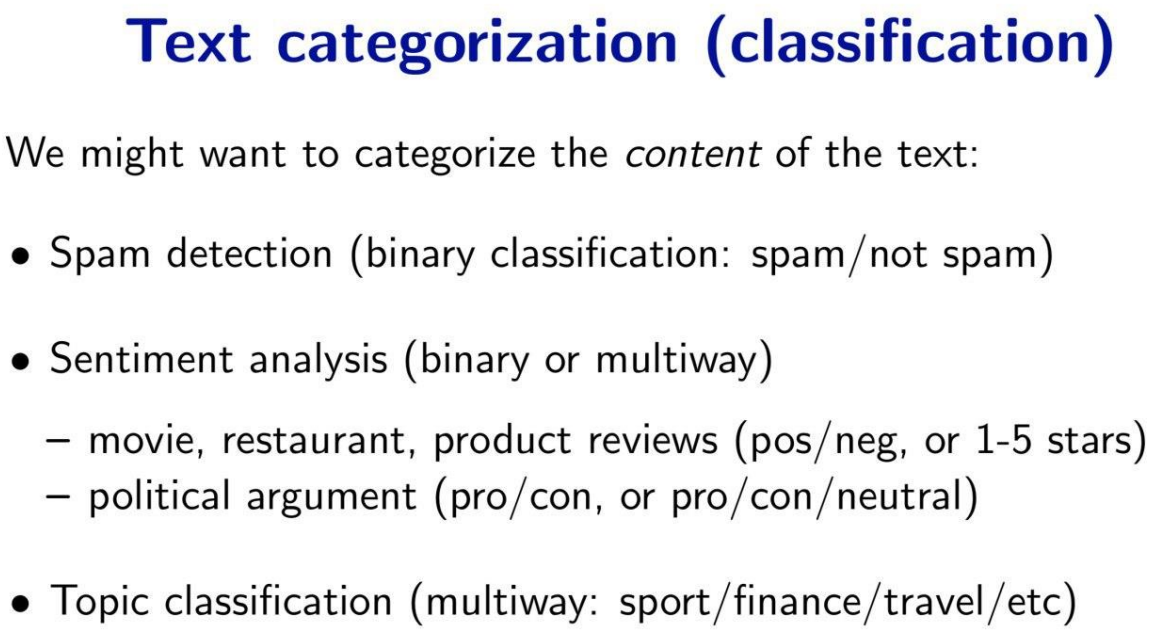
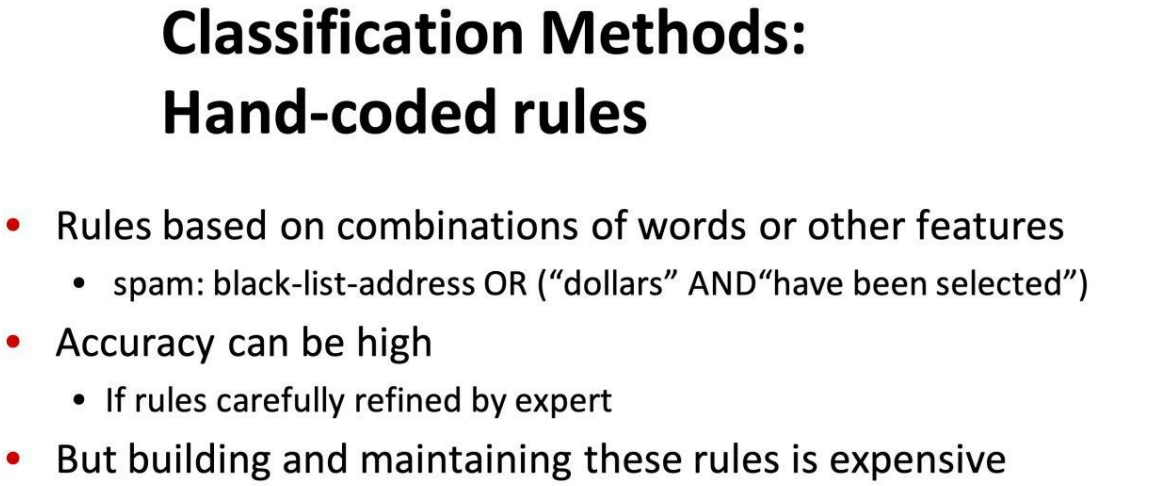
halo temen2 DSI semua. kenalin, gw kemal, research scientist di kata.ai. malam ini gw akan coba share dikit tentang text classification, materinya mungkin temen2 banyak yg udah familiar, tapi semoga tetap bermanfaat ya. Pertama2, apa sih text classification itu?



text classification itu adl sebuah task dmn kita ingin mengkategorikan/mengklasifikasikan teks berdasarkan (biasanya) kontennya. contohnya spt di atas: deteksi spam, analisis sentimen, atau klasifikasi topik, atau kalo konteksnya chatbot seperti di kata, klasifikasi intent. pesan dari user ini intent-nya apa ya? nanya tentang produk kah? mau beli tiket pesawat kah? Dll oke semoga jelas ya text classification itu ngapain, sekarang, gimana caranya?



salah satu cara yg paling sederhana adalah pakai hand-coded rules

contohnya di atas, utk deteksi spam, kalau alamat pengirim itu ada dlm blacklist, atau kontennya mengandung kata "dollar" atau "have been selected" (misal, "congrats you have been selected as the winner..."), kita mark emailnya sebagai spam

tapi bikin rules kyk gini (1) perlu domain expert yg ongkosnya tidak murah, dan (2) capek bikinnya kalau kita punya domain expert dan/atau resource utk bikin ini, silakan aja ga masalah. yg penting kan sistemnya berjalan sesuai kebutuhan

tapi gmn kalau ngga?

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of m hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $\gamma:d \rightarrow c$

ML to the rescue!

sebagai ganti domain expert dan resource utk bikin rules, kita sekarang perlu labeled data utk pakai ML, jadi, imo, sebaiknya beralih ke ML kalo emg cost utk ngelabelin data dirasa lebih rendah daripada cost utk mendapatkan/melatih domain expert + resource, oke sekarang ceritanya costnya ternyata emang lebih rendah (kalo lebih tinggi yaudah kulgram ini selesai aja hahaha. setelah punya labeled data, algo ML apa yg bisa dipake? Hampir semua bisa, tapi di NLP biasanya utk text classification, algo paling sederhana yg dipake adalah naive Bayes dan itulah yg mau gw bahas di kulgram ini

gimana sih naive Bayes classifier itu? namanya ada Bayes-nya karena dia pakai teorema Bayes dlm rumusannya

Formalizing the task

- Given document d and set of categories C, we want to assign d to the most probable category $\hat{c}$:

$$\begin{aligned}\hat{c} &= \operatorname*{argmax}_{c \in C} P(c|d) \\ &= \operatorname*{argmax}_{c \in C} \frac{P(d|c)P(c)}{P(d)} \\ &= \operatorname*{argmax}_{c \in C} P(d|c)P(c)\end{aligned}$$

langkah ke-2 adalah teorema Bayes

nah jadi saat training yg perlu dihitung adalah P(d|c) dan P(c)

ini kan dokumen ya, jadi dia teks yang punya banyak kata2

utk bisa hitung P(d|c), kita perlu representasikan d ini dlm bentuk yg lebih enak

Document model

Each document d is represented by **features** $f_1, f_2, \ldots f_n$, e.g.:

- For topic classification: 2000 most frequent words, excluding **stopwords** like *the*, *a*, *do*, *in*.
- For sentiment classification: words from a **sentiment lexicon**

In fact, we only care about the feature *counts*, so this is a **bag-of-words** (unigram) model.

ini salah satu caranya

edited

kita representasikan tiap dokumen/teks sebagai fitur2. contohnya kalau mau klasifikasiin topik, fitur2nya adalah misal 2000 kata yg paling sering muncul di training data kita

edited

dan kita cuma peduli dgn brp kali tiap2 kata tsb muncul di dokumen tersebut. ini namanya bag-of-words model/representation

The Bag of Words Representation

ini ilustrasi dari bag-of-words

tiap dokumen dianggap sbg "kantong berisi kata2", ga peduli gmn kata2 tsb disusun

Example documents

- Possible feature counts from training documents in a spam-detection task (where we did not exclude stopwords):

	the	your	model	cash	Viagra	class	account	orderz
doc 1	12	3	1	0	0	2	0	0
doc 2	10	4	0	4	0	0	2	0
doc 3	25	4	0	0	0	1	1	0
doc 4	14	2	0	1	3	0	1	1
doc 5	17	5	0	2	0	0	1	1

sekarang, misalnya kita punya 5 dokumen dengan fitur2 seperti ini. tiap entri pada tabel menunjukkan berapa kali kata (pada kolom) muncul dalam dokumen (pada baris) jadi, kata "the" muncul 12x dalam dokumen 1, "cash" muncul 4x dalam dokumen 2, dst

Document model, cont.

- Representing d using its features gives us:

$$P(d|c) = P(f_1, f_2, \ldots f_n|c)$$

- But we can't estimate this joint probability well (too sparse).
- So, make a **Naive Bayes** assumption: features are conditionally independent given class.

$$P(d|c) \approx P(f_1|c)P(f_2|c)\ldots P(f_n|c)$$

kalau kita langsung pakai representasi ini, berarti kan P(d|c) jadi joint probability dari fitur2 penyusun tapi joint probability ini besar banget. misal ada 10 kata, frekuensi kata maks adl 10, dan banyak kelas ada 2, berarti besarnya jadi 2x10^10. gede banget, data yg kita punya kemungkinan ga bakal cukup jadi naive Bayes mengasumsikan fitur2nya independen given class-nya. jadi P(d|c) bisa dipecah jadi kyk bentuk yg di bawah itu, tinggal perkalian dari P(f|c) aja, masing2 P(f|c) ini besarnya jadi cuma 2x10 aja.

Full model

- Given document with features $f_1, f_2, \ldots f_n$ and set of categories C, choose

$$\hat{c} = \operatorname*{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(f_i|c)$$

jadi beginilah naive Bayes classifier yg lengkap oke sekarang masuk ke gmn trainingnya biasanya cukup pakai MLE aja.

Learning the class priors

- $P(c)$ normally estimated with MLE:

$$\hat{P}(c) = \frac{N_c}{N}$$

  - N_c = the number of training documents in class c
  - N = the total number of training documents

ini cara hitung P(c), i.e. class prior prob, gampang kan tinggal hitung aja proporsi masing2 kelas

Learning the class priors: example

- Given training documents with correct labels:

	the	your	model	cash	Viagra	class	account	orderz	spam?
doc 1	12	3	1	0	0	2	0	0	-
doc 2	10	4	0	4	0	0	2	0	+
doc 3	25	4	0	0	0	1	1	0	-
doc 4	14	2	0	1	3	0	1	1	+
doc 5	17	5	0	2	0	0	1	1	+

- $\hat{P}(\text{spam}) = 3/5$

ini contohnya lanjut dari yg atas.

ini cara hitung P(f|c), idenya masih sama, tinggal hitung aja proporsi suatu kata/fitur f muncul di kelas c. Di situ tapi ada smoothing parameter alpha, tujuannya utk menghindari zero probability kalo kata/fitur f ga pernah muncul di suatu class c (tapi muncul di class lain) kalo alpha = 1, biasanya disebut add-one smoothing atau Laplace smoothing.

Learning the feature probabilities: example

	the	your	model	cash	Viagra	class	account	orderz	spam?
doc 1	12	3	1	0	0	2	0	0	-
doc 2	10	4	0	4	0	0	2	0	+
doc 3	25	4	0	0	0	1	1	0	-
doc 4	14	2	0	1	3	0	1	1	+
doc 5	17	5	0	2	0	0	1	1	+

$$\hat{P}(\text{your}|+) = \frac{(4+2+5+\alpha)}{(\text{all words in } + \text{ class})+\alpha F} = (11+\alpha)/(68+\alpha F)$$
$$\hat{P}(\text{your}|-) = \frac{(3+4+\alpha)}{(\text{all words in } - \text{ class})+\alpha F} = (7+\alpha)/(49+\alpha F)$$
$$\hat{P}(\text{orderz}|+) = \frac{(2+\alpha)}{(\text{all words in } + \text{ class})+\alpha F} = (2+\alpha)/(68+\alpha F)$$

ini contoh cara hitungnya, lanjutan dari yg di atas,

Classifying a test document: example

- Test document d:

  get your cash and your orderz

- Suppose there are no other features besides those in previous table (so get and and are not counted). Then

$$\begin{aligned} P(+|d) &\propto P(+)\prod_{i=1}^{n} P(f_i|+) \\ &= P(+)\cdot\frac{11+\alpha}{(68+\alpha F)}\cdot\frac{7+\alpha}{(68+\alpha F)} \\ &\quad\cdot\frac{11+\alpha}{(68+\alpha F)}\cdot\frac{2+\alpha}{(68+\alpha F)} \end{aligned}$$

lalu, ini contoh gmn caranya ngitung probability dokumen baru (mis. dari test data) setelah classifier di-train, tinggal kali2in aja semua class prior dan prob dari fitur2nya, perhatikan bahwa krn "get" dan "and" gak ada dlm fitur, jadi mereka dilewatin aja ga dihitung. Nanti utk dapetin class dari teksnya, tinggal ambil aja class yg probability-nya paling besar.

Advantages of Naive Bayes

- Very easy to implement
- Very fast to train and test
- Doesn't require as much training data as some other methods
- Usually works reasonably well

This should be your baseline method for any classification task

ini beberapa keunggulan dari naive Bayes, tapi naive Bayes juga punya bbrp kekurangan.

Non-independent features

- Features are not usually independent given the class
- Adding multiple feature types (e.g., words and morphemes) often leads to even stronger correlations between features
- Accuracy of classifier can sometimes still be ok, but it will be highly **overconfident** in its decisions.
  - Ex: NB sees 5 features that all point to class 1, treats them as five independent sources of evidence.
  - Like asking 5 friends for an opinion when some got theirs from each other.

ini salah satunyam, intinya: asumsi fitur2 independen bikin naive Bayes suka kepedean, jadi jgn terlalu dipercaya nilai probability-nya.

Sesi pertanyaan

Dari Nur Khotimah :

Misal ada kata yang pakai imbuhan dipisahkan nya gimana ya kak? *Kata dalam bahasa Indonesia?

Jawaban :

terus mau "-an" dipisah dari "bagusan" gitu ya?

"-an"-nya mau tetap ada di dokumen jadi "bagus an merek x" atau hilang aja jadi "bagus merek x"? oke. kalo gitu, salah satu yg bisa dilakukan adalah melakukan stemming/lemmatization pada teks sebelum masuk ke naive Bayes classifier-nya. Lemmatization adl proses mengubah suatu kata jadi lemma-nya (kata dasar dalam kamus). Ini task berbeda lagi dari text classification. gw kurang tau apakah ada lemmatizer bhs indonesia yg siap pakai. mungkin temen2 DSI ada yg lebih tau dan bisa merekomendasikan.

Dari Raudhoh Fitra Humamy :

Kak, nanya dong...
kalau pake bag of words, apakah semua kata yang muncul (kecuali stopwords) akan kita manfaatkan sebagai feature atau cuma sejumlah kata dengan frekuensi kemunculan tertinggi? Misalnya cuma 500 kata dengan frekuensi kemunculan tertinggi. Nah, angka 500 ini apakah ada rule nya atau intuitif saja?

Jawaban :

semuanya boleh2 aja. mau semua kata jadi fitur boleh, mau 500 most frequent words boleh. apakah ada rule-nya? setau gw sih ngga ya, tapi kita bisa anggap angka threshold ini sebagai hyperparameter yg nilainya kita tune ke validation set.

perhatikan bahwa dengan naikin/nurunin angka threshold ini, kita essentially mainin bias-variance trade-off. kalo fiturnya dikit modelnya underfit, kalo fiturnya banyak modelnya kemungkinan overfit. jadi boleh angka itu dianggap sbg hyperparameter.

hal yg lazim dilakukan adalah ngatur threshold frekuensi kemunculan fitur. jadi yg masuk sebagai fitur cuma kata2 muncul lebih dari misal 5 kali. intinya sama aja dgn ngebatesin banyak fitur (kyk tadi 500 kata paling sering muncul). dan ini juga masuk akal scr statistik krn kata2 yg jarang muncul ini kemungkinan kita ga bisa estimate probability-nya dgn reliable krn sampelnya terlalu sedikit.

Dari Raudhoh Fitra Humamy :

Kalo untuk conversational text, ideal ga sih make Naive Bayes?

Jawaban :

boleh2 aja. selama akurasi dan speed-nya sesuai dgn harapan, algo apa aja sah (menurut gw).

Dari indra :

Kalo text classification sekarang ini ada state of the art method nya ga ya? Atau trend nya gmn?

Jawaban :

Krn text classification itu luas banget ya tergantung mau dipakai buat apa, jadi agak susah jawabnya. tapi sekilas liat utk sentiment analysis aja, neural network masih merajai sih ndra
https://nlpprogress.com/english/sentiment_analysis.html

Dari Fakhri Kurniawan:

Mau nanya dong, kalo utk membuat Sentiment tapi beda2 temanya itu dataset trainingnya harus dibuat ulang atau sekarang ada 1 dataset yg universal? (Khusus utk NLP B. Indonesia)

Jawaban :

wah maaf gw kurang tau kalo ada dataset sentiment bhs indonesia yg tersedia utk publik. temen2 DSI yg lain mungkin ada yg lebih tau.

Dari admaja :

kalau dari set seperti ini apa mungkin kita bisa cari persamaan tiap dokument nya, berdasarkan isi dari dokumen tersebut terus di bandingkan dengan dokumen lain?

Jawaban :

Menurut gw bisa. salah satu cara paling gampang adalah tinggal hitung aja cosine similarity dari 2 dokumen tsb. tiap dokumen kan udah jadi vektor tuh, jadi mungkin dilakukan.