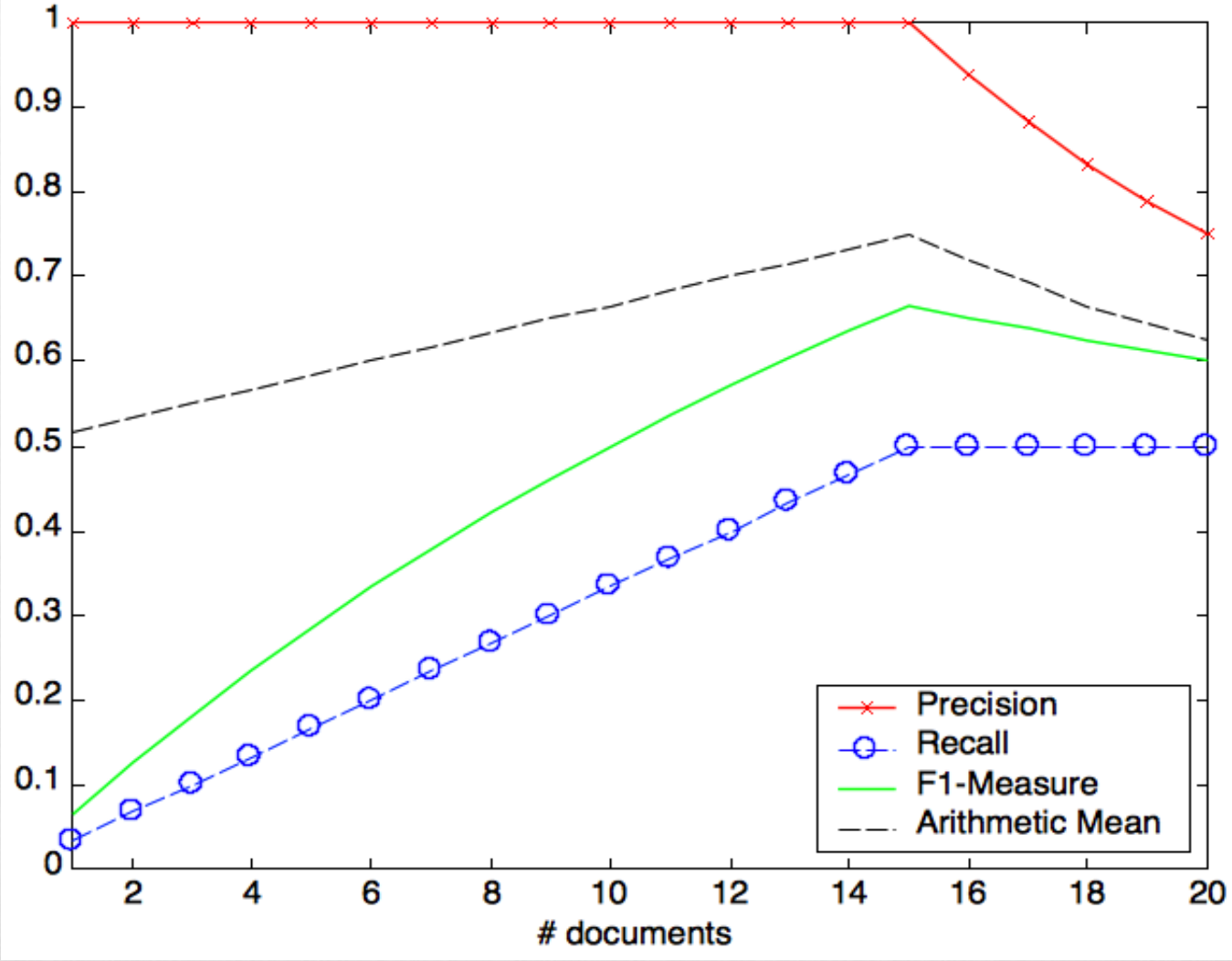


True

A5.a

- If $P > R$
  - $P = P/2 + P/2 > P/2 + R/2 > R/2 + R/2 = R$

A5.b: The harmonic mean (F1-measure) curve is always sandwiched between the P and R curve

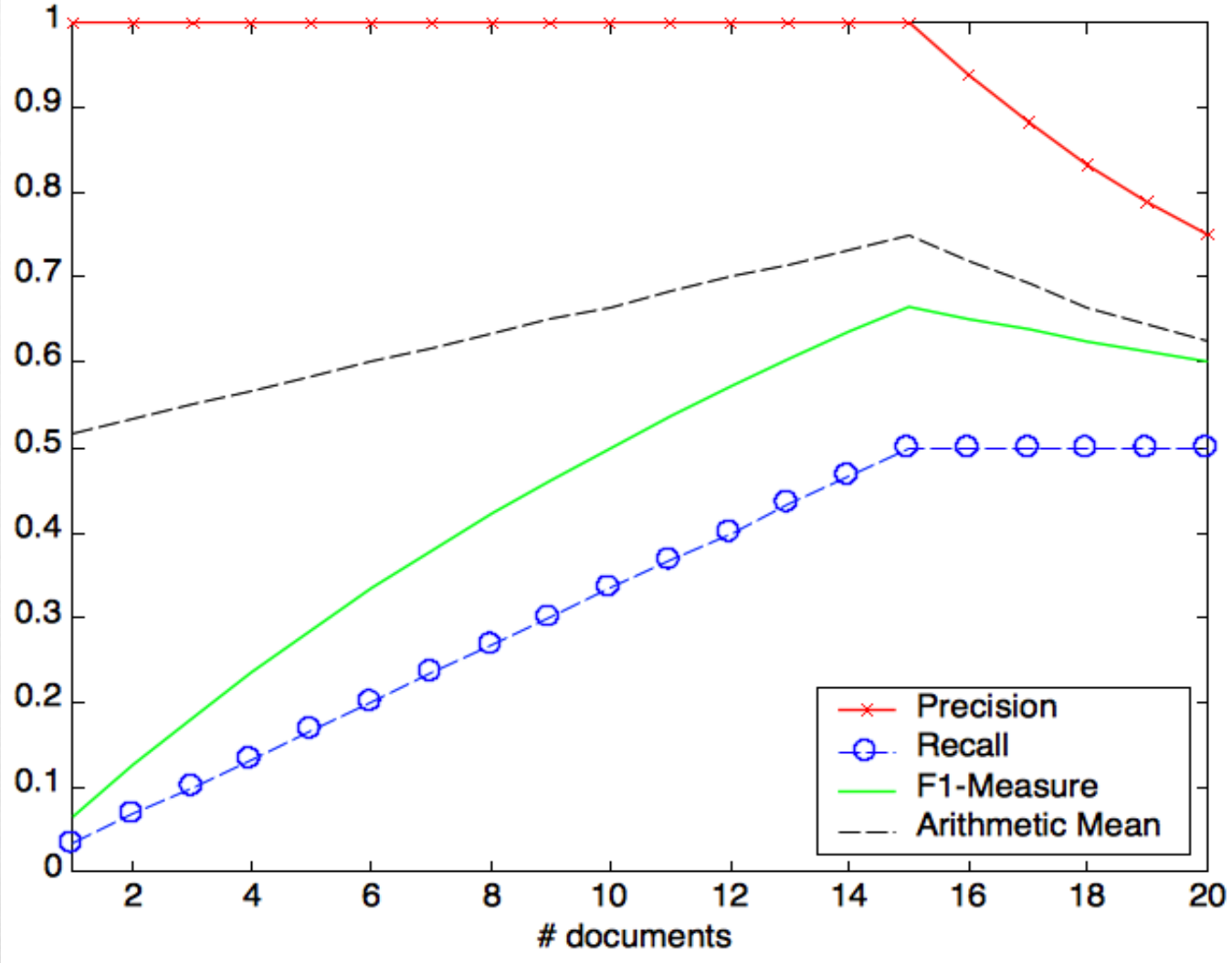


True

A5.b

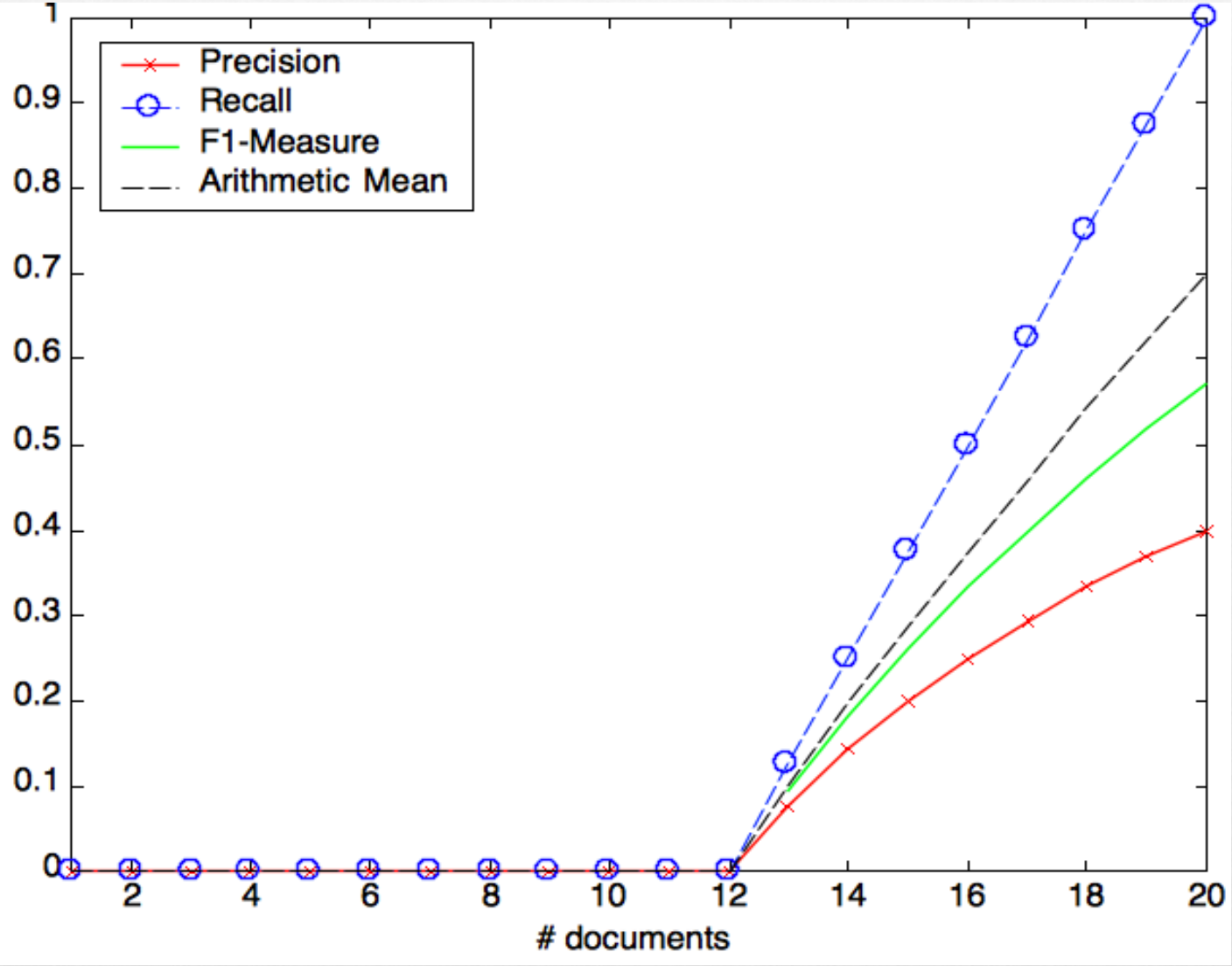
- If P > R
  - $P = P*2R/(R+R) > 2PR/(P+R) > 2PR/(P+P) = R$

A5.c: The arithmetic mean is always larger or equal to the harmonic mean (F-Measure).



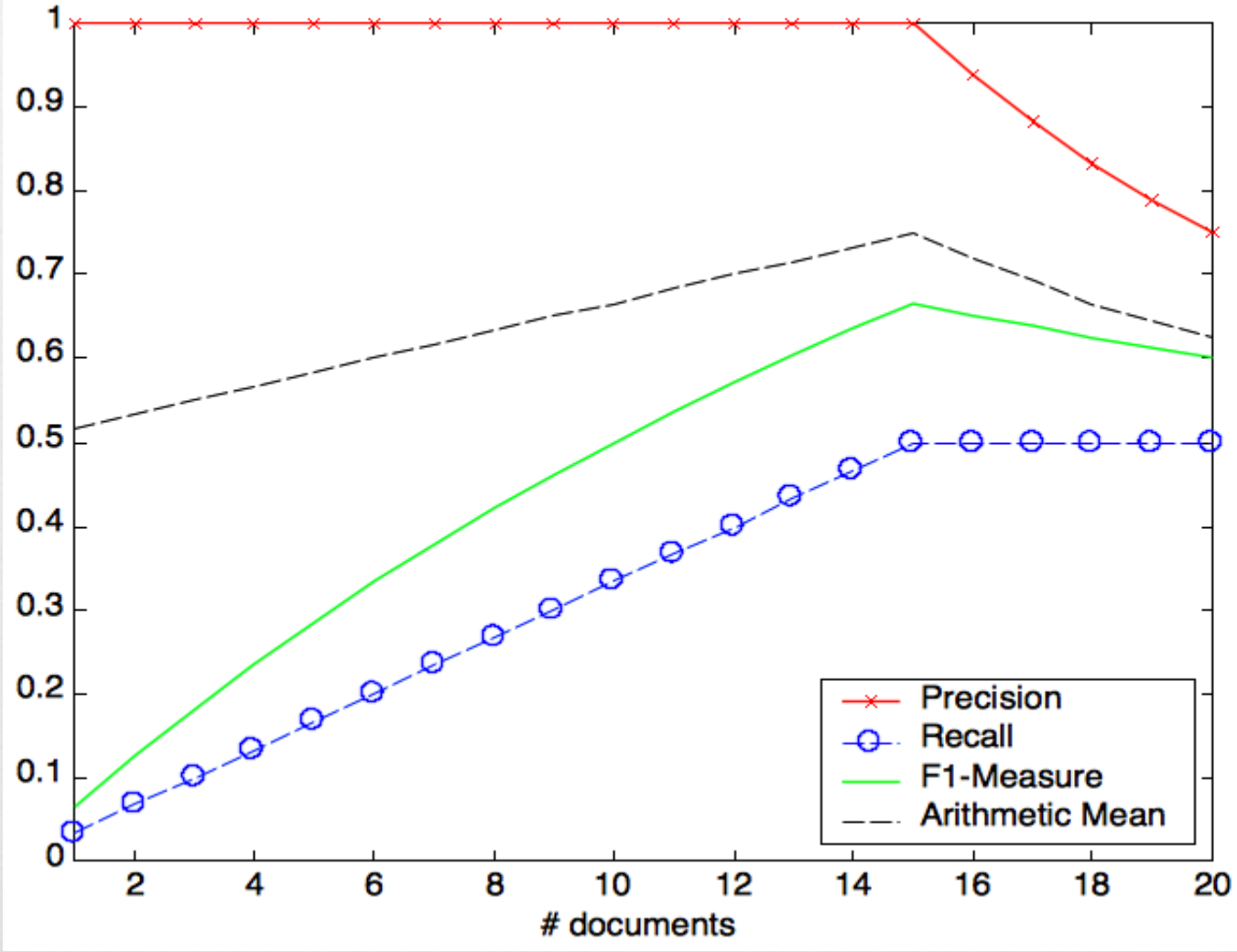
True

A5.d: The P versus N curve always starts from 1



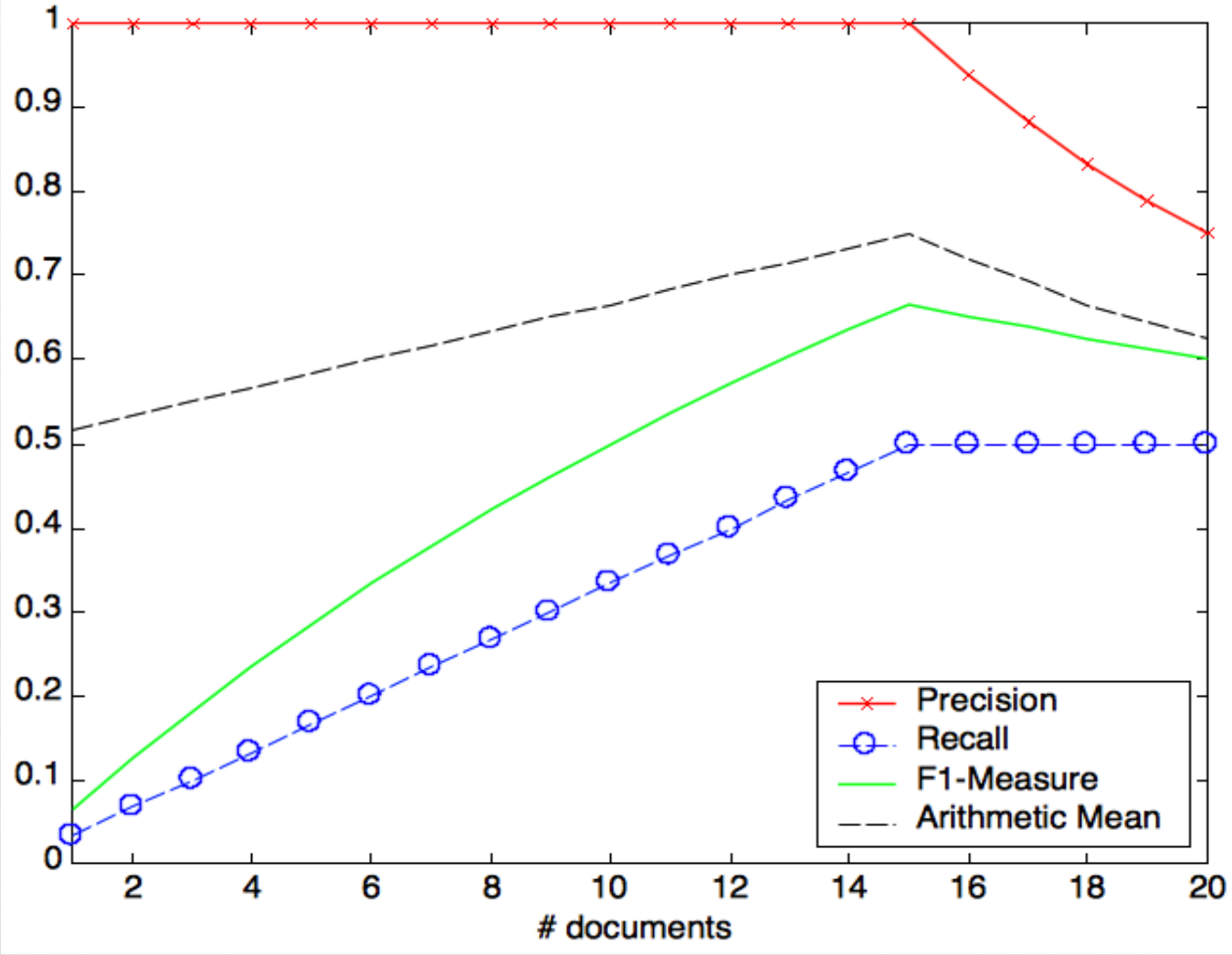
False

A5.e: The R versus N curve always starts from 0



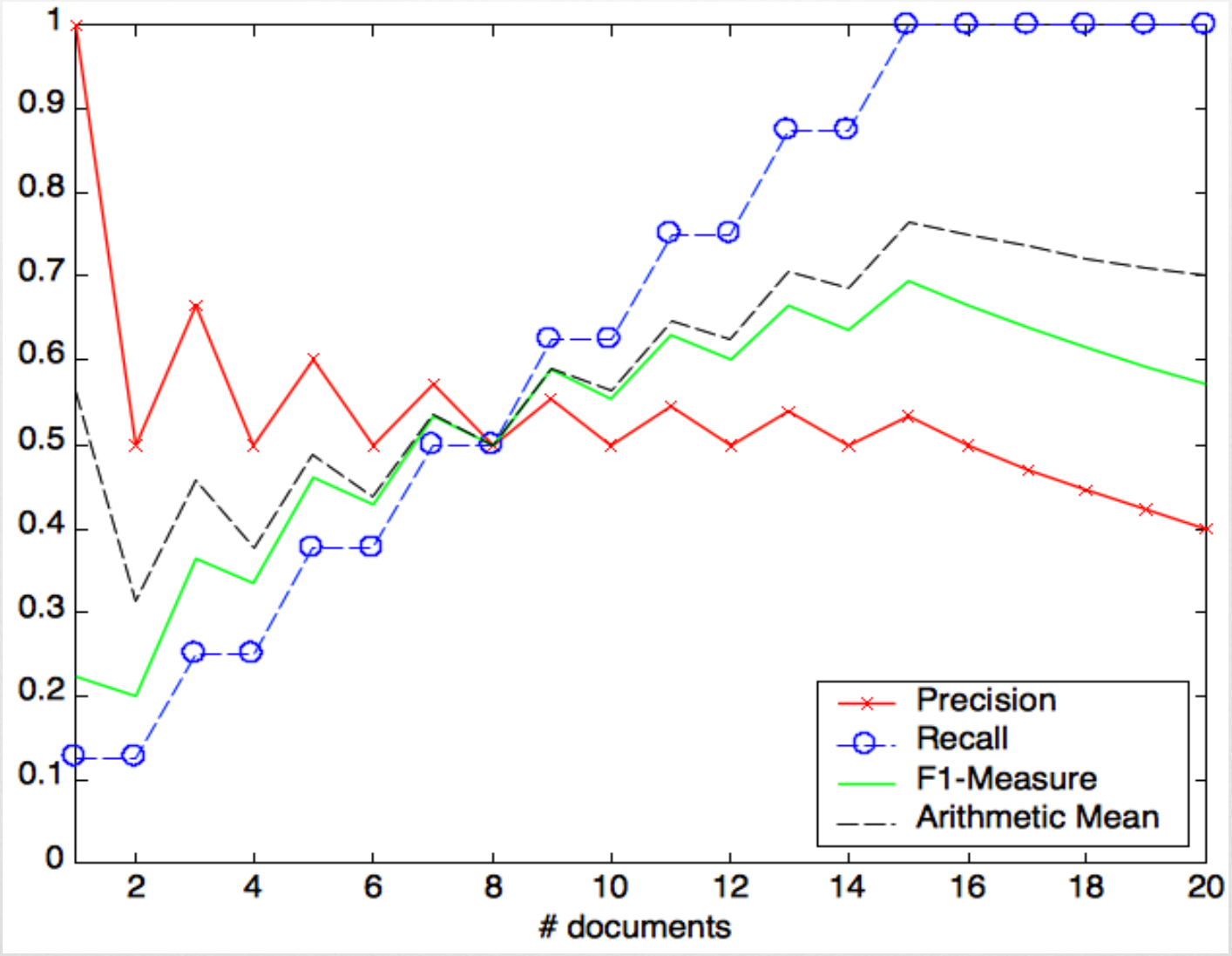
False

A5.f: The R versus N curve always ends at 1



False

A5.g: The P and R curve will always intersect @ N = R0 (total # of relevant docs in corpus)



True

A5.g

	Relevant	Nonrelevant
Retrieved	tp	fp
Not Retrieved	fn	tn

- Precision P = tp/(tp + fp)
- Recall R = tp/(tp + fn)

- P=tp/N, R=tp/R0
- when N=R0

Q6

- Given that N=# docs retrieved, N_R=# relevant docs retrieved, R0=# relevant docs in corpus
  a) What is the condition for the P and R curve to intersect?
  b) How many times will the P and R curve intersect at non-zero values?
  c) What is the value of the F_1-Measure at the intersection?

Do try this at home

A6

- Precision = #(relevant among retrieved) / #(retrieved)
  - N_R/N
- Recall = #(relevant among retrieved) / #(relevant)
  - $N_R/R0$
- a) Intersection condition: N = R0
- b) Once
- c) F_1-measure $= \frac{2PR}{P+R} = P = \frac{N_R}{N} = R = \frac{N_R}{R0}$

Q7

- Suppose you have a corpus of Web documents with a vocabulary of 5 words {thaksin, gst, thailand, fine, raise}
- Modify the initial query of {gst, raise} based on the 4 returned results below
- set of relevant documents
  - D_r = {[1 0 1 0 1], [1 1 1 0 1]}, and
- set of irrelevant documents
  - D_n = {[0 0 0 1 0], [0 1 0 1 1]}

Q7

- using Rocchio's method with the following parameters:
  - $\alpha=1, \beta=1, \gamma=0.1$
  - $\alpha=1, \beta=1, \gamma=0.5$
  - $\alpha=1, \beta=1, \gamma=1$

Initial query/results

- Initial query: *New space satellite applications*

+ 1. 0.539, 08/13/91, NASA Hasn't Scrapped Imaging Spectrometer
+ 2. 0.533, 07/09/91, NASA Scratches Environment Gear From Satellite Plan
3. 0.528, 04/04/90, Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes
4. 0.526, 09/09/91, A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget
5. 0.525, 07/24/90, Scientist Who Exposed Global Warming Proposes Satellites for Climate Research
6. 0.524, 08/22/90, Report Provides Support for the Critics Of Using Big Satellites to Study Climate
7. 0.516, 04/13/87, Arianespace Receives Satellite Launch Pact From Telesat Canada
+ 8. 0.509, 12/02/87, Telecommunications Tale of Two Companies

- User then marks relevant documents with "+".

Key concept: Centroid

- The centroid is the center of mass of a set of points
- Recall that we represent documents as points in a high-dimensional space
- Definition: Centroid

$$\vec{\mu}(C) = \frac{1}{|C|}\sum_{d \in C}\vec{d}$$

where C is a set of documents.

Rocchio Algorithm

- Used in practice:

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

- D_r = set of known relevant doc vectors
- D_{nr} = set of known irrelevant doc vectors
  - Different from C_r and C_{nr}
- q_m = modified query vector; q_0 = original query vector; α, β, γ: weights (hand-chosen or set empirically)
- New query moves toward relevant documents and away from irrelevant documents

Do try this at home

A7

$q_0 = [0\ 1\ 0\ 0\ 1]$

$D_r = \{[1\ 0\ 1\ 0\ 1], [1\ 1\ 1\ 0\ 1]\}$

$D_n = \{[0\ 0\ 0\ 1\ 0], [0\ 1\ 0\ 1\ 1]\}$

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

A

B

- A

(1+1)/2	(0+1)/2	(1+1)/2	(0+0)/2	(1+1)/2	1	0.5	1	0	1

- B

(0+0)/2	(0+1)/2	(0+0)/2	(1+1)/2	(0+1)/2	0	0.5	0	1	0.5

- α=1, β=1, γ=0.1

0 + 1+ 0	1 + 0.5 – 0.05	0 + 1 - 0	0 + 0 – 0.1	1 + 1 – 0.05	1	1.45	1	-0.1	1.95

$\alpha q_0 \quad \beta_A \quad -\gamma B$

- α=1, β=1, γ=0.5

0 + 1+ 0	1 + 0.5 – 0.25	0 + 1 - 0	0 + 0 – 0.5	1 + 1 – 0.25	1	1.25	1	-0.5	1.75

- α=1, β=1, γ=1

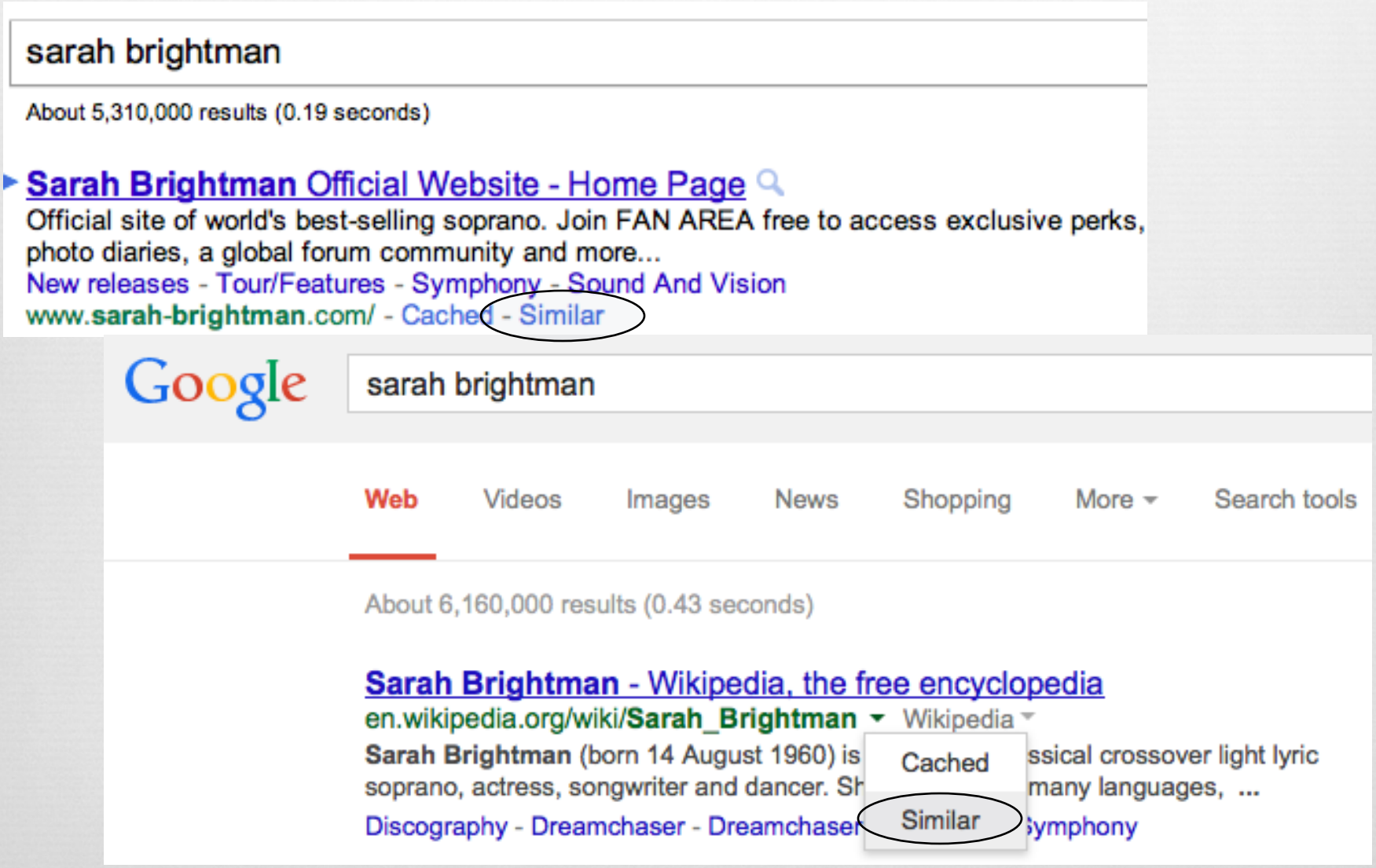
0 + 1+ 0	1 + 0.5 – 0.5	0 + 1 - 0	0 + 0 – 1	1 + 1 – 0.5	1	1	1	-1	1. 5

Q8

- In Rocchio's algorithm, what weight setting for α, β, γ does a "Find pages like this one" search correspond to?

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$$

Similar pages



Do try this at home

A8

- α=0, β=1, γ=0

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$$

Q9

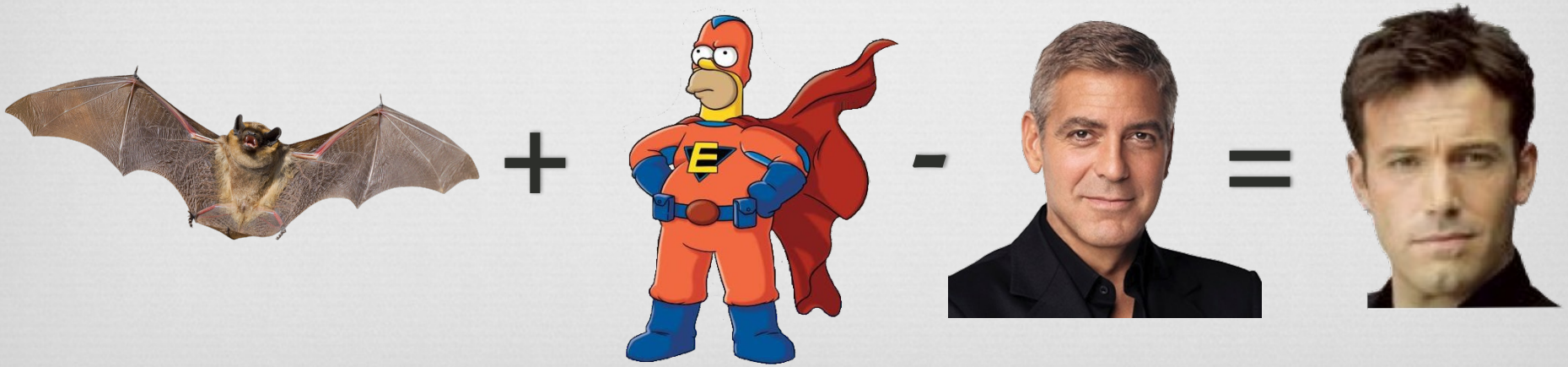
- Why is positive feedback likely to be more useful than negative feedback to an IR system?
- Why might only using one non-relevant document be more effective than using several?

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

Do try this at home

A9

- Relevant documents are assumed to be in coherent ‘clusters’
- Are non-relevant documents assumed so, too?
- If only 1 ‘bad’ document is specified, it is unlikely to be similar to the ‘good’ cluster



Tutorial 6

SC4021/CE4034/CZ4034

Q1

- Consider the following 10 class conditioned word probabilities (c_0=non-spam, c_1=spam):

Word w_i	brand	huge	hottest	incredible	million	new	offers	pay	save	family
$p(w_i \mid c_0)$	0.10	0.20	0.30	0.05	0.05	0.10	0.20	0.02	0.03	0.40
$p(w_i \mid c_1)$	0.98	0.92	0.91	0.99	0.98	0.99	0.93	0.99	0.99	0.02

- For each of the 3 email snippets below, ignoring case, punctuations, and words beyond the 10 known vocabulary words, compute the class conditioned document probabilities for each of the 3 documents, namely $P(d_1 \mid c_0)$, $P(d_2 \mid c_0)$, $P(d_3 \mid c_0)$, $P(d_1 \mid c_1)$, $P(d_2 \mid c_1)$, and $P(d_3 \mid c_1)$, using the Naïve Bayes model.

Q1

- d_1 : OEM software - throw packing case, leave CD, use electronic manuals. **Pay** for software only and **save** 75-90%! Find **incredible** discounts! See our special **offers**!
- d_2: Our **Hottest** pick this year! **Brand new** issue Cana Petroleum! VERY tightly held, in a booming business sector, with a **huge** publicity campaign starting up, Cana Petroleum (CNPM) is set to bring all our readers **huge** gains. We advise you to get in on this one and ride it to the top!
- d_3: Dear friend, How is your **family**? hope all of you are fine, if so splendid. Yaw Osafo-Maafo is my name and former Ghanaian minister of finance. Although I was sacked by President John Kufuor on 28 April 2006 for the fact I signed 29 **million** book publication contract with Macmillan Education without reference to the Public Procurement Board and without Parliamentary approval.

Naïve Bayes Classifier

$$d = \langle x_1, x_2, \ldots, x_n \rangle$$

$$c_{MAP} = \underset{c_j \in C}{\operatorname{argmax}}\, P(c_j \mid x_1, x_2, \ldots, x_n)$$

The probability of a document *d* being in class *c*.

$$= \underset{c_j \in C}{\operatorname{argmax}}\, P(x_1, x_2, \ldots, x_n \mid c_j) P(c_j)$$

Bayes' Rule

$$= \underset{c_j \in C}{\operatorname{argmax}}\, P(x_1 \mid c_j) P(x_2 \mid c_j) \ldots P(x_n \mid c_j) P(c_j)$$

Conditional Dependence Assumption

$$\hat{P}(c_j) = \frac{N(C = c_j)}{N} \qquad \hat{P}(x_i \mid c_j) = \frac{N(X_i = x_i, C = c_j) + 1}{N(C = c_j) + k}$$

Do try this at home

A1

$$p(d_j \mid c_k) \propto \prod_{i=1}^{t} p(w_i \mid c_k)^{f(w_i, d_j)}$$

- where $f(w_i, d_j)$ = frequency of word w_i in document d_j

Word w_i	brand	huge	hottest	incredible	million	new	offers	pay	save	family
$p(w_i \mid c_0)$	0.10	0.20	0.30	0.05	0.05	0.10	0.20	0.02	0.03	0.40
$p(w_i \mid c_1)$	0.98	0.92	0.91	0.99	0.98	0.99	0.93	0.99	0.99	0.02

we ignore case, punctuations, and words beyond the 10 known vocabulary words

A1

d_2: Our **Hottest** pick this year! **Brand new** issue Cana Petroleum! VERY tightly held, in a booming business sector, with a **huge** publicity campaign starting up, Cana Petroleum (CNPM) is set to bring all our readers **huge** gains. We advise you to get in on this one and ride it to the top!

$$p(d_j \mid c_k) \propto \prod_{i=1}^{t} p(w_i \mid c_k)^{f(w_i, d_j)}$$

$$p(d_2 \mid c_0) = p(\text{hottest} \mid c_0) * p(\text{brand} \mid c_0) * p(\text{new} \mid c_0) * p(\text{huge} \mid c_0)^2$$

Word w_i	brand	huge	hottest	incredible	million	new	offers	pay	save	family
$p(w_i \mid c_0)$	0.10	0.20	0.30	0.05	0.05	0.10	0.20	0.02	0.03	0.40
$p(w_i \mid c_1)$	0.98	0.92	0.91	0.99	0.98	0.99	0.93	0.99	0.99	0.02

A1

$$p(d_1 \mid c_0) \propto 0.05^1 \times 0.20^1 \times 0.02^1 \times 0.03^1 = 6 \times 10^{-6}$$
$$p(d_1 \mid c_1) \propto 0.99 \times 0.93 \times 0.99 \times 0.99 \approx 9.02 \times 10^{-1}$$

$$p(d_2 \mid c_0) \propto 0.10 \times 0.20^2 \times 0.30 \times 0.10 = 1.2 \times 10^{-4}$$
$$p(d_2 \mid c_1) \propto 0.98 \times 0.92^2 \times 0.91 \times 0.99 \approx 7.47 \times 10^{-1}$$

$$p(d_3 \mid c_0) \propto 0.05 \times 0.40 = 2 \times 10^{-2}$$
$$p(d_3 \mid c_1) \propto 0.98 \times 0.02 = 1.96 \times 10^{-2}$$

Q2

- Compute the posterior probabilities of each document in Q1, namely $P(c_0 | d_1)$, $P(c_1 | d_1)$, $P(c_0 | d_2)$, $P(c_1 | d_2)$, $P(c_0 | d_3)$, $P(c_1 | d_3)$, assuming that 80% of all emails received are spam, and finally decide whether each document is spam or non-spam.

$$p(c_k | d_j) \propto p(d_j | c_k) p(c_k)$$

Do try this at home

A2

- "assuming that 80% of all emails received are spam" means that prior class probability $P(c_1) = 0.8$
- hence, $P(c_0) = 1 - P(c_1) = 0.2$

A2

$P(c_0 \mid d_1) \approx P(d_1 \mid c_0) \times P(c_0) = 6 \times 10^{-6} \times 0.2 = 1.2 \times 10^{-6}$

$\mathbf{P(c_1 \mid d_1) \approx P(d_1 \mid c_1) \times P(c_1) = 0.902 \times 0.8 = 0.72}$

} d_1 is spam

$P(c_0 \mid d_2) \approx P(d_2 \mid c_0) \times P(c_0) = 1.2 \times 10^{-4} \times 0.2 = 2.4 \times 10^{-5}$

$\mathbf{P(c_1 \mid d_2) \approx P(d_2 \mid c_1) \times P(c_1) = 0.747 \times 0.8 = 0.6}$

} d_2 is spam

$P(c_0 \mid d_3) \approx P(d_3 \mid c_0) \times P(c_0) = 0.02 \times 0.2 = 0.004$

$\mathbf{P(c_1 \mid d_3) \approx P(d_3 \mid c_1) \times P(c_1) = 0.0196 \times 0.8 = 0.016}$

} d_3 is spam

Q3

- Build a Naïve Bayes classifier using words as features for the training set in Table 2 and use the classifier to classify the test set in the table.

	docID	words in document	in $c = China$?
training set	1	Taipei Taiwan	yes
	2	Macao Taiwan Shanghai	yes
	3	Japan Sapporo	no
	4	Sapporo Osaka Taiwan	no
test set	5	Taiwan Taiwan Sapporo	?

Bayes probability

- Prior probability: $P(c_k)$
  - Probability of expecting class c_k before taking in account any evidence
- Likelihood: $P(doc_j|c_k) = \Pi_{i=1}^{n} P(x_i|c_k)$
  - True only because we make the **"naïve" conditional independence assumptions**
- Posterior probability: $P(c_k|doc_j) \propto P(c_k)P(doc_j|c_k)$

Naïve Bayes: Learning

$$P(c_K) = \frac{N(c_k)}{N}$$

Number of documents belonging to class c_k

Total number of documents

$$P(x_i|c_K) = \frac{N(x_i, c_k) + 1}{N(X, c_k) + |Vocabulary|}$$

Number of occurrence of term x_i in docs of class c_k

Number of terms appearing in docs of class c_k

MAP classifier

- MAP is maximum a posteriori
  - **Detect the class that maximize our posteriori probability**

$$\underset{c_k}{\operatorname{argmax}}\, P(c_k|doc_j) \propto \underset{c_k}{\operatorname{argmax}}\, P(c_k)\Pi_{i=1}^{n} P(x_i|c_k)$$

- We calculate P for all classes c_k and take the max

Do try this at home

A3

- Prior probability:
  - p(China) = 2/4, p(~China) = 2/4

	docID	words in document	in $c = China$?
training set	1	Taipei Taiwan	yes
	2	Macao Taiwan Shanghai	yes
	3	Japan Sapporo	no
	4	Sapporo Osaka Taiwan	no
test set	5	Taiwan Taiwan Sapporo	?

A3 (Learning)

Doc Id	Terms		
1	Taipei	Taiwan	
2	Macao	Taiwan	Shanghai
3	Japan	Sapporo	
4	Sapporo	Osaka	Taiwan

Vocabulary = {Taipei, Taiwan, Macao, Shanghai, Japan, Sapporo, Osaka}
|Vocabulary| = 7

Doc class	#Terms
yes	5
no	5

A3 (Learning)

P(Taipei|yes) = (1+1)/(5+7) = 2/12

P(Taipei|no) = (0+1)/(5+7) = 1/12

P(Taiwan|yes) = (2+1)/(5+7) = 3/12

P(Taiwan|no) = (1+1)/(5+7) = 2/12

$$P(x_i|c_k) = \frac{N(x_i, c_k) + 1}{N(X, c_k) + |Vocabilary|}$$

P(Sapporo|yes) = (0+1)/(5+7) = 1/12

P(Sapporo|no) = (2+1)/(5+7) = 3/12

A3 (Classifying)

Doc Id	Terms		
5	Taiwan	Taiwan	Sapporo

$$P(c_k|d_5) \propto P(c_k)\Pi_{i=1}^{n} P(x_i|c_k)$$

$$P(\text{yes}|d_5) = \frac{2}{4} \times \left(\frac{3}{12}\right)^2 \times \frac{1}{12} \approx 2.60 \times 10^{-3}$$

$$P(\text{no}|d_5) = \frac{2}{4} \times \left(\frac{2}{12}\right)^2 \times \frac{3}{12} \approx 3.47 \times 10^{-3}$$

Answer: d_5 belongs to the class ‘no’

Q4

- 48% of the 71 PSC (Public Service Commission) scholars lived in HDB. 77% of Singapore's population (estimated to be 6 million) lives in HDB. Use the Chi-Square test at significance level p=0.001 (table below) to test whether the distribution of PSC scholars is a reflection of the underlying population distribution.

p	χ^2 critical value
0.1	2.71
0.05	3.84
0.01	6.63
0.005	7.88
0.001	10.83

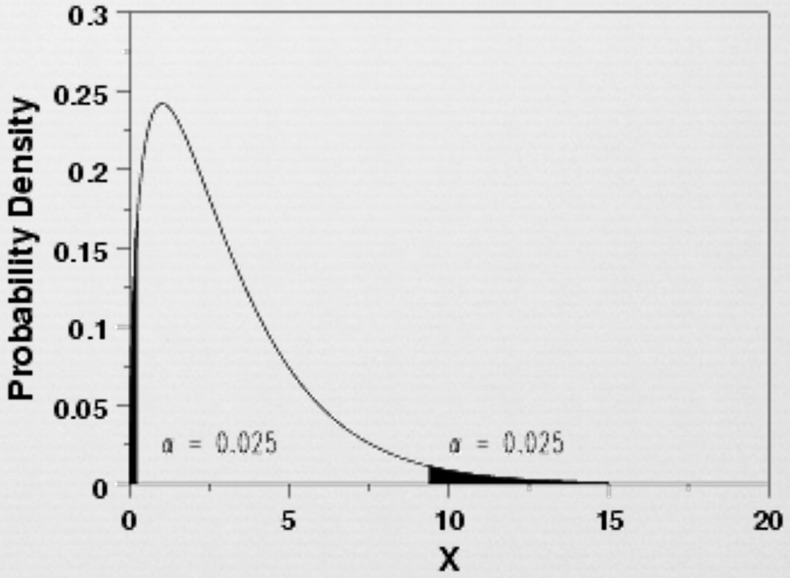
χ^2 statistic

- It is a probability distribution used to measure the error between some expected hypothesis and the real collected values

$$\chi^2 = \sum_{i=1}^{n} (O_i - E_i)^2$$

O_i Observed value for term i

E_i Expected value for term i



if x > 10.83 (the value for .999 confidence), the null hypothesis of independence is rejected with 99.9% confidence

χ^2 statistic

Simple formula for 2x2:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

$A = \#(t,c)$	$C = \#(\neg t,c)$
$B = \#(t,\neg c)$	$D = \#(\neg t, \neg c)$

t: term or data type (PSC, ~PSC)
c: category (HDB, ~HDB)

$N = A + B + C + D$

Do try this at home

Data

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

	PSC Scholars	Non PSC Scholars		
HDB	34 **A**	4,619,966 **C**	4,620,000	**A+C**
Condo/landed	37 **B**	1,379,963 **D**	1,380,000	**B+D**
	71 **A+B**	5,999,929 **C+D**	6,000,000	**N**

- 48% of the 71 PSC (Public Service Commission) scholars lives in HDB. 77% of Singapore's population (estimated to be 6 million) lives in HDB.

A4

$$\chi^2 = \frac{6{,}000{,}000 \times (34 \times 1{,}379{,}963 - 4{,}619{,}966 \times 37)^2}{(34 + 4{,}619{,}966) \times (37 + 1{,}379{,}963) \times (34 + 37) \times (4{,}619{,}966 + 1{,}379{,}963)}$$

- 33.97 > 10.83: Rejected with 99.9% confidence
- The null hypothesis is that the distribution of PSC scholars is independent of the population distribution in HDB and non-HDB. If the chi-square is higher than the critical value of p=0.001, you will reject the null hypothesis (which means the two variables are dependent and the distribution of PSC scholars is a reflection of the underlying population distribution).

Q5

- Table 5 shows the vector representation of five documents, where the training set includes d_1, d_2, d_3, and d_4, and d_5 belongs to the test set. Among the training documents, only the first three have been labeled as belonging to the target class u_c.
  a) Classify document d_5 using vector space classification
     i. Use Euclidean distance
     ii. Use cosine similarity
  b) Classify document d_5 using 1NN
     i. Use Euclidean distance
     ii. Use cosine similarity

Table 5

vector	term weights					
	Chinese	Japan	Tokyo	Macao	Beijing	Shanghai
$\vec{d}_1$	0	0	0	0	1.0	0
$\vec{d}_2$	0	0	0	0	0	1.0
$\vec{d}_3$	0	0	0	1.0	0	0
$\vec{d}_4$	0	0.71	0.71	0	0	0
$\vec{d}_5$	0	0.71	0.71	0	0	0
$\vec{\mu}_c$	0	0	0	0.33	0.33	0.33
$\vec{\mu}_{\bar{c}}$	0	0.71	0.71	0	0	0

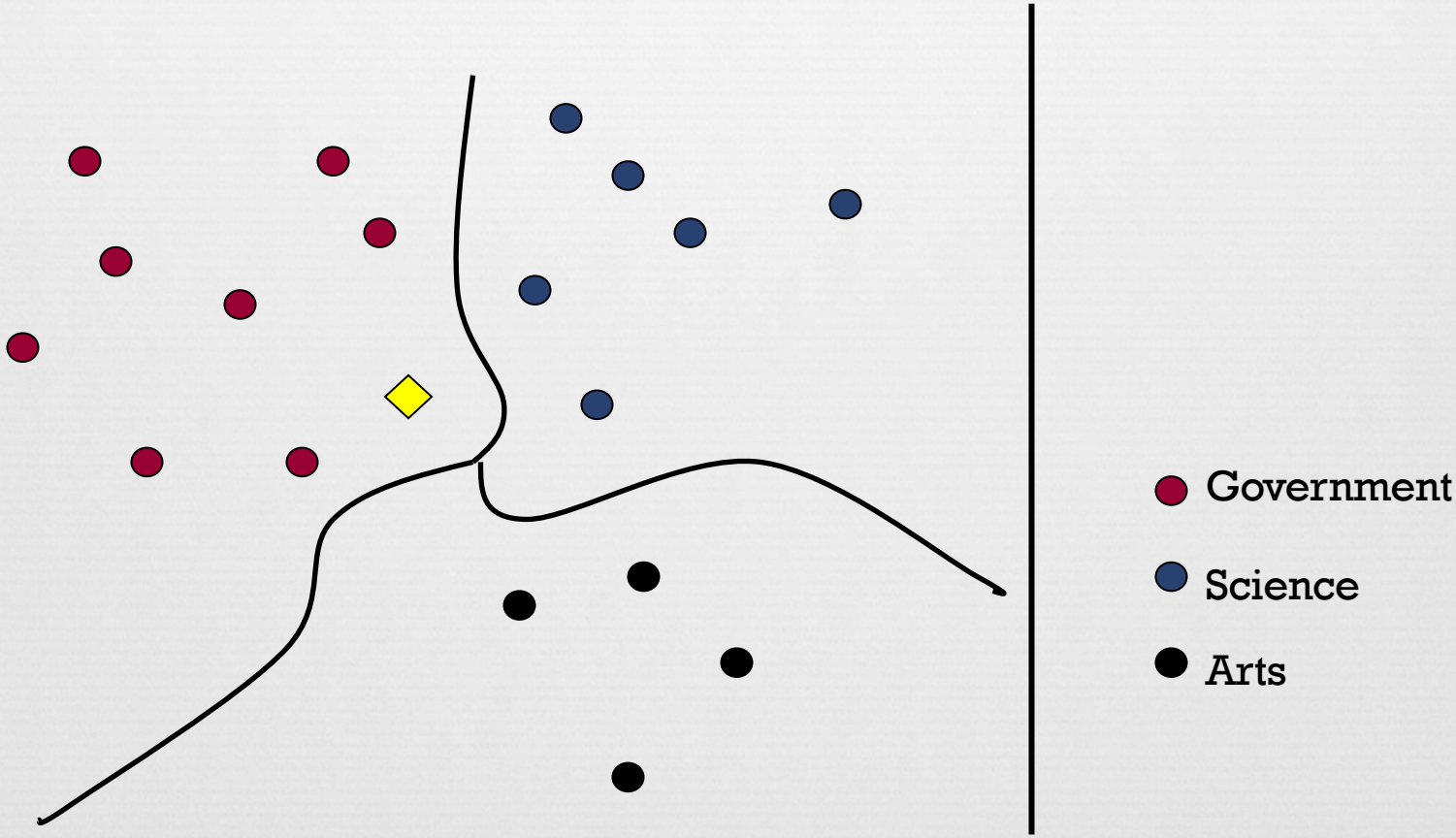
a) vector space classification

- Classify a new document by choosing the *nearest* centroid (minimum distance between new document and all centroids)

$$\vec{\mu}(c) = \frac{1}{|D_c|} \sum_{d \in D_c} \vec{v}(d)$$

- Where D_c is the set of all documents that belong to class c and $v(d)$ is the vector space representation of d

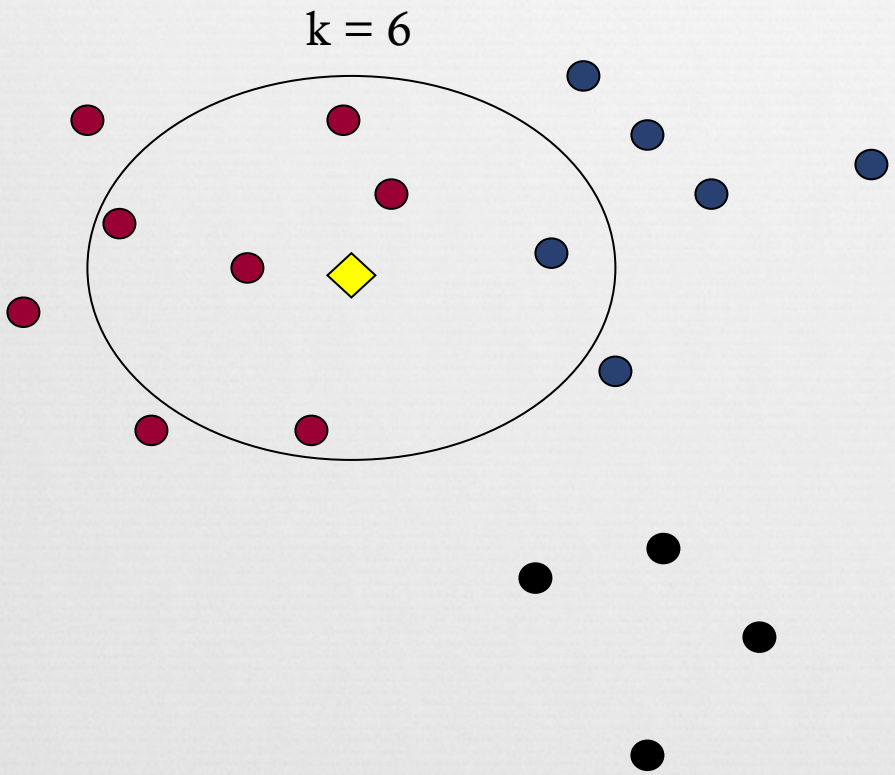
a) vector space classification



b) kNN classification

- To classify a document *d* into class c:
  - Define *k*-neighborhood N as *k* nearest neighbors of *d*
  - Count number of documents i in N that belong to c
  - Estimate P(c|*d*) as i/k
- Choose as class argmax_c P(c|*d*) [= majority class]

b) kNN classification



P(Government|◇)?
P(Science|◇)?
P(Arts|◇)?

- Government
- Science
- Arts

Do try this at home

Table 5

	term weights					
vector	Chinese	Japan	Tokyo	Macao	Beijing	Shanghai
$\vec{d}_1$	0	0	0	0	1.0	0
$\vec{d}_2$	0	0	0	0	0	1.0
$\vec{d}_3$	0	0	0	1.0	0	0
$\vec{d}_4$	0	0.71	0.71	0	0	0
$\vec{d}_5$	0	0.71	0.71	0	0	0
$\vec{\mu}_c$	0	0	0	0.33	0.33	0.33
$\vec{\mu}_{\bar{c}}$	0	0.71	0.71	0	0	0

centroids

A5.a i)-ii)

- Euclidean distance $\left|\vec{x}-\vec{y}\right| = \sqrt{\sum_{i=1}^{M}(x_i - y_i)^2}$
- Cosine similarity

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{\left|\vec{q}\right|\left|\vec{d}\right|} = \frac{\vec{q}}{\left|\vec{q}\right|}\bullet\frac{\vec{d}}{\left|\vec{d}\right|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

$\vec{d}_5$	0	0.71	0.71	0	0	0
$\vec{\mu}_c$	0	0	0	0.33	0.33	0.33
$\vec{\mu}_{\bar{c}}$	0	0.71	0.71	0	0	0

A5.a i)-ii)

- Euclidean distance

$$\left|\overrightarrow{d_5} - \overrightarrow{\mu_C}\right| = \sqrt{0.71^2 + 0.71^2 + 0.33^2 + 0.33^2 + 0.33^2} \approx 1.155$$

$$\left|\overrightarrow{d_5} - \overrightarrow{\mu_{\bar{C}}}\right| = 0$$

- Cosine similarity

$$\cos(\vec{d}_5, \vec{\mu}_C) = 0$$

$$\cos(\vec{d}_5, \vec{\mu}_{\bar{C}}) = \frac{0.71^2 + 0.71^2}{\sqrt{0.71^2 + 0.71^2}\sqrt{0.71^2 + 0.71^2}} = 1$$

$\vec{d}_5$	0	0.71	0.71	0	0	0
$\vec{\mu}_C$	0	0	0	0.33	0.33	0.33
$\vec{\mu}_{\bar{C}}$	0	0.71	0.71	0	0	0

A5.b i)-ii)

d1, d2 d3 all equally close to the uc centroid

$$\left|\vec{d_5}-\vec{d_1}\right| = \left|\vec{d_5}-\vec{d_2}\right| = \left|\vec{d_5}-\vec{d_3}\right| = \sqrt{0.71^2+0.71^2+1^2} \approx 1.417$$

$$\left|\vec{d_5}-\vec{d_4}\right| = 0$$

d4 closest to uc~ centroid

$$\cos(\vec{d}_5,\vec{d}_1) = \cos(\vec{d}_5,\vec{d}_2) = \cos(\vec{d}_5,\vec{d}_3) = 0$$

$$\cos(\vec{d}_5,\vec{d}_4) = \frac{0.71^2+0.71^2}{\sqrt{0.71^2+0.71^2}\sqrt{0.71^2+0.71^2}} = 1$$

$\vec{d}_1$	0	0	0	0	1.0	0
$\vec{d}_2$	0	0	0	0	0	1.0
$\vec{d}_3$	0	0	0	1.0	0	0
$\vec{d}_4$	0	0.71	0.71	0	0	0
$\vec{d}_5$	0	0.71	0.71	0	0	0

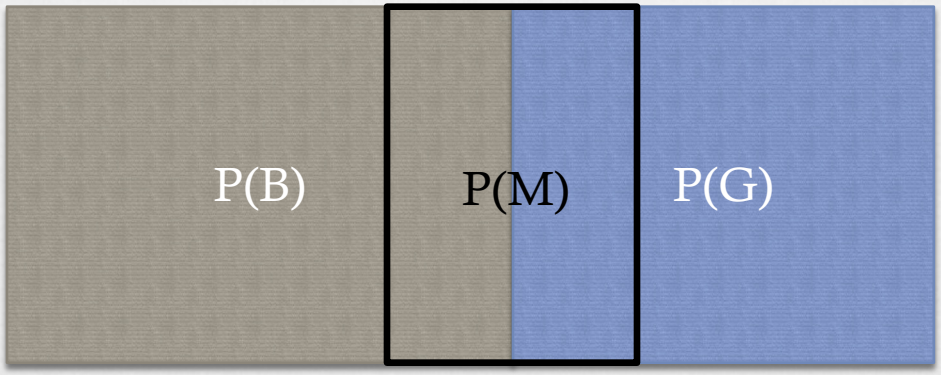
Tutorial 7

SC4021/CE4034/CZ4034

Q1

- Suppose 100 people attended a party at Zouk, 40 girls (G) and 60 boys (B), among which are 20 married couples (1 married couple M = 1 girl + 1 boy). Compute the following probabilities.
  - What is the probability that you will run into a boy if you attend the party?
  - Suppose you throw a bunch of flowers randomly at a person in the party, what is the probability that it will land on a girl?
  - Suppose you see an attractive girl at the party, what is the probability that she is married?
  - Suppose you see a gorgeous hunk at the party, what is the probability that he is married?

Hint



Do try this at home

A1

a) $P(B) = 60/100 = 0.6$

b) $P(G) = 40/100 = 0.4$

c) $P(M|G) = P(M,G)/P(G) = (20/100)/(40/100) = 0.50$

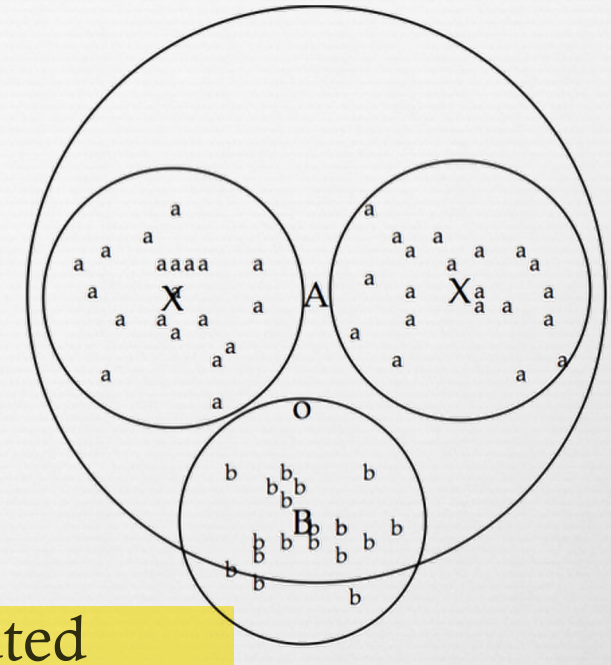
d) $P(M|B) = P(M,B)/P(B) = (20/100)/(60/100) = 0.33$

Q2

- Briefly explain why kNN handles multimodal classes better than the standard vector space model (VSM) where you classify based on cosine similarity.

Do try this at home

A2



- VSM assumes each class is cohesive and can be represented by 1 centroid, thus for multimodal classes, the computed centroid may fall into an area with no training data
- kNN does not make any assumptions on the class distribution, it simply looks at the k nearest neighbors of the test point. If the neighborhood has more training points from 1 class, the test point will simply be assigned to that class
- kNN can be viewed as a local density based classifier

Q3

- Consider a British news article about Toyota automobiles and an American news article about Toyota cars, where the former uses “automobile” and the latter uses “car”. Why are documents that do not use the same term for the concept *car* likely to end up in the same cluster in K-means clustering?

What is clustering?

- Clustering: the process of grouping a set of objects into classes of similar objects
  - Documents within a cluster should be similar
  - Documents from different clusters should be dissimilar
  - Classes are not pre-defined
- The most common form of *unsupervised learning*
  - A common and important task that finds many applications in IR and other places

K-means

- Assumes documents are real-valued vectors
  - as in Vector Space Classification, e.g. (0.1, 0.02, …, 0.0001)
- Clusters based on *centroids* (aka the *center of gravity* or mean) of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

- Reassignment of instances to clusters is based on distance to the current cluster centroids
  - Or one can equivalently phrase it in terms of similarities

Do try this at home

A3

- Documents in the same cluster are similar, where similarity is defined by the dot product of the cosine similarity or Euclidean distance
- Documents about the same concept (e.g., cars) but not using the same term (i.e., *car* versus *auto*) are likely to have a lot of other common terms like Toyota, brakes, tires, radio, wiper, speed, etc., which will put them into the same cluster

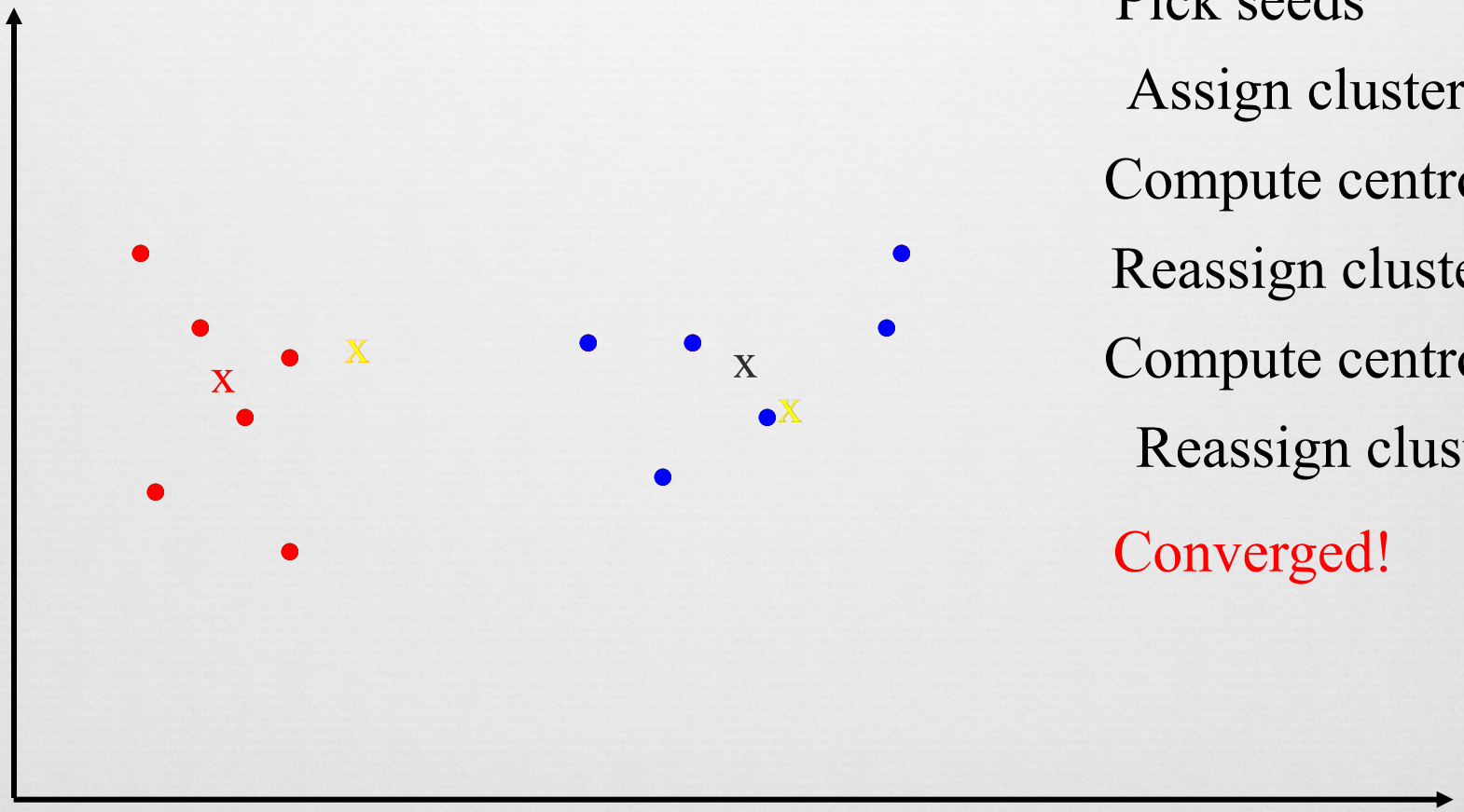
Q4

- Two of the possible termination conditions for K-means are (1) assignment does not change, and (2) centroids do not change. Do these two conditions imply each other?

Termination conditions

- Several possibilities, e.g.,
  - A fixed number of iterations.
  - Doc clusters unchanged.
  - Centroid positions don't change.

K Means Example (*K*=2)



Pick seeds

Assign clusters

Compute centroids

Reassign clusters

Compute centroids

Reassign clusters

Converged!

Do try this at home

A4

- Yes
- Cluster assignments do not change => Each cluster contains the same points as in the previous iteration=> Centroids do not change
- Centroids do not change => Each point is still closest to the same centroid as in the previous iteration => point assignment does not change

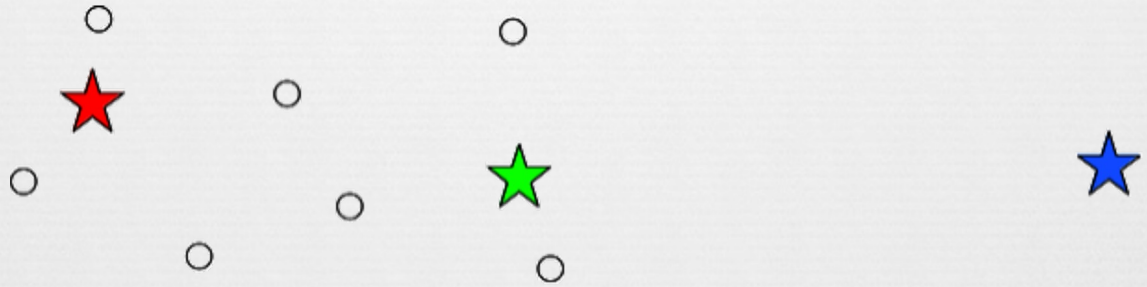
Q5

- Give an example of a set of points and three initial centroids (which need NOT be members of the set of points) for which 3-means converges to a clustering with an empty cluster

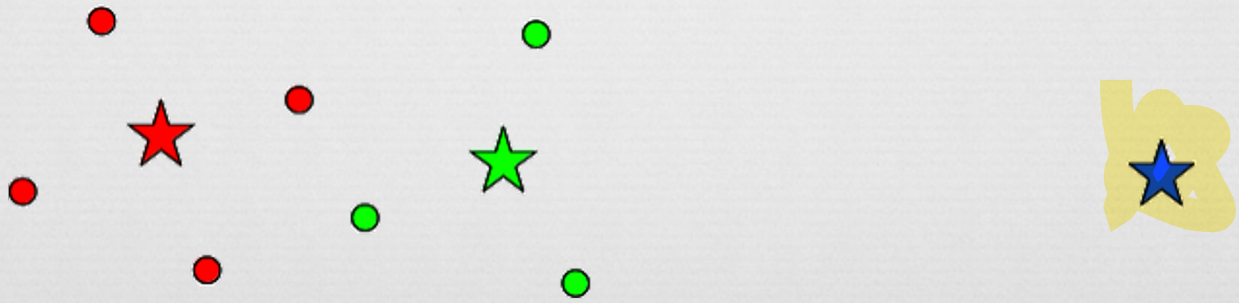
Do try this at home

A5

- Initial centroids

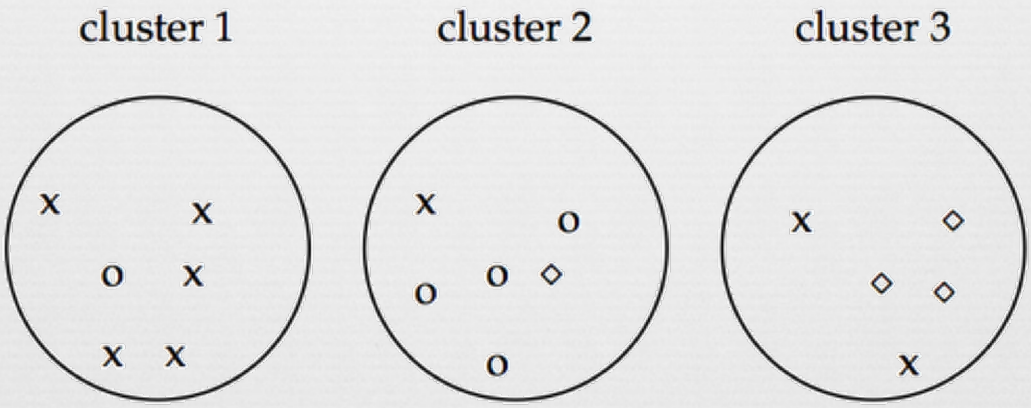


- After convergence, with an empty cluster for blue centroid



Q6

- Explain how to compute the Rand Index using the example in Figure 1.



► **Figure 1** Purity as an external evaluation criterion for cluster quality. Majority class and number of members of the majority class for the three clusters are: x, 5 (cluster 1); o, 4 (cluster 2); and ◇, 3 (cluster 3). Purity is $(1/17) \times (5 + 4 + 3) \approx 0.71$.

Rand index between pair decisions

Number of points	Same cluster in clustering	Different clusters in clustering
Same class in ground truth	20 A	24 C
Different classes in ground truth	20 B	72 D

Rand index and cluster F-measure

$$RI = \frac{A + D}{A + B + C + D}$$

A: true positives
B: false positive
C: false negatives
D: true negatives

Compare with standard Precision and Recall:

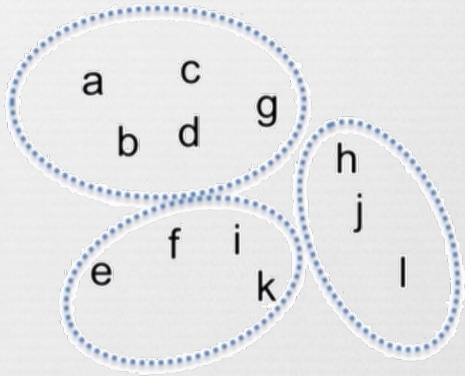
$$P = \frac{A}{A + B} \qquad R = \frac{A}{A + C}$$

People also define and use a cluster F-measure, which is probably a better measure

Rand Index

- Sample pairs x_i, x_j
  - ask human if x_i, x_j should be in the same group
  - easy task (cognitively)
  - can't ask them to "cluster" dataset manually

a,b = Yes
c,d = No
e,h = Yes
g,h = No



- System produces clusters
- Count errors, compute accuracy, F1, etc
  - FN: matching pairs x_i, x_j that are in different clusters (e,h)
  - FP: non-matching pairs x_i, x_j that are in same cluster (c,d)

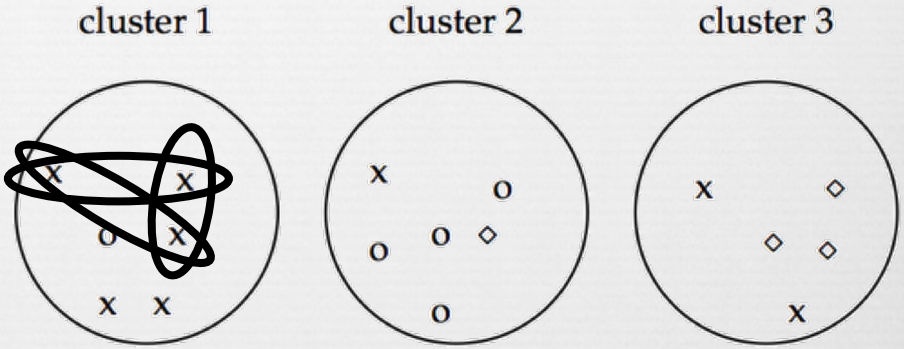
http://youtu.be/HdE9h__Xb2A

Do try this at home

A6

A: true positives
B: false positive
C: false negatives
D: true negatives

same cluster, same class



5: number of x in cluster 1
5*(5-1)/2: number of x pairs in cluster 1
4: number of o in cluster 2
4*(4-1)/2: number of o pairs in cluster 2
3: number of ◇ in cluster 3
3*(3-1)/2: number of ◇ pairs in cluster 3
1: number of x pair in cluster 3
A = 5*4/2 + 4*3/2 + 3*2/2 +1 = 20

5: number of (x, o) pairs in cluster 1
9: number of (x, o) pairs + (o, ◇) pairs + (x, ◇) pair in cluster 2
6: number of (x, ◇) pairs in cluster 3
B = 5 + 9 + 6 = 20

same cluster, different class

---

17 = 5*1 (x pairs from clusters 1&2) + 5*2 (x pairs from clusters 1&3) + 1*2 (x pairs from clusters 2&3)
4 = 1*4 (o pairs from clusters 1&2)
3 = 1*3 (◇ pairs from clusters 2&3)
C = 17 + 4 + 3 = 24

different cluster, same class

D = 17*(17-1)/2 (total number of pairs) - (A+B+C)
D = 136 – (20+20+24) = 72

different cluster, different class

A6

Number of points	Same cluster in clustering	Different clusters in clustering
Same class in ground truth	20 A	24 C
Different classes in ground truth	20 B	72 D

$$RI = \frac{A + D}{A + B + C + D} = (20+72)/(20+20+24+72) = 0.68$$

Tutorial 8

SC4021/CE4034/CZ4034

Q1

- The Goto method ranked click-through advertisements matching a query by bid: the highest-bidding advertiser got the top position, the second- highest the next, and so on. What can go wrong with this when the highest-bidding advertiser places an advertisement that is irrelevant to the query? Why might an advertiser with an irrelevant advertisement bid high in this manner?

Goto (1996)

- Ads are ranked simply based on bids
  - The highest bidder gets the first rank
  - The bidder pays the money every time somebody clicks the link
- Revenue maximization
- No relevance ranking

Do try this at home

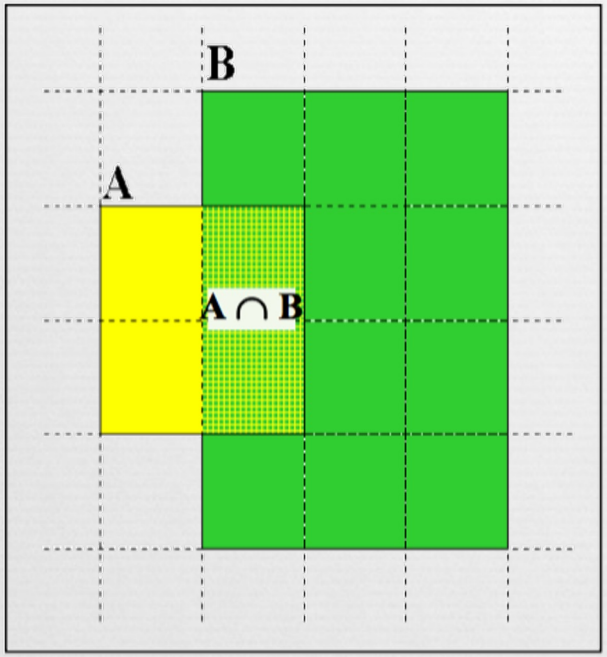
A1

- Since the ad is irrelevant, it is seldom clicked by users
- No click => no $ is paid to the search engine
- Ad placement = free branding/publicity for company

Q2

- Each of two Web search engines A and B generates a large number of pages uniformly at random from their indexes. 30% of A's pages are present in B's index, while 50% of B's pages are present in A's index. What is the ratio between the number of pages in A's index and the number of pages in B's?

Relative size of search engines



Sample URLs randomly from A

Check if contained in B and vice versa

$$A \cap B = (1/2) * \text{Size } A$$

$$A \cap B = (1/6) * \text{Size } B$$

$$(1/2)*\text{Size } A = (1/6)*\text{Size } B$$

$$\therefore \text{ Size } A / \text{ Size } B = (1/6)/(1/2) = 1/3$$

Each test involves: (i) Sampling (ii) Checking

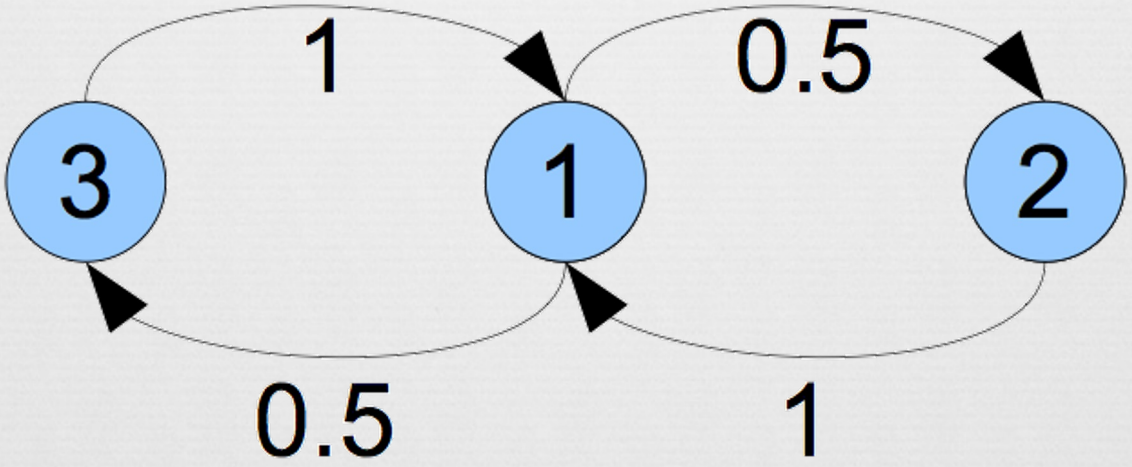
Do try this at home

A2

- 30% x A = 50% x B
- 30/100 x A = 50/100 x B
- A = (50/100) x (100/30) x B
- A/B = 5/3

Q3

- Write down the transition probability matrix for the following Markov chain. Is this Markov chain ergodic?

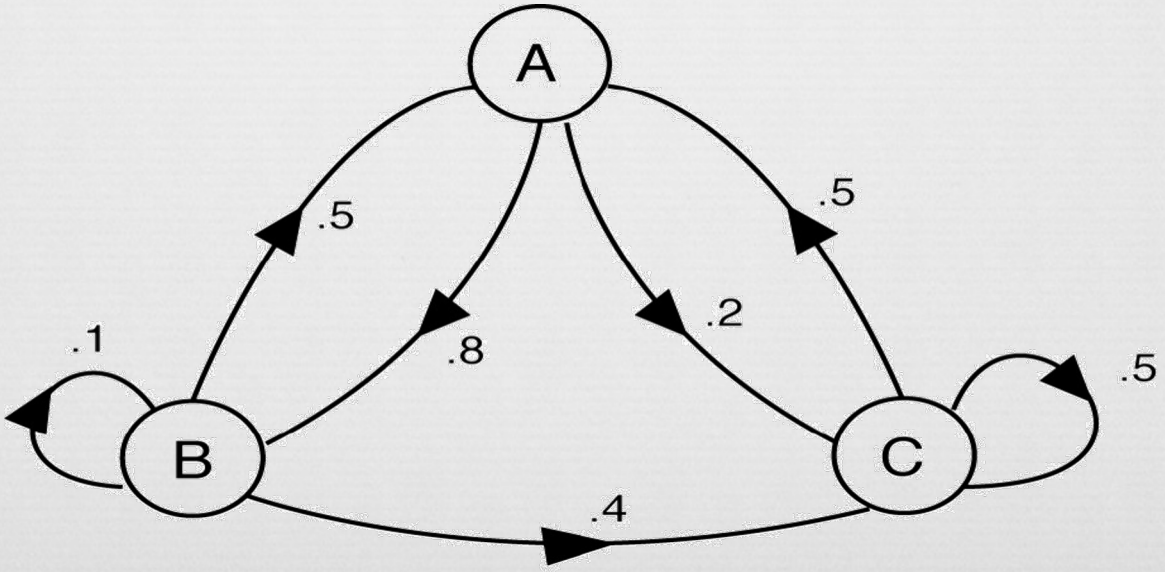


Page Rank

- Compute the long-term visiting rate of a random walker on the web (probability of visiting page d at a given point in time)
- How do we compute the long-term visiting rate?
  - Model the random walker as a **Markov chain** model

Markov Chain

- Markov Chain: stochastic process where the current state depend only on the previous one
- The position of a random walker on the Web only depends on the previous visited page



Markov Chain

- Composed by:
  - A set of **N state**
  - A **transition matrix** NxN that define the probability to jump to the next state
- The long-term visiting rate is computed as the **steady-state** of the Markov chain.

Ergodic Markov Chain

- A Markov chain is ergodic if
  - It has a path from any state to any other
  - a positive integer T0 exists such that for any start state, the probability of being in any state at all t > T0 is nonzero

- Two technical conditions
  - Irreducibility. Roughly: there is a path from any page to any other page (teleportation)
  - Aperiodicity. Roughly: The pages cannot be partitioned such that the random walker visits the partitions sequentially (we don't do every time the same paths)

Hint

- Try to find the steady state probability when starting from any initial state
- If converges, it is ergodic; otherwise, not.
- Initial position: $\vec{\mathbf{x}}$ (probability vector of being on a give page)
- Transition matrix: $\mathbf{P}$

$$\vec{\mathbf{x}}\,\mathbf{P}^{\mathbf{n}} = \vec{\mathbf{x}}$$

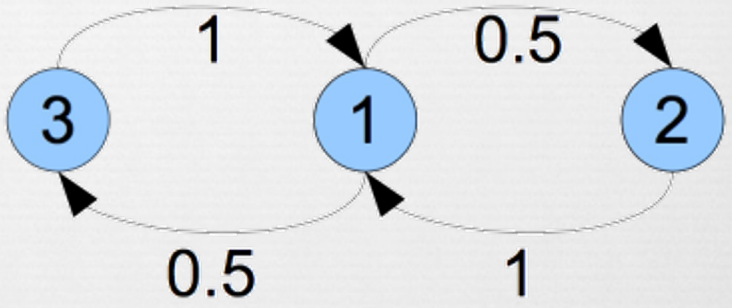
The Markov chain is ergodic

Do try this at home

A3

P

	d_1	d_2	d_3
d_1	0	0.5	0.5
d_2	1	0	0
d_3	1	0	0



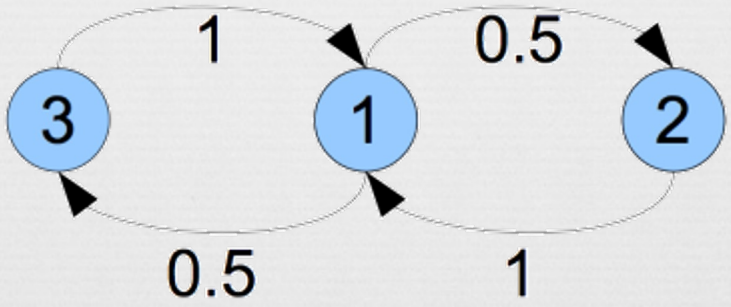
Initial probability vector **x**

0	0	1

	d_1	d_2	d_3
t(0) = x	0	0	1
t(1) = t(0) P	1	0	0
t(2) = t(1) P	0	0.5	0.5
t(3)	1	0	0
t(4)	0	0.5	0.5

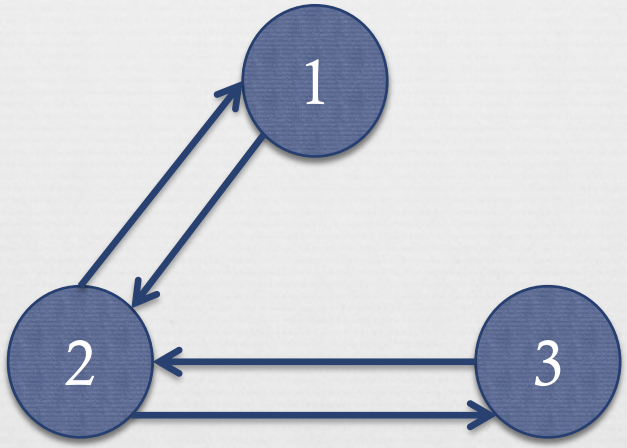
A3

- The chain is periodic
  - e.g., starting from [0 0 1], the final state oscillates between [1 0 0] and [0 .5 .5]



Q4

- Write down the transition probability matrices for the surfer's walk with teleporting probability $\alpha = 0.3$.



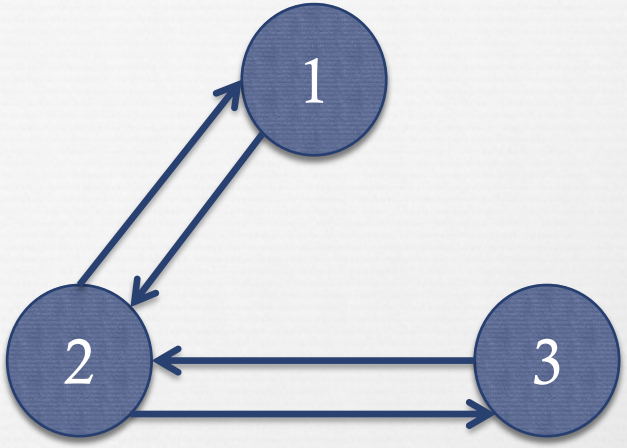
Transition probability matrix

- Step1: If there is a hyperlink from page i to page j, then Aij = 1, otherwise Aij = 0
- Step 2: Divide each 1 in A by the number of 1's in its row
- Step 3: Multiply the resulting matrix by 1 - α
- Step 4: Add α/N to every entry

	d_0	d_1	d_2
d_0	0	0	1
d_1	0	1	1
d_2	1	0	1

Do try this at home

A4



Step1: If there is a hyperlink from page i to page j: Aij = 1, else: Aij = 0

d_1	d_2	d_3
0	1	0
1	0	1
0	1	0

Step 2: Divide each 1 in A by the number of 1's in its row

d_1	d_2	d_3
0	0.7	0
0.35	0	0.35
0	0.7	0

Step 3: Multiply the resulting matrix by 1 - α

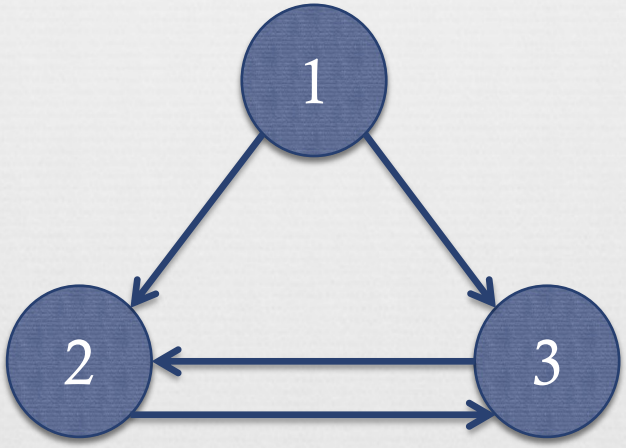
d_1	d_2	d_3
0	1	0
0.5	0	0.5
0	1	0

Step 4: Add α/N to every entry

d_1	d_2	d_3
0.1	0.8	0.1
0.45	0.1	0.45
0.1	0.8	0.1

Q5

- Compute the Hubs and Authorities scores for each of the 3 nodes in the following graph. Normalize your hubs and authorities values to lie within [0, 1].



Hubs and authorities

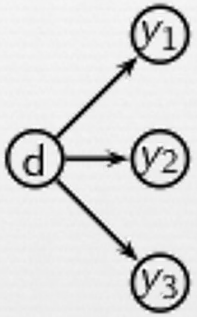
- A good hub page for a topic links to many authority pages for that topic.
- A good authority page for a topic is linked to by many hub pages for that topic.
- Circular definition – we will turn this into an iterative computation.

Hubs and authority scores

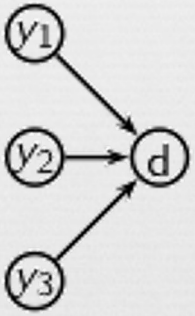
- Compute for each page d in the base set a hub score h(d) and an authority score a(d)
- Initialization: for all d: h(d) = 1, a(d) = 1
- Iteratively update all h(d), a(d)
- After convergence:
  - Output pages with highest h scores as top hubs
  - Output pages with highest a scores as top authorities
  - So we output two ranked lists

Iterative update

- For all d: h(d) = $\sum_{d \to y} a(y)$



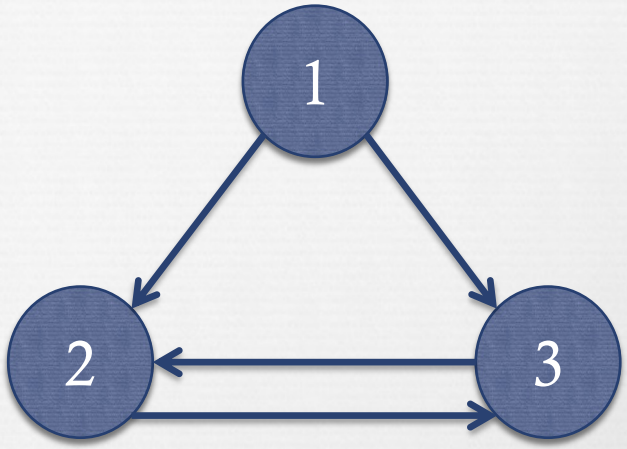
- For all d: a(d) = $\sum_{y \to d} h(y)$



- Iterate these two steps until convergence

Do try this at home

A5



		d_1	d_2	d_3
	$a(d)$	1	1	1
	$h(d)$	1	1	1

		d_1	d_2	d_3
Iteration 1	$a(d)$	0	2	2
	$h(d)$	2	1	1

		d_1	d_2	d_3
Iteration 1' (normalize)	$a(d)$	0	1	1
	$h(d)$	1	0.5	0.5

		d_1	d_2	d_3
Iteration 2	$a(d)$	0	1.5	1.5
	$h(d)$	2	1	1

		d_1	d_2	d_3
Iteration 2' (normalize)	$a(d)$	0	1	1
	$h(d)$	1	0.5	0.5

it converges because Iteration 2' is equal to Iteration 1'

Q6

- If all the hub and authority scores are initialized to 1, what is the hub/authority score of a node after one iteration?

Do try this at home

A6

- Initialize h(v)=a(v)=1
- After 1 iteration
- $h(v) = \sum_{v \to y} a(y)$ = # outlinks in page v
- $a(v) = \sum_{y \to v} h(y)$ = # inlinks to page v

Key concepts (2nd half)

- IR evaluation
- Classification
- Feature selection
- Clustering
- IR on the Web