GOPI

Page No.
Date :

→ Document classific^n

eg- Naive Bayes, Bernoulli

→ Role of doc. classific^n in IRS

↳ entertainment
↳ politics
↳ doc.3
} mapping of doc.

- we have training & testing.
- Repo. of doc. with labels available (eg- doc-1 :- sports etc.)
- when new doc comes, we can automatically classify entertainment or politics /others

- By creating clusters we can reduce our search space.
- more relevant queries
- reduction of search time. (optimiz^n of retrieval time)
- eg - in email :- spam or not spam
- more precision & recall.
- if image, audio, videos etc. already classified, then helpful for specific query like - cat image

- Naive Bayes most popular for text classific^n.

→ Supervised Learning

- A doc. d
  classes :- $C = \{c_1, c_2, \ldots, c_j\}$
  - A training set of doc. with a label from c.

Bernoulli Trial ⇒ 1 variable, 2 values ⇒ 0 or 1
multivariate Bernoulli trial ⇒ multiple variable with 0 or 1 for each test.

$$P(c/d) = \frac{P(c) \times P(d/c)}{P(d)}$$ ↳ for Naive Bayes

- Determine
  - A learning classifier to classify class y.
  - we assign class y to d
    $y(d) \in C$
  - we train a model to classify on test object/data.

⇒ we use Naive Bayes

$$P(c/d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k/c)$$

- $P(c)$: prob. of d belonging to class c
- $\prod$: multiplication of P(term present in d) for all terms
- denominator term ignored assume for all $\frac{3}{5} > \frac{2}{5}$ ↳ $\frac{3}{5} > \frac{2}{5}$

- If class c is popular, then new doc. is highly probable for new test doc d.
  (eg 90 of 100 are C, so C is probable for test d).

- n_d = length of doc. (no. of tokens)
- $P(t_k/c)$ is conditional prob. of term t_k occuring in doc. of class c (including multiple occurrences)
- $P(t_k/c)$ as a measure of how much evidence t_k contributes that c is correct class
  ↳ we check for each term t, how many times it appears in all docs. of c.

Page No.
Date :

- Our goal in naive Bayes is to find best class.
- Best class is most likely or maximum a posterial (MAP) class. · C_{map} is calculated
- use argmax.

C_{map} $P(c) = \frac{N_c}{N}$ → docs of class c. → Prior prob.

N → total docs in corpus

conditional prob. : $P(t/c) = \frac{T_{ct}}{\sum_{t \in V} T_{ct'}}$

→ eg- toppers 3, 2, 4 | 11 mg in dal 23. ∴ T_{ct} = 3+2+1 = 6

- we make naive bayes positional independance assumption

$P(t_{k_1}/c) = P(t_{k_2}/c)$

↓ 1st place or any other position

↳ same prob.

- we need to add 0s conditional probability due to term which may not be in training phase, prob. becomes 0
  ↳ so need for Laplacian correction.

eg class a, b, c

$P(W/a) = 0$ ↳ if W not in a.

Page No.
Date

So, we add +1 to numerator, & no. of classes in denominator.

∴ ~~P(c)~~ eg - Test = chinese chinese chinese Tokyo Japan.

3 classes - c
1 class - j

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

$P(\text{chinese} / c) = \frac{5+1}{8+6}$

→ 5 chinese in c
→ laplacian correctn
8 → total words in c
6 → vocabulary (entire corpus) ~~total chinese in doc of c~~

$P(\text{tokyo} / c) = \frac{0+1}{8+6} = \frac{1}{14}$

$P(\text{Japan} / c) = \frac{0+1}{8+6} = \frac{1}{14}$

→ chinese appears 3 times.

$P(c / d_5) \propto \frac{3}{4} \times \left(\frac{3}{7}\right)^3 \times \frac{1}{14} \times \frac{1}{14}$

Similarly for P(chinese / j), P(tokyo / j) & P(Japan / j)

multiple ⇒ $P(j / d_5)$

Whichever greater is class.

$P(w/c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}$ → vocabulary.

Page No.
Date :

⇒ Bernoulli

- Doesn't see term freq. only presence or absence.
- P(t/c) estimated as fract^n of doc of class containing term t.
- Probability of non occurence is also calculated.
- Binary values put - 0 & 1.

~~eg P(chinese/c) = 3+1~~

- we calculate probability of each unique term in corpus (to check if term may not appear).

⇒ P(chinese/c) = $\frac{3+1}{3+2}$ → in how many docs it appears → Laplacian.
↓ total docs of c ↓ total classes (2)

P(Japan/c) = P(Tokyo/c) = $\frac{0+1}{3+2} = \frac{1}{5}$

P(chinese/J) = $\frac{1+1}{1+2}$

similarly others.

∴ P(c/d) ∝ P(c) × P(chinese/c) × P(Japan/c) × P(Japan Tokyo/c) × (1 - P(macau/c)) × (1 - P(shanghai/c)) × (1 - P(Beijing/c))
~~× P(macau/c) × P(shanghai/c)~~

multinomial

Naive Bayes question

training set	doc	words	in c=China?
	1	Chinese Beijing Chinese	yes
	2	Chinese Chinese Shanghai	yes
	3	Chinese Macao	yes
	4	Tokyo Japan Chinese	no
test →	d 5	Chinese Chinese Chinese Tokyo Japan	?

→ $P(\text{Chinese}/c) = \frac{5+1}{8+6} = \frac{6}{14} = \frac{3}{7}$

$P(\text{Tokyo}/c) = \frac{0+1}{8+6} = \frac{1}{14}$

$P(\text{Japan}/c) = \frac{0+1}{8+6} = \frac{1}{14}$

$P(c) = \frac{3}{4}$

$\therefore P(c/d_5) \propto \frac{3}{4} \times \left(\frac{3}{7}\right)^3 \times \frac{1}{14} \times \frac{1}{14} \approx 0.0003$

→ $P(\text{Chinese}/\bar{c}) = \frac{1+1}{3+6} = \frac{2}{9}$

$P(\text{Tokyo}/\bar{c}) = \frac{1+1}{3+6} = \frac{2}{9}$

$P(\text{Japan}/\bar{c}) = \frac{1+1}{8+6} = \frac{2}{9}$, $P(\bar{c}) = \frac{1}{4}$

$P(\bar{c}/d_5) \propto \frac{1}{4} \times \left(\frac{2}{9}\right)^3 \times \frac{2}{9} \times \frac{2}{9} \approx 0.0001$

$\therefore$ c is more probable

. . Laplacian correction is for smoothening.

Page No.
Date :

steps

- From training corpus extract vocabulary → unique words
- Calculate req. $P(C_j)$ & $P(x_k / C_j)$ terms.

For each class C_j in C do:

- $docs_j \leftarrow$ subset of docs for which the target class is C_j

$$P(C_j) = \frac{|docs|_j}{\text{total docs}} \quad \text{i.e. } \frac{\text{docs of class } C_j}{\text{total docs}}$$

- $P(x_k / C_j) = \frac{n_{jk} + \alpha}{n_j + \alpha |\text{vocabulary}|}$

where
$k = 1$ to all unique words in query

n_j → no of words in doc_j

n_{jk} → in how many docs x_k appears. → for laplacian correct^n

$|\text{vocabulary}|$ → no of unique words in corpus.

positions = all word position in current doc. which contain tokens found in vocab.

we calculate ⇒ $C_{NB} = \underset{C_j \in C}{\text{argmax}} \left(P(C_j) \times \prod_{i \in \text{positions}} P(x_i / C_j) \right)$

→ for class having max^m value.

→

Page No.
Date :

Bernoulli Naive Bayes Classifier

- considers presence & absence of words (0 or 1)
- ∴ 2 possible values for a word ⇒ 0 or 1.

$$P(x_i/c_j) = \frac{N(\text{no. of time word appears and } x_i = x_i,\ C = c_j) + 1}{N(C = c_j) + k} \quad \text{(laplacian correction)}$$

- $N(C = c_j)$: no. of docs with label C
- k: no. of values for a word (here 2 - 0 or 1)

	doc ID	words	in c = china?
training set	1	Chinese Beijing Chinese	yes
	2	chinese chinese Shanghai	yes
	3	chinese macao	yes
	4	Tokyo Japan chinese	no
test set	d_5	Chinese Chinese Chinese Tokyo Japan	?

$$P(\text{Chinese}/c) = \frac{3+1}{3+2} = \frac{4}{5}$$

(3 = chinese appears in 3 docs of c; 3 + 2 = total docs of c; + 2 = appears or not (0,1))

$$P(\text{Tokyo}/c) = \frac{0+1}{3+2} = \frac{1}{5}$$

$$P(\text{Japan}/c) = \frac{0+1}{3+2} = \frac{1}{5}$$

$$P(\text{Beijing}/c) = \frac{1+1}{3+2} = \frac{2}{5}, \quad P(\text{macao}/c) = \frac{1+1}{3+2} = \frac{2}{5} = P(\text{Shanghai}/c)$$

$P(c) = \frac{3}{4}$ ∴ $P(d_5/c) \propto \frac{3}{4} \times \frac{4}{5} \times \frac{1}{5} \times \frac{1}{5} \times (1 - \frac{2}{5}) \times (1 - \frac{2}{5}) \times (1 - \frac{2}{5}) \approx 0.005$

$P(\text{Chinese}/\bar{c}) = \frac{1+1}{1+2} = \frac{2}{3}$

$P(\text{Tokyo}/\bar{c}) = \frac{1+1}{1+2} = \frac{2}{3} = P(\text{Japan}/\bar{c})$

$P(\text{Beijing}/\bar{c}) = P(\text{Macao}/\bar{c}) = P(\text{Shanghai}/\bar{c})$

$= \frac{0+1}{1+2} = \frac{1}{3}$

$P(\bar{c}) = \frac{1}{4}$

$\therefore P(\bar{c}/d_5) \propto \frac{1}{4} \times \frac{2}{3} \times \frac{2}{3} \times \frac{2}{3} \times (1-\frac{1}{3}) \times (1-\frac{1}{3})$

≈ 0.022

$\therefore \boxed{\bar{c}}$ is predicted for d_5

→ How to improve relevance through feedback

- Objective is to provide relevant docs to user
- we want to improve relevance of docs ⇒ goal
- eg- Purchase implies high rating
  - if we click of retrieved result, like we click 4th docs out of 5 retrieved, then 1st 3 are not relevant, but 4 was. This is a kind of feedback
- we do query expansion.
- we want to provide better relevance to users.
- Improved relevance model, can be provide to same or other users, when same query was asked.
  ↳ all this is query expansion.

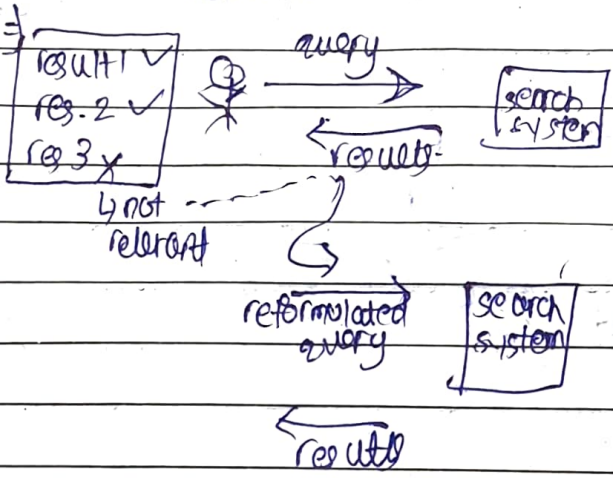
eg - we ask "plane"
- results :- (synonyms as well as wrongly spelled ones are retrieved)
  = flight
  = aircraft
  = by air
  = aeroplane
  = by air
  = fly
  = ~~flight~~ figt
  = arcrft.

→ How to ensure good results

automatic query refinement

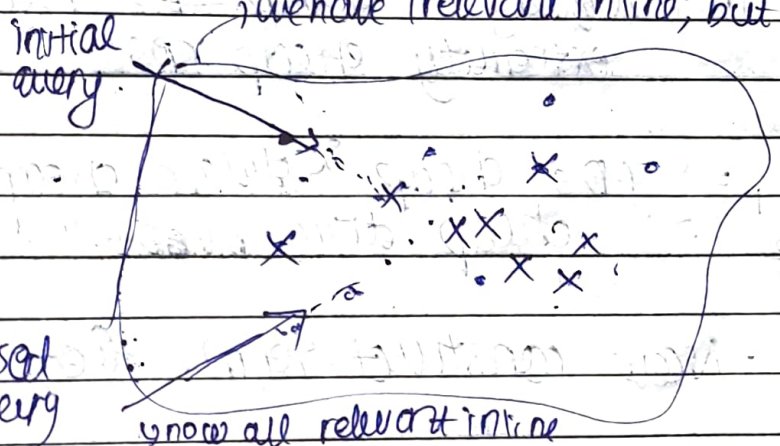
local
- use query or retrieved results to reformulate & improve query.
- for relevance
eg - pseudorelevance
- indirect relevance feedback.
- relevance feedback

global
- we do not use query or results to improve & reformulate query for relevance
eg - thesaurus
- spelling correct



- Refmet takes explicit feedback from user.
- It asks if result is relevant or not.
- But its bad
  - user doesn't have time & patience to provide explicit feedback
- But, still it is one of the way.
- Indexed content is unknown to user
  - ↳ we don't know, if other relevant docs. are present but not shown
- "Inform" need" changes after looking at the results.
  - ↳
  - eg- we search for a specific song. But looking at song results, we change our mind & search other song.
  - so need changes & its necessary to do query expansion..

→ Rocchio algorithm (Relevance feedback (local))



- can we shift our query from non relevant to relevant docs.? → do query expansion

- In practice we only know few relevant & non relevant.

$\vec{q_0}$ = initial query vector
D_r = set of relevant docs
D_{nr} = set of non rel. docs

↑ its weight
↓ its weight.

$$\vec{q_m} = \alpha \vec{q_0} + \beta \frac{1}{|D_r|} \sum_{d_j \in D_r} \vec{d_j} - \gamma \frac{1}{|D_{nr}|} \sum_{d_j \in D_{nr}} \vec{d_j}$$

- α, β, γ are weights.

- we represent query as weighted vector.
- use Rocchio formula to get new query vector.
- Now calculate cosine similarities
- Rocchio model increases recall
- +ve feedback is much more valuable than -ve feedback

relevant

eg- initial query = "cheap CDs cheap DVDs extremely cheap CDs"

relevant ← d_1 = "CDs cheap software cheap CDs"
not relevant ← d_2 = "cheap drills DVDs".

- Now construct term. freq. vector

	cheap	cds
q_0	3	-
d_1	2	
d_2	1	

- eg- to move (2,2) to (5,2), add 3 to x.

$q_{new} = 1 \times q_0 + 0.75 \times d_1 - 0.25 \times d_2$

- do this for each term.

eg- $q_{new} = 1 \times 3 + 0.75 \times 2 - 0.25 \times 1$

$= 4.25$

$q_{new} =$ | 4.25 | | | |

- we ~~cant~~ can have $q_{new} = -0.5$, this is wrong, so -ve values are put as 0.

$\therefore q_{new} = 0$

→ Pseudo Relevance (Blind) Relevance Feedback

- No user judgement
- we completely trust our web search engine
- we assume top k given by model are relevant to query
- now we can use rocchio algorithm to change the query.

→

problem

- we search "Dhoni retires from 1st class cricket".
  - now, southafrica tests, etc. are also relevant hence retrieved.
  - so matrix includes southafrica, newzealand etc.
  - The vect query ⇀ vector will thus drift.
  - so, query drift problem occurs, as more weightage to terms which are not relevant.

→ Indirect (implicit) Relevance Feedback

- we do not ask explicit feedback.
- we consider clickstream mining
  - user clicks on nth doc, then top n-1 not relevant, & n is relevant
- query expansion would be be done based on implicit user preference (clicks)

→ In global approaches, we do not consider either query or retrieved results.

→

Global (user / result - independent)

- automatic thesaurus gener^n.
  eg - fast = rapid
  four = boundary ; cricket
  sound = noise.
  tall = height
- how to improve relevance?
- slangs?
  ↳ brother ≈ bro
- we can do co-occurrence analysis for the doc.

eg - mainframes (doc)

primarily, we are ....

mainframe (doc)

usually, we refer

primarily & usually are used like synonyms.

eg - term incident matrix :-

words, books	Book 1	B 2	B3	B4
w1	100	1	110	2
w2	50	9	47	8
w3	25	3	24	4
w4	75	6	73	7

Book 1 & 3, Book 2 & 4 are similar.

- So, we can do co-occurence analysis
- 2 terms are similar if term vectors are similar.
- Context is important
  - same video may not be relevant on weekend with but relevant on week.

eg - term-doc matrix

term-doc matrix . term doc matric (transpose)
= term-term matrix .

from term-term matrix, we can see results :-

	t1	t2	t3	t4	t5	t6
t1	3	1	2	1	2	3
t2	1					1
t3	2					2
t4	1					1
t5	2					2
t6	3	1	2	1	2	3

similar

so, t1 ≈ t6

→ Precision & Recall

- Precision - fract^n of retrieved docs. are relevant.

$$P\left(\frac{\text{relevant}}{\text{retrieved}}\right)$$

- Recall - fract^n of relevant doc that are retrieved

$$P\left(\frac{\text{retrieved}}{\text{relevant}}\right)$$

	relevant	non relevant	
retrieved	tp	fp	→ false +ve
not retrieved	fn	tn	→ false -ve
	→ false -ve		→ confusion matrix

→ true +ve

- Precision P = $\frac{tp}{tp+fp}$ → correct predict^n / → all retrieved docs

- Recall = $\frac{tp}{tp+fn}$ → correctly predicted / → total relevant in corpus

- There has to be balance b/w precision & recall
- If we try to increase one, other may decrease
  eg - we retrieve more docs for more recall, but precis^n may decrease, & vice versa
- Diff. b/w recall & precision must not be more than 5%.
  eg - if precis^n = 95%, recall = 10%, then diff. > 5% so bad

- For recommender systems,

	Item1	i2	i3	in
User1				
U2				
U3				
:				
Un				

→ we try to predict each cell rating.

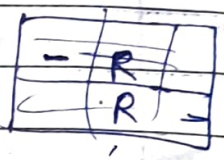
- Then we calculate error by $\sum$(actual rating − predicted rating)^2 (Root mean square error)
- This is our accuracy parameters.

eg- IRS retrieved 20 docs.
- There are 100 relevant in corpus.
8 are correct & relevant

Similarly 20 cells

$\therefore$ Precision = $\frac{8}{20}$

Recall = $\frac{8}{100}$



→ here all the R are correctly recommended

eg- Retrieved docs → RRNNNNNNRNRNNNR NNNNR

precision = $\frac{6}{20}$

- let 8 relevant docs (total)

$\therefore$ recall = $\frac{6}{8}$

Page No
Date :

→ F measure

. Takes both precisn & recall into account

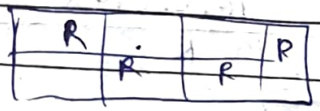
$\therefore F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}+\frac{1}{P}}$

Q - why F measures take harmonic mean? (not arith. or geometric)?

A - arithmetic - biased towards high values. (tail)

HW geometric mean - ?

→ eg -



case 1

case 2

↳ ~~Both have~~ same precisn & recall, but case 2 is better, as 1 to 8 are ranked, and case 2 correctly retrieves ranking wise.

relevant

→ solution - precision at Kth index

→

retrieved
→ relevant

$P@1 = \frac{0}{1} = 0$ ← At 1st index / location

$P@2 = \frac{1}{2}$ ← At 2nd index

$P@3 = \frac{2}{3}$

$P@4 = \frac{2}{4}$

→ Interpolated precision

cutoff at kth relevance level.

1st R.

$P@1 = \frac{1}{2}$

1st location where R appears.

2nd R.

$P@2 = \frac{2}{3}$

avg. precision = $\frac{0.5 + 0.66}{2}$

↳ eg- if total 5 relevant, then 5 levels.
1st location = where first R appears, similarly 2nd location where 2nd R appears & likewise.

★ case 1 → TO calculate avg precision

avg-precision = $\frac{\frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2}}{5}$

Page No.
Date :

∴ If 10 docs,

$$avg = \frac{\frac{1}{2} + \frac{1}{2} + \ldots \text{ 10 times}}{10}$$

case 2 → For Avg. precsn (AP) at level 4,

$$\frac{P@1 + P@2 + P@3 + P@4}{4}.$$

If multiple queries are given we can average MAP of the 2 queries

→ MAP (mean avg. precision) is important - calculates precsn for all queries

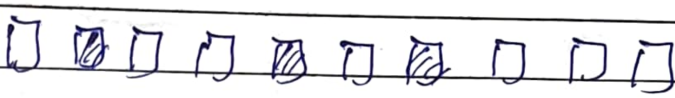
query -

eg - 1) 

$$\therefore \frac{P@1 + P@2 + P@3 + P@4 + P@5}{5}$$ → 5 levels of relevance

$$= \frac{\frac{1}{1} + \frac{2}{3} + \frac{3}{6} + \frac{4}{9} + \frac{5}{10}}{5}$$

$$= \frac{1 + 0.66 + 0.5 + 0.5 + 0.44}{5} = 0.62$$

query-2)  → 3 levels of relevance

$$\frac{P@1 + P@2 + P@3}{3}$$

$$= \frac{\frac{1}{2} + \frac{2}{5} + \frac{3}{7}}{3} = \frac{0.5 + 0.4 + 0.428}{3} = 0.4$$

∴ final MAP = $\frac{0.62+0.44}{2}$ = 0.53

Q- case 1:- P@1

Q- query 1 =>