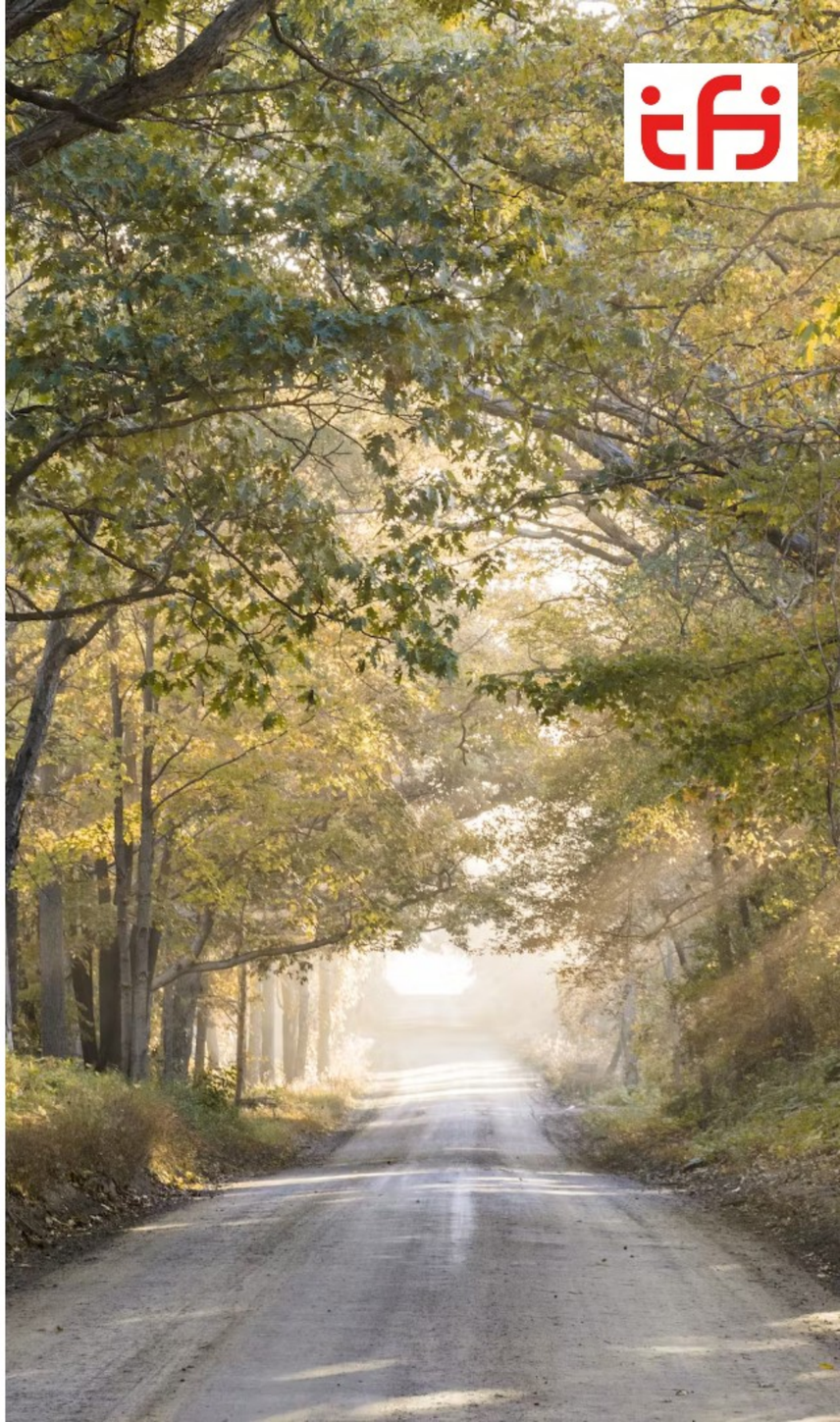
IN[34]120 Søketeknologi - Collections, relevance, ranked retriveal

2023-09-29 10:15 @ Chill

Repetisjon:
- Zipf's law
- Heap's law
- TFIDF
- Cosine similarity

Etter:
- Oblighjelp for oblig B



Gruppetimen 13.10

- → Om 2 uker
- → Sergey fra gruppe 2 tar over
- → Ta ham godt i mot!

Collections-'lovene'

Eksamensrelevant!!!

Heaps' law

- Typer, tokens
- Husk: "typer" unike ord, "tokens" er alle ord
- "type-token relation"
- 'Dersom du har så mange tokens, har du omtrent så mange typer'

Zipf's law

- Kvantitativ lingvistikk
- Sier noe om hvor ofte et gitt ord forekommer i et corpus
- Inverse proportionality
- Det mest frekvente ordet forekommer dobbelt så ofte som ord #2, 3 ganger så ofte som ord #3 osv
- Stoppord, 'the'

Søket vårt til nå

- Binært - enten match eller ikke
- Vet ikke hvor bra match
- Hvordan få til noe bedre?

Ranked retrieval

- → Sorter matches basert på "brahet"
- → Brahet = fx hvor bra et document matcher queryen
- → Nyttig feature

Document similarity (Relevance)

- → Tf-idf
- → Cosine similarity
- → Hvorfor? Ranked retrieval

Tf-idf

- "Term frequecy-inverted document frequency"
- NB: Streken er ikke 'minus'
- Anbefaler: https://ted-mei.medium.com/demystify-tf-idf-in-indexing-and-ranking-5c3ae88c3fa0

Demystify TF-IDF in Indexing and Ranking

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

(Stjålet fra linken i forrige foil)

Cosine similarity

- Bullet 1
- Bullet 2
- Bullet 3

Diskusjosoppgave coming up

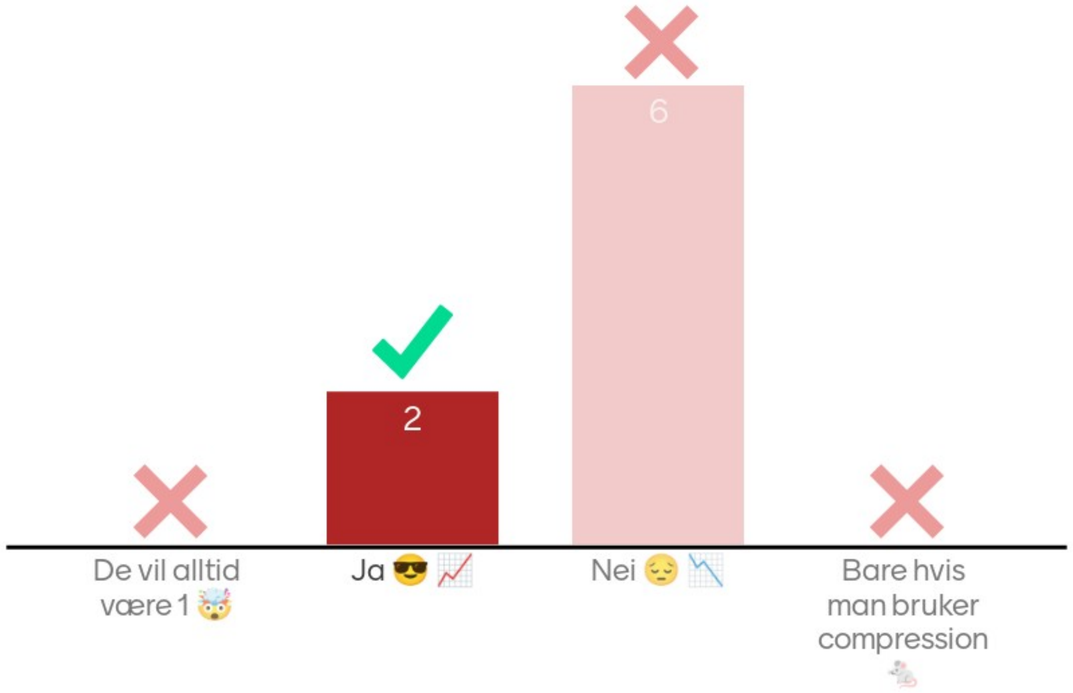
Om oblig B-pensum, spesielt papers

(b) [6%] You do not want to suggest queries to the user that could be perceived as somehow offensive. To this end, you have a big dictionary with about 250,000 offensive words and phrases, and queries that contain at least one of these dictionary entries should not end up in the data material used for suggestions. Explain how you would match the logged queries against your dictionary as efficiently as possible, and where/how you would inject this logic into your MapReduce job.

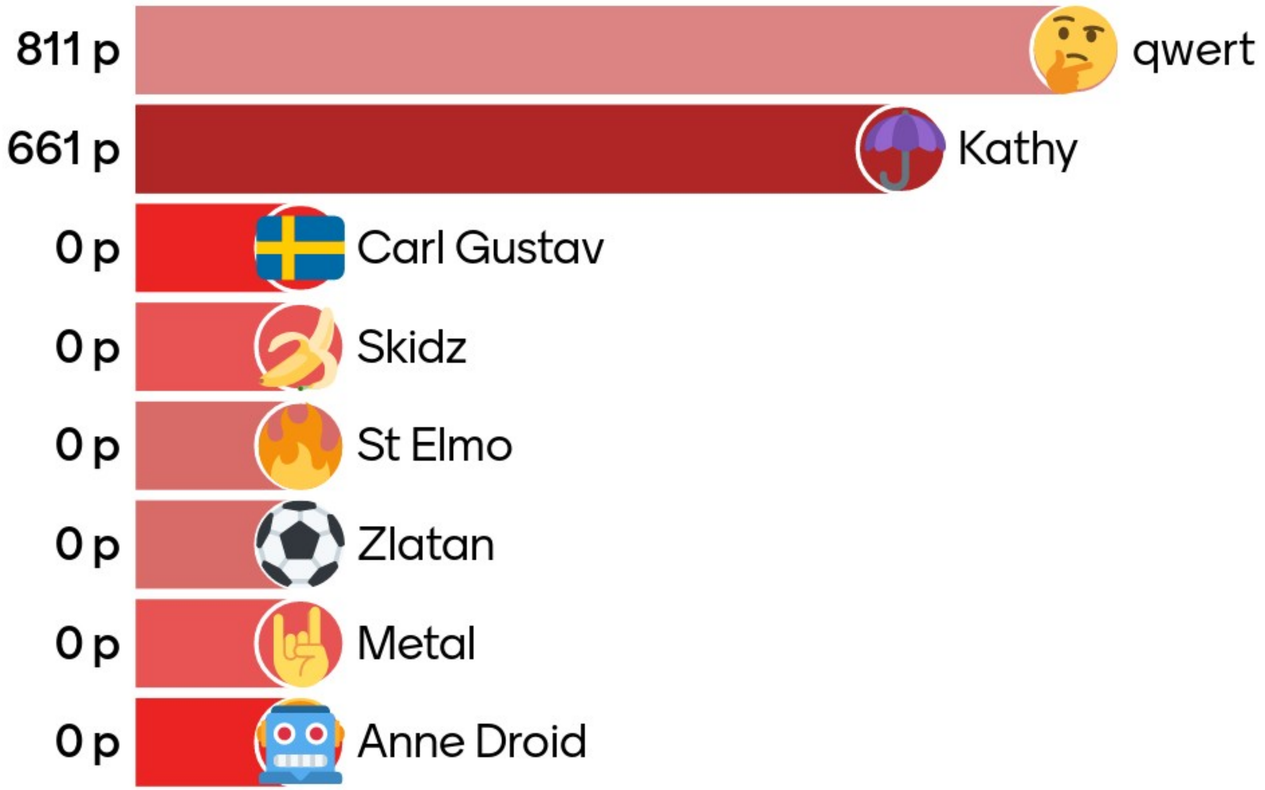
Given the overall query volume and that suggestions need to be generated per keypress, you aim for an as-efficient-as-possible in-memory solution for serving the suggestions.

Noen tanker om hvordan man kan få til dette? (Fra eksamen 2021)

Kan en tf-idf-verdi overstige 1? (Eks 2020)



Leaderboard



Om en tf-idf-verdi kan overstige 1:

- → Ja!
- → Log-funksjonen vokser monotont til infinity
- → Kan også justifyes med et eksempel

Oblig B

- → Frist i dag
- → Oblighjelp resten av tiden
- → (Utsettelse: 3 dager)

Skriv noe og det vil bli snakket om

Anonymt alternativ til 1-1-hjelp