INFO221v12
IR V

Truls Pedersen
Institutt for informasjons- og medievitenskap
Universitetet i Bergen

Oversikt

- `(Fast)CosineScore`
- Heuristikk
- de beste $\pm$
- Huristiske metoder
- IR system
- Kvalitetsvurdering av IR system
- Testdata
- Urangerte resultatmengder

Vektorer

Cosinus poeng er gitt ved

$$sim(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|} = \cos(\theta)$$

hvor

$$\vec{x} \cdot \vec{y} = \sum_i x_i y_i$$

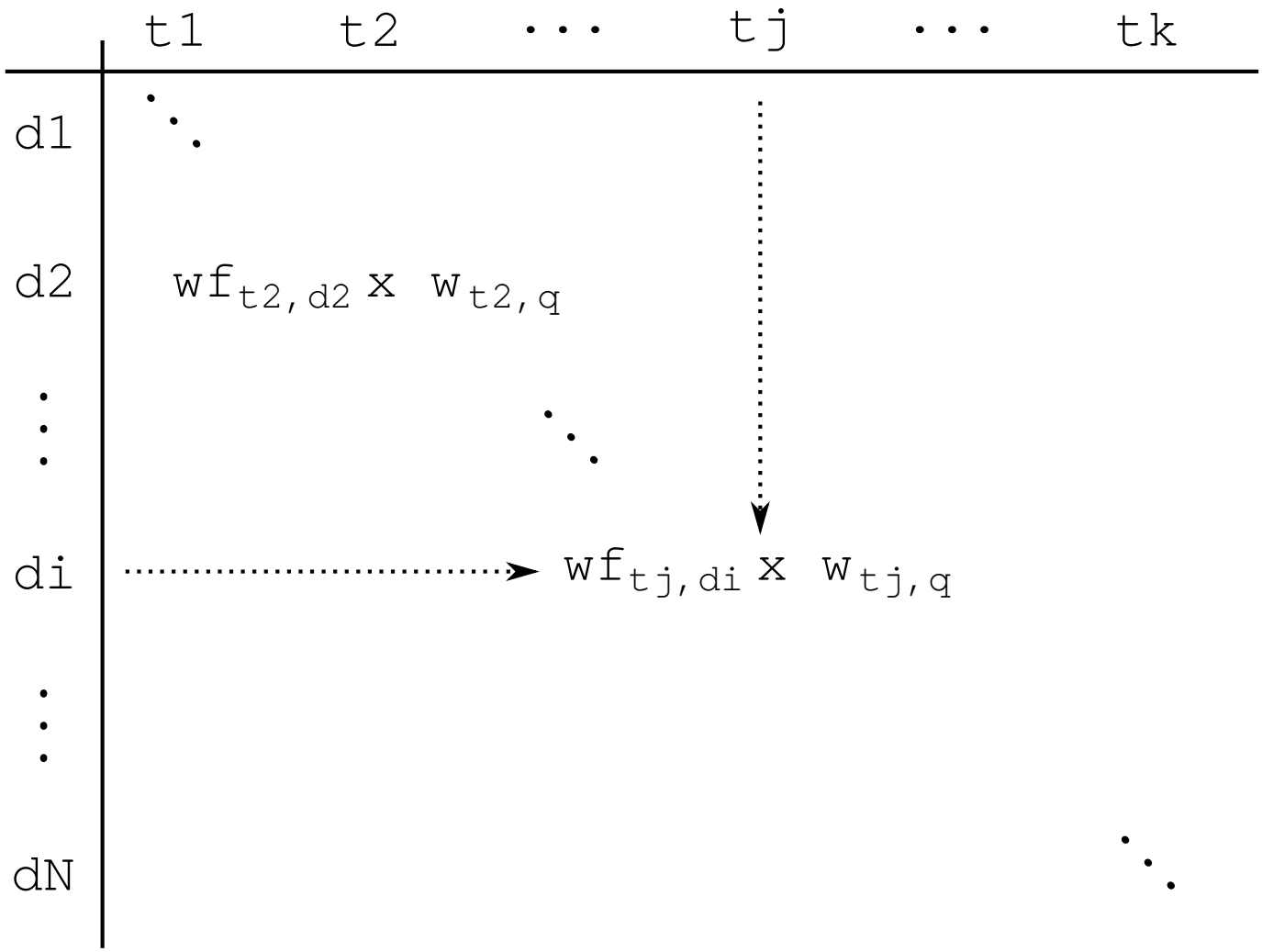
og

$$\vec{V}(d) = (w_{d,t_1}, w_{d,t_2}, \ldots, w_{d,t_k})$$

F.eks.

$$w_{d,t_1} = \text{tf-idf}_{d,t_1}$$

CosineScore



CosineScore

1. float `Scores[N] = 0`
2. Klargjør `Length[N]`
3. **for each** sprørreterm `t`
4. **do** beregn $w_{t,q}$ og hent treffene for `t`
5. **for each** $(d, tf_{t,d})_t$
6. **do** `Scores[d]` += $wf_{t,d} \times w_{t,q}$
7. Les inn `Length[d]`
8. **for each** d
9. **do** `Scores[d] = Scores[d] / Length[d]`
10. **return** beste K komponentene i `Scores[]`

CosineScore

1. float `Scores[N] = 0`
2. Klargjør `Length[N]`
3. **for each** sprørreterm `t`
4. **do** beregn $w_{t,q}$ og hent treffene for `t`
5. **for each** $(d, tf_{t,d})_t$
6. **do** `Scores[d]` += $wf_{t,d} \times w_{t,q}$
7. Les inn `Length[d]`
8. **for each** d
9. **do** `Scores[d] = Scores[d] / Length[d]`
10. **return** beste K komponentene i `Scores[]`

CosineScore

1. float `Scores[N] = 0`
2. Klargjør `Length[N]`
3. **for each** sprørreterm `t`
4. **do** hent treffene for `t`
5. **for each** $(d, tf_{t,d})_t$
6. **do** `Scores[d]` += $wf_{t,d}$
7. Les inn `Length[d]`
8. **for each** d
9. **do** `Scores[d] = Scores[d] / Length[d]`
10. **return** beste K komponentene i `Scores[]`

CosineScore

1. float `Scores[N] = 0`
2. Klargjør `Length[N]`
3. **for each** sprørreterm `t`
4. **do** hent treffene for `t`
5. **for each** $(d, tf_{t,d})_t$
6. **do** `Scores[d]` += $wf_{t,d}$
7. Les inn `Length[d]`
8. **for each** d
9. **do** `Scores[d] = Scores[d] / Length[d]`
10. **return** beste K komponentene i `Scores[]`

Heuristikk

Hvor viktig er det at vi returnerer de K dokumentene med mest poeng?

Heuristikk

Hvor viktig er det at vi returnerer de K dokumentene med mest poeng?

Vi vet at poengene vi tildeler et dokument *anslag* av relevans.

Vet også at tallverdiene til poengene er *vilkårlige*.

De “beste” resultatene

Vi skal se på noen alternative løsninger.

Fremgangsmåtene har alle samme hovedstruktur:

1. Finn en mengde dokumenter A
   Vi ønsker å finne en mengde A slik at
   - $K < |A| << N$, og
   - så mange “gode” resultater er i A som mulig

2. Returner de beste K resultatene fra A

De “beste” resultatene

Vi skal se på noen alternative løsninger.

Fremgangsmåtene har alle samme hovedstruktur:

1. Finn en mengde dokumenter A
   Vi ønsker å finne en mengde A slik at
   - $K < |A| << N$, og
   - så mange “gode” resultater er i A som mulig

2. Returner de beste K resultatene fra A

En umiddelbar reduksjon er å se bort fra all dokumenter som ikke inneholder minst en av søketermene.

Indekseliminasjon

1. Vi kan se bort fra termer med lav *idf*

   Hvis vi har fire søketermer: ``catcher in the rye'', vil ikke ``in'' og ``the'' bidra på en meningsfull måte, og vi kan like gjerne søke etter ``catcher rye''.

Indekseliminasjon

1. Vi kan se bort fra termer med lav *idf*

   Hvis vi har fire søketermer: ``catcher in the rye'', vil ikke ``in'' og ``the'' bidra på en meningsfull måte, og vi kan like gjerne søke etter ``catcher rye''.

2. Vi kan styrke kravet om at resultatene må inneholde minst en søketerm, til å kreve at de må inneholde *alle* søketermene (evt. *mange*)

Topplister

Vi kan konstruere en toppliste for hver søketerm.

Topplisten for en term t er de r dokumentene med høyest tf-poeng, L_t.

For en spørring som består av flere søketermer $t_1, \ldots, t_n$, kan vi søke gjennom unionen

$$A = L_{t_1} \cup \ldots \cup L_{t_n}$$

Topplister

Vi kan konstruere en toppliste for hver søketerm.

Topplisten for en term t er de r dokumentene med høyest tf-poeng, L_t.

For en spørring som består av flere søketermer $t_1, \ldots, t_n$, kan vi søke gjennom unionen

$$A \quad = \quad L_{t_1} \cup \ldots \cup L_{t_n}$$

Hvor stor må r være?

Hva om K ikke er kjent i forkant?

Kan det være rimelig å la r være avhenging av termen?

Kvalitetsvurderinger

I mange kontekster kan vi anslå kvaliteten til et dokument og gi det en poengsum.

F.eks. hvor mange "thumbs-up" denne artikkelen har fått, hvor mange web-sider som linker til denne siden, ...

$$\begin{aligned} poeng(q,d) &= g(d) + sim(q,d) \\ &= g(d) + \frac{\vec{V}(q) \cdot \vec{V}(d)}{|\vec{V}(q)||\vec{V}(d)|} \end{aligned}$$

hvor $g(d) \in [0,1]$ er kvalitetsmålet vårt.

Betydningsordning *(Impact ordering)*

Vi har sett at vi kan ta snittet av flere lister i lineær tid hvis disse er ordnet.

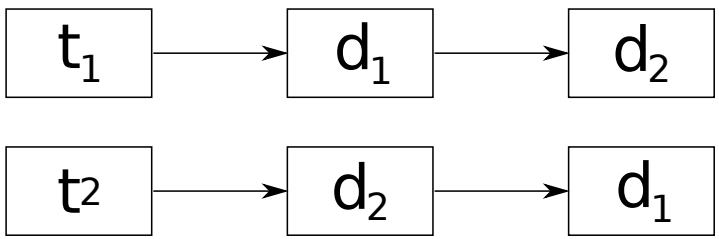
For hver søketerm har vi en liste med dokumenter

$$(t_i, [d_{j_1}, \ldots, d_{j_k}])$$

vi har tidligere sortert listen med en felles *global* tallverdi.

Nå ordner vi listen, for hver term t, synkende etter $\mathrm{tf}_{t,d}$.

Her kan det forekomme at



Betydningsordning *(Impact ordering)*

1. Når vi beregner poengene kan vi slutte å regne poeng før vi har gått gjennom hele listen
   - etter vi har beregnet et tilstrekkelig antall r, eller
   - etter poengene bidrar mindre enn t poeng.

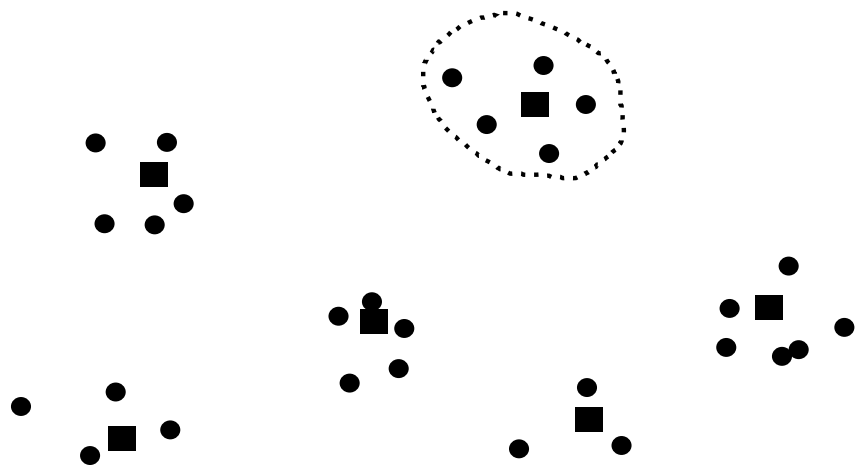
2. Vi kan ordne søketermene etter synkende *idf* og stoppe poengberegningen etter *idf*-verdien har falt under en terskelverdi

Klasetrimming *(Cluster pruning)*

Vi kan redusere antall dokumenter vi trenger å vurdere fra N til $\sqrt{N}$ ved å dele inn søkerommet inn i $\sqrt{N}$ klaser.

1. Velg $\sqrt{N}$ tilfeldige dokumenter. Disse kalles *ledere*.
2. For hver av de resterende dokumentene (*disipler*), finn den nærmeste lederen.

En leder og alle dens disiples utgjør en *klase*.

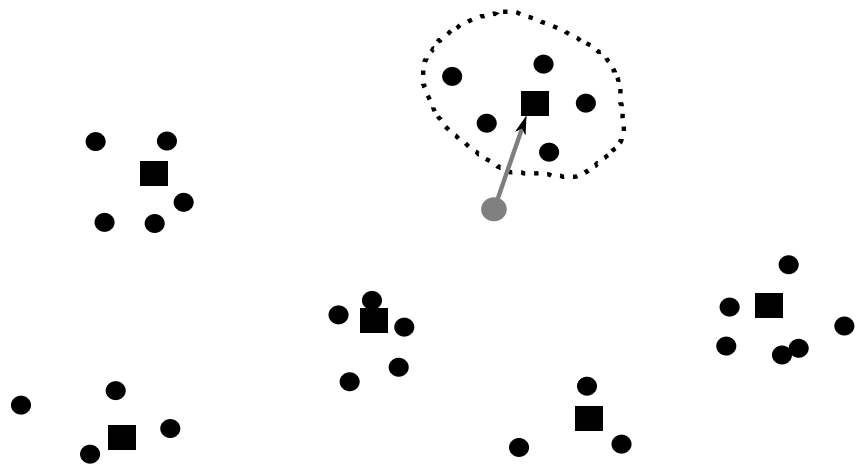


Klasetrimming *(Cluster pruning)*

Vi kan redusere antall dokumenter vi trenger å vurdere fra N til $\sqrt{N}$ ved å dele inn søkerommet inn i $\sqrt{N}$ klaser.

1. Velg $\sqrt{N}$ tilfeldige dokumenter. Disse kalles *ledere*.
2. For hver av de resterende dokumentene (*disipler*), finn den nærmeste lederen.

En leder og alle dens disiples utgjør en *klase*.



Vi søker bare gjennom den *nærmeste klasen*.

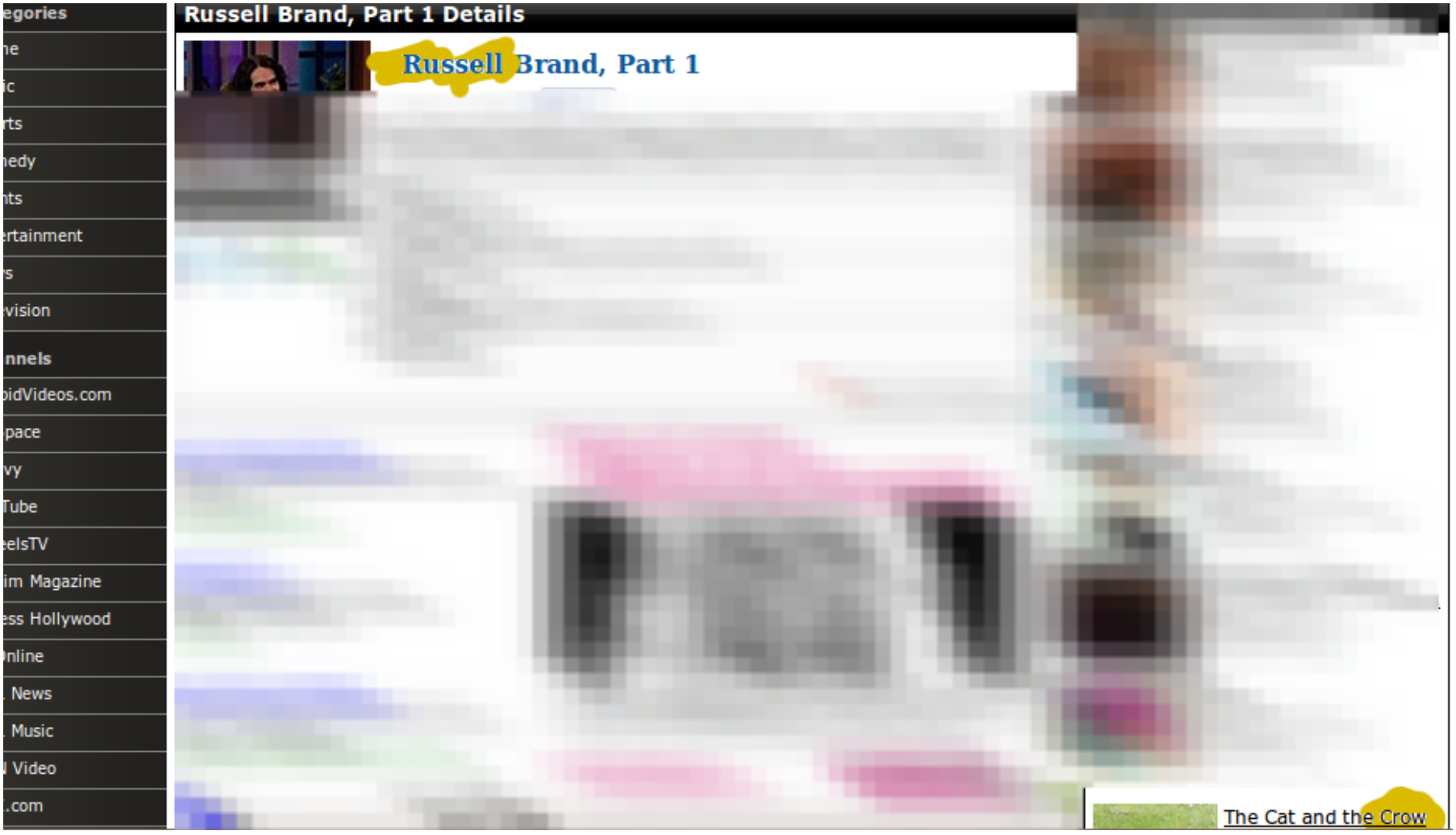
Nivåinndeling *(Tiered indexes)*

En annen måte å redusere antall beregninger på er å dele inn trefflistene våre i nivåer.

Vi har en indeks per nivå.

Hvis vi ikke får nok treff på nivå 1 (f.eks ≥ 20 tf-poeng), søker vi gjennom nivå 2 (f.eks $20 > tf \geq 10$), osv...

Term-tetthet

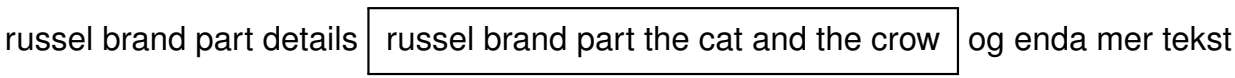


Term-tetthet

Vi kan ganske enkelt dømme dette resultatet som et dårlig resultat for søket ``Russell Crow''.

Når vi parser dokumentet kan vi lage en liste med alle ordene.

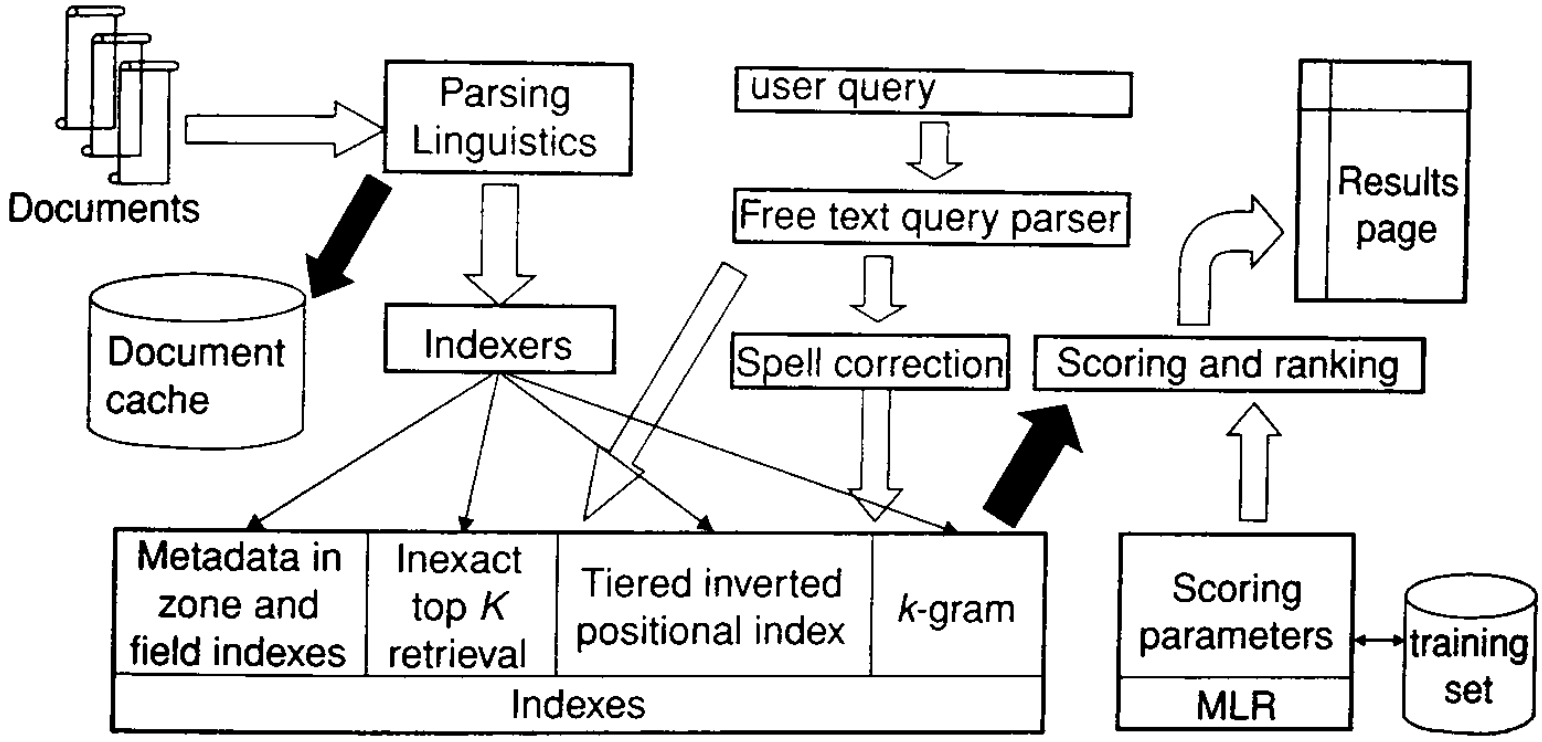
Vi kan beregne det minste vinduet som inneholder alle søketermene:



Dette vinduet er åtte ord bredt.

Mindre vindu indikerer at dokumentet er nærmere spørringen.

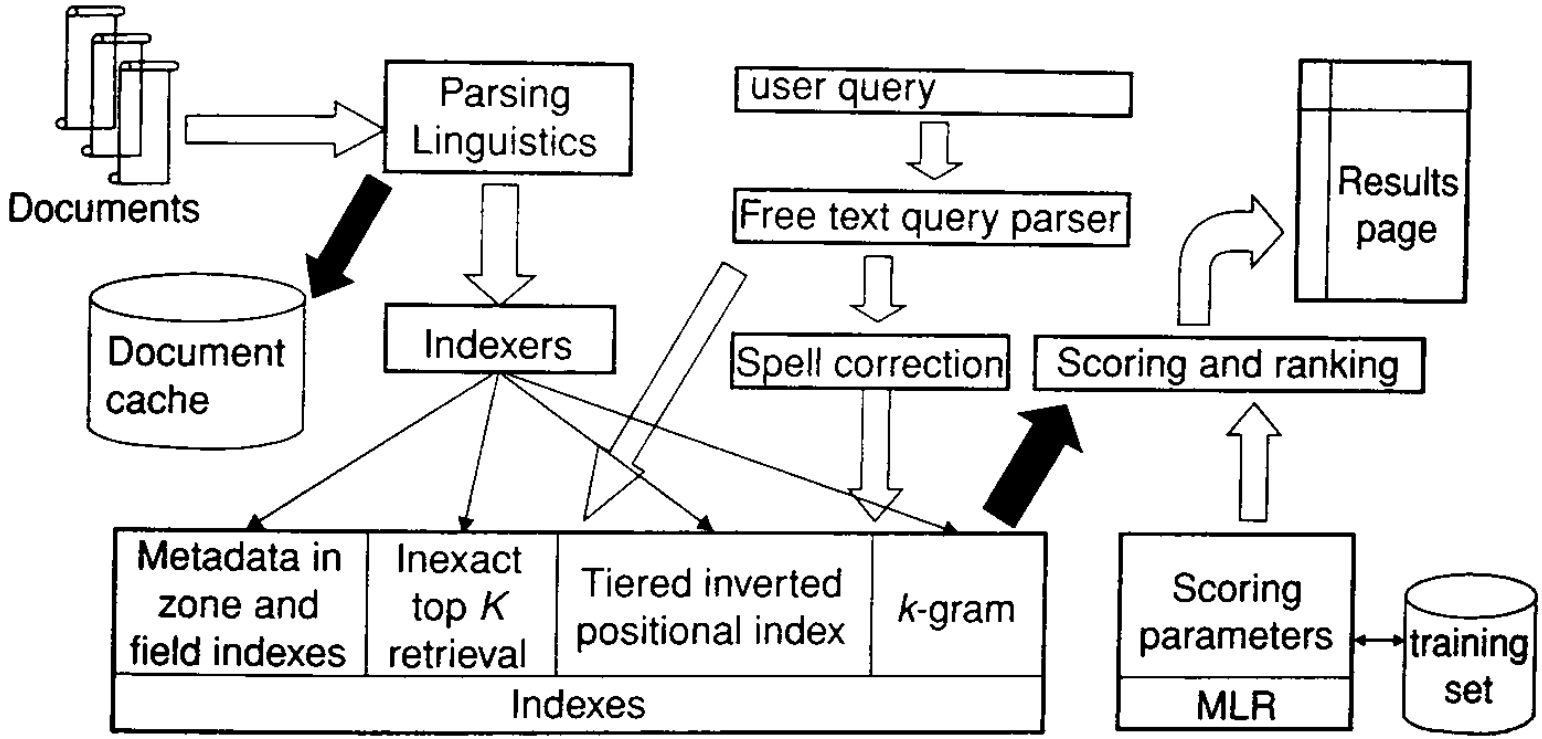
Putting it all together



Forhåndsbehandling dokumenter inn: parsing og lingvistisk behandling

Behandlet tekst i) lagre kopi, og ii) indeksering

Putting it all together



Søking søket parses og genererer en spørring

Læring ranger resultatene (basert på trening)

Presentasjon generer en presentasjon av resultatene

IR systemer

Vi har sett på mange av detaljene som inngår i et IR system.

Dette har avdekket mange variabler som har en innvirkning på hvilke resultater systemet returnerer for et søk.

IR systemer

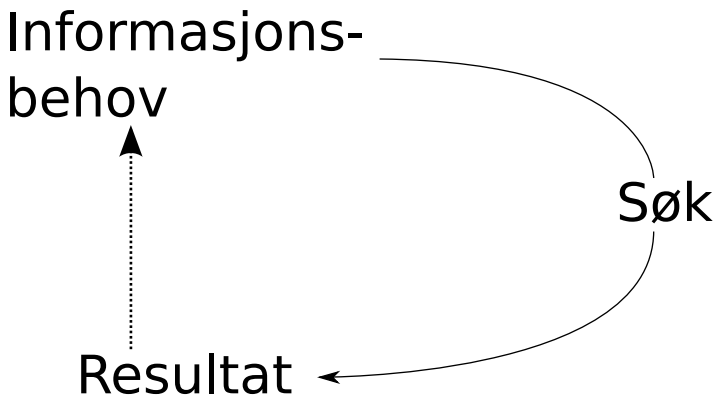
Vi har sett på mange av detaljene som inngår i et IR system.

Dette har avdekket mange variabler som har en innvirkning på hvilke resultater systemet returnerer for et søk.

Avhengig av

1. hvilke metoder/algoritmer,
2. hvilke verdier vi gir parametrene, og
3. hvilke data vi søker gjennom.

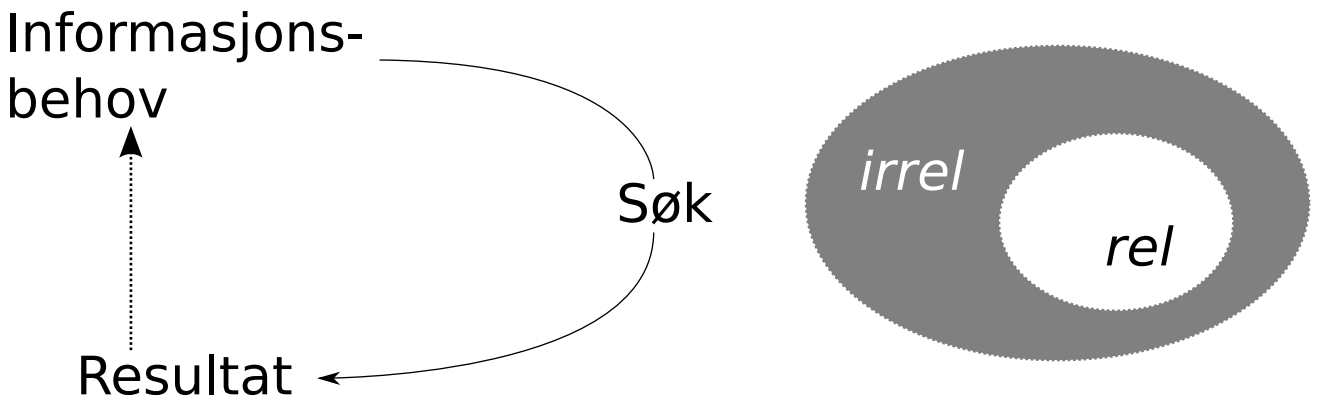
IR systemer



Vi utfører et søk for å møte et *informasjonsbehov*.

Et resultat er *relevant* hvis, og bare hvis, det svarer til *informasjonsbehovet*, og er ikke avhengig av hvorvidt det er et "rimelig" resultat for et søk.

IR systemer

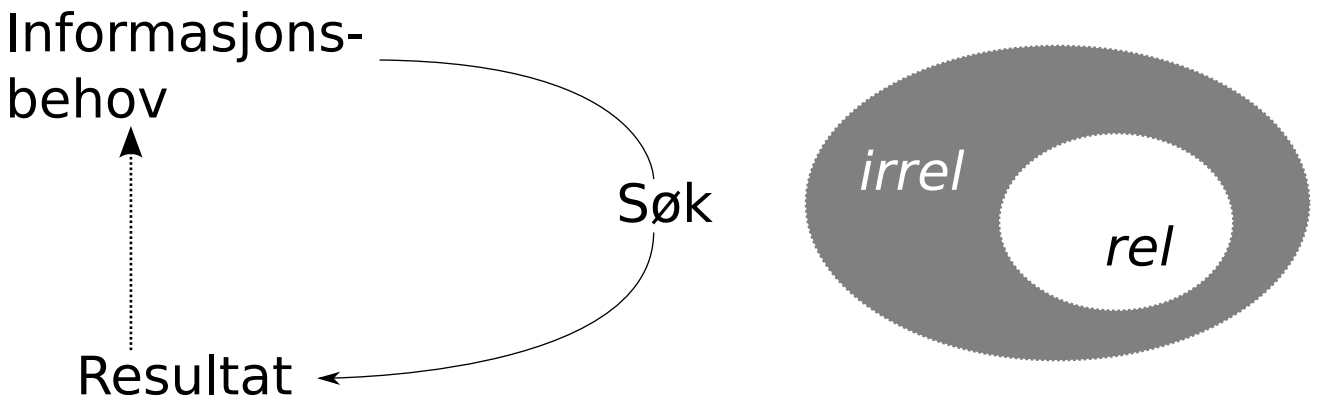


Vi utfører et søk for å møte et *informasjonsbehov*.

Et resultat er *relevant* hvis, og bare hvis, det svarer til *informasjonsbehovet*, og er ikke avhengig av hvorvidt det er et "rimelig" resultat for et søk.

Gitt et informasjonsbehov er en del av datamengden relevant.

IR systemer



Vi utfører et søk for å møte et *informasjonsbehov*.

Et resultat er *relevant* hvis, og bare hvis, det svarer til *informasjonsbehovet*, og er ikke avhengig av hvorvidt det er et "rimelig" resultat for et søk.

Gitt et informasjonsbehov er en del av datamengden relevant.

Det er selvsagt ikke så enkelt. I de fleste tilfeller er et dokument relevant til en viss grad, eller for noen.

IR systemer

Når vi klargjør testdata må vi

- velge ut en “liten” mengde dokumenter,
- definere noen informasjonsbehov,
- og klassifisere hvilke av dataene som er relevant for hvert av informasjonsbehovene.

IR systemer

Når vi klargjør testdata må vi

- velge ut en “liten” mengde dokumenter,
- definere noen informasjonsbehov,
- og klassifisere hvilke av dataene som er relevant for hvert av informasjonsbehovene.

Hypotesen er at hvis IR systemet fungerer bra for testdataene, vil det også fungere bra for hele datamengden.

Testdata

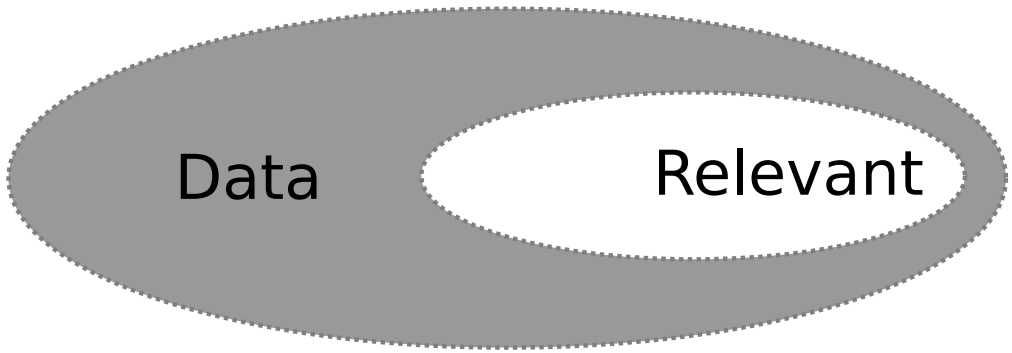
Det finnes flere datamengden tilgjengelig på internett som kan lastes ned.

Det følger vanligvis ikke med relevansvurderinger, så dette må gjøres i tillegg.

Datamengden som er utlevert er samlet inn av Øyvind vha. en *crawler*.

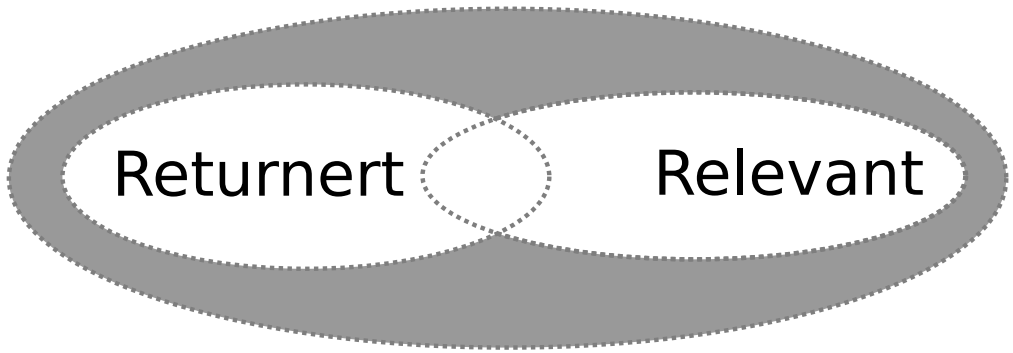
Urangerte resultatmengder

For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



Urangerte resultatmengder

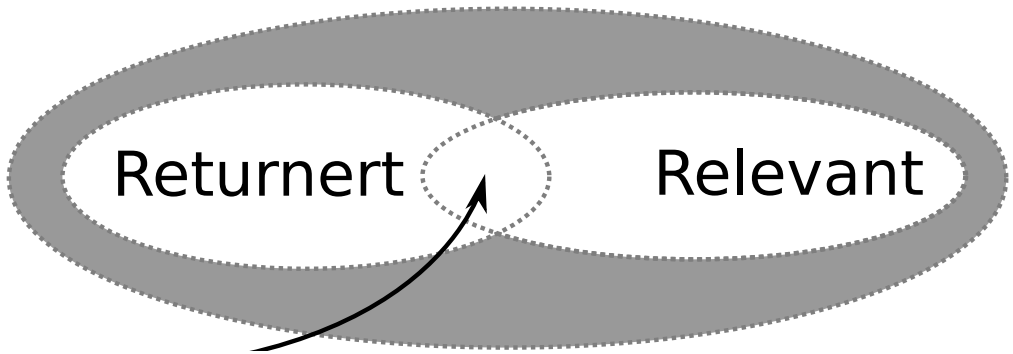
For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



En del av de returnerte resultatene vil (forhåpentligvis) være relevante.

Urangerte resultatmengder

For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.

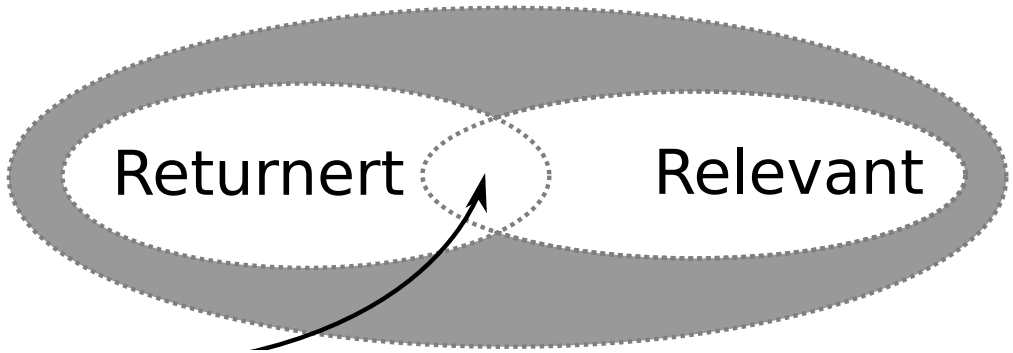


En del av de returnerte resultatene vil (forhåpentligvis) være relevante.

Mengden av *returnerte relevante dokumenter* er grunnlag for to ofte brukte kvalitetsmål.

Urangerte resultatmengder

For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.

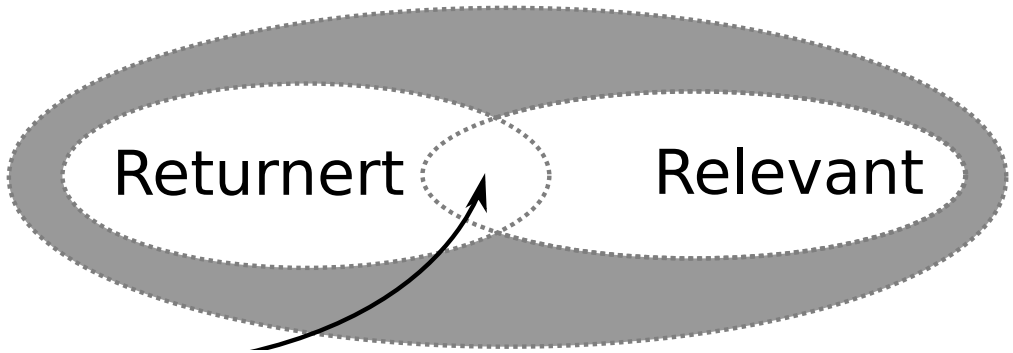


Nøyaktighet (*precision*)

$$\frac{\#(\text{relevante elementer returnert})}{\#(\text{elementer returnert})}$$

Urangerte resultatmengder

For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



Nøyaktighet (*precision*)

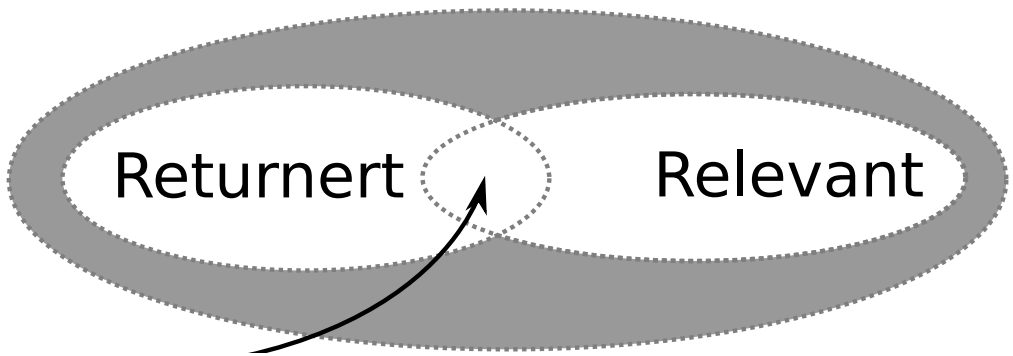
$$\frac{\#(\text{relevante elementer returnert})}{\#(\text{elementer returnert})}$$

Gjennkalling (*recall*)

$$\frac{\#(\text{relevante elementer returnert})}{\#(\text{relevante dokumenter})}$$

Urangerte resultatmengder

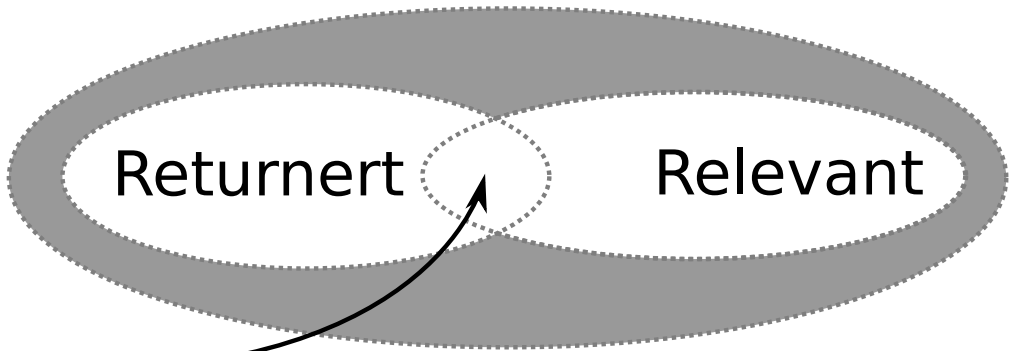
For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



	relevant	*irrelevant*
returnert	sann positiv (*tp*)	
ikke returnert		sann negativ (*tn*)

Urangerte resultatmengder

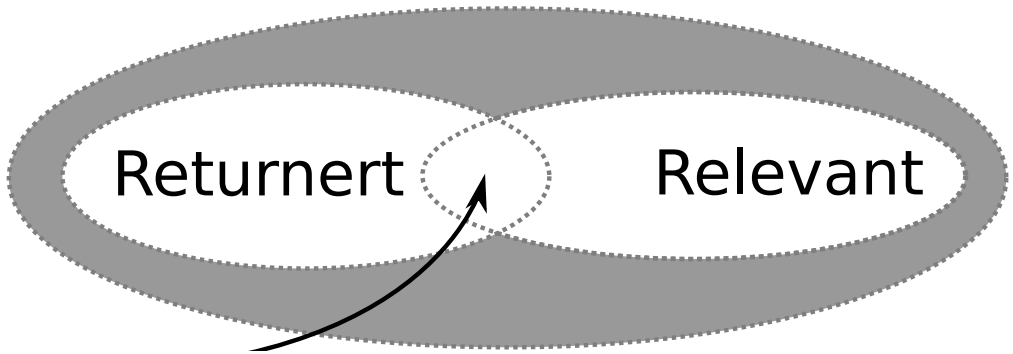
For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



	relevant	*irrelevant*
returnert	sann positiv (*tp*)	falsk positiv (*fp*)
ikke returnert		sann negativ (*tn*)

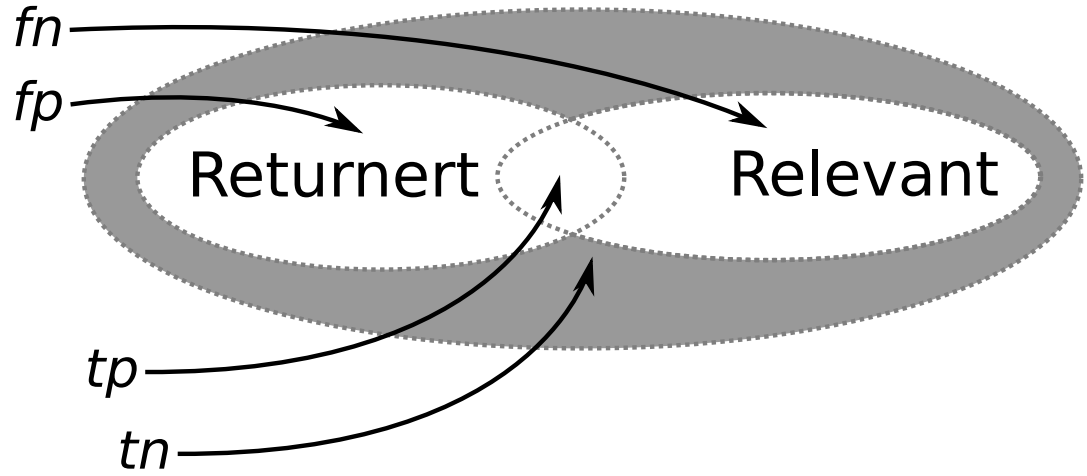
Urangerte resultatmengder

For et gitt informasjonsbehov regner vi alle dokumentene nå enten som relevant eller irrelevant.



	relevant	*irrelevant*
returnert	sann positiv (*tp*)	falsk positiv (*fp*)
ikke returnert	falsk negativ (*fn*)	sann negativ (*tn*)

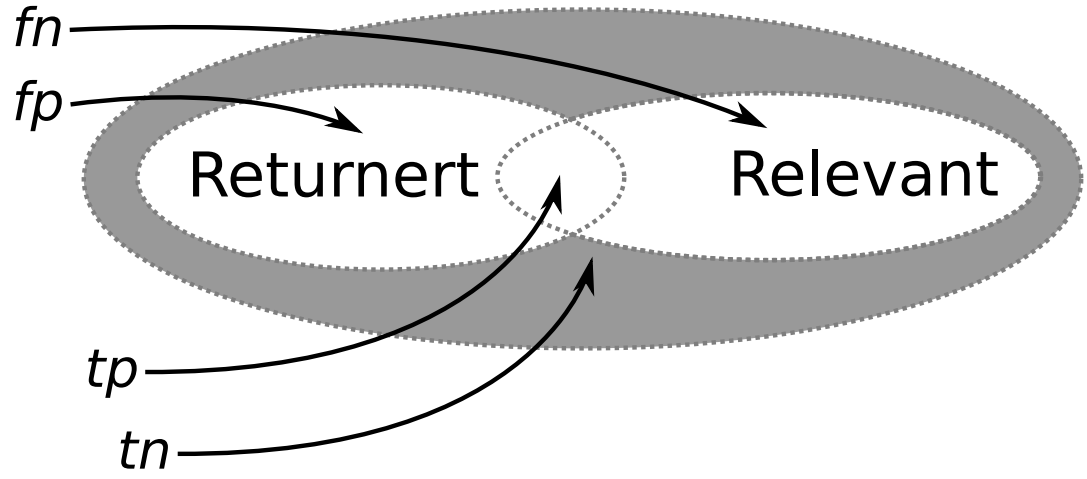
Urangerte resultatmengder



Nøyaktighet

$$P = \frac{tp}{tp + fp}$$

Urangerte resultatmengder



Nøyaktighet

$$P = \frac{tp}{tp + fp}$$

Gjennkalling

$$R = \frac{tp}{tp + fn}$$

Urangerte resultatmengder

Ideelt vil vi selvsagt at $P = R = 1$, men det klarer vi generelt ikke.

Hvis vi returnerer flere dokumenter vil *recall* øke (i det minste ikke minske).

Hvis vi returnerer alle dokumentene får vi 100% *recall*.

Urangerte resultatmengder

Ideelt vil vi selvsagt at $P = R = 1$, men det klarer vi generelt ikke.

Hvis vi returnerer flere dokumenter vil *recall* øke (i det minste ikke minske).

Hvis vi returnerer alle dokumentene får vi 100% *recall*.

Generelt er det motsatte tilfelle for *precision*.

Hvis systemet er ganske bra har vi allerede samlet en god del av de relevante dokumentene, og ofte vil nøyaktigheten *synke* dersom vi returnerer flere dokumenter.