Università degli Studi di Cagliari
Corso di Laurea DSBAI

Web Analytics e Analisi Testuale

http://agilegroup.eu

A.A. 2019/2020

Ing. Marco Ortu
Via Porcell 4, primo piano
mail: marco.ortu@unica.it

Information Retrieval e Fondamenti di Web e Text Search

Information Retrieval

Il settore dell'Information Retrieval è stato studiato fin dagli anni `70.

Negli anni '90, l'esplosione del Web ha cambiato radicalmente l'interesse per IR.

Il Web infatti non è altro che un'enorme collezione di documenti (Web 1.0 almeno), sui quali gli utenti vogliono fare ricerche.

- ➔ Una sorgente di informazioni virtualmente illimitata
- ➔ Accesso universale ed a basso costo
- ➔ Non esiste un controllo editoriale centralizzato
- ➔ Molti nuovi problemi si pongono: IR è vista come una area chiave per identificare soluzioni appropriate

Il problema è che non è semplice caratterizzare esattamente i bisogni informativi dell'utente.

Ciò deriva dall'ambiguità e complessità dei linguaggi naturali

Information Retrieval

Caratterizzare l'interesse di un utente non è facile:

"*trova tutte le pagine che contengono informazioni su squadre di tennis di Università Americane, e che partecipino alle gare NCAA. Mi interessano solo documenti che riportino anche la qualifica di queste squadre negli ultimi 3 anni e la email o telefono del loro allenatore*"

Irrilevante per la query e necessitano una qualche forma di ragionamento per dare alla macchina la conoscenza di fatti.

Esempio

MIT è una Università_Americana
march_2017 è incluso nell'intervallo [january_2015, january_2018]

Information Retrieval

IR tratta problemi di rappresentazione, memorizzazione, organizzazione e accesso ad informazioni

Obiettivo

facilitare l'accesso alle informazioni cui un utente è interessato

Esempio

Un testo libero non può direttamente essere usato per interrogare gli attuali motori di ricerca su Web. Una descrizione va trasformata in una query, una struttura di dati (usualmente un **vettore**) adatta all'elaborazione da parte di un motore di ricerca.

Information Retrieval

Un sistema di **Data Retrieval** (ad esempio un DBMS) gestisce dati che hanno una struttura ben definita.

Un sistema di Information Retrieval gestisce testi scritti in linguaggio naturale, spesso non ben strutturati e semanticamente ambigui.

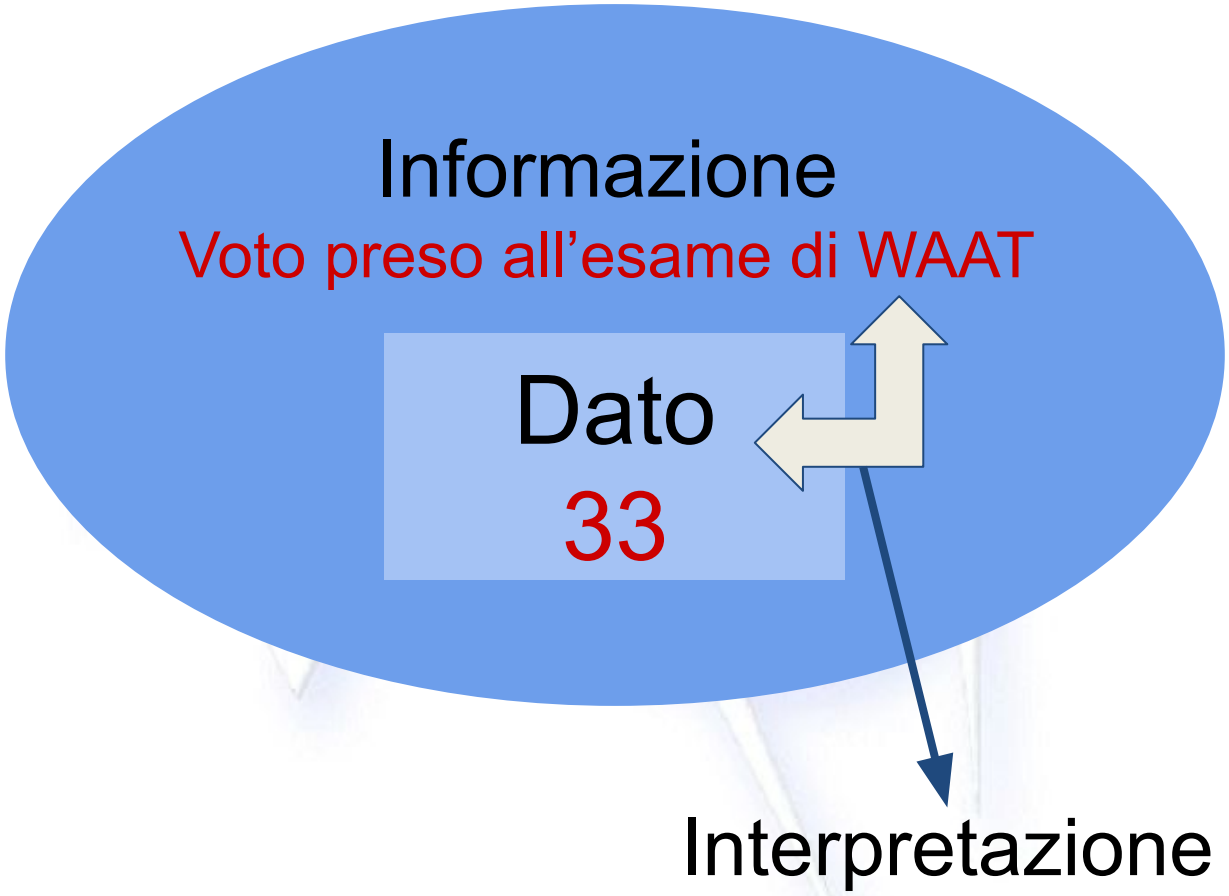
Di conseguenza

Un linguaggio per **Data Retrieval** permette di trovare tutti gli oggetti che soddisfano esattamente le condizioni definite.

Tali linguaggi (algebra relazionale, SQL) garantiscono una risposta corretta e completa e di manipolare la risposta.

Un sistema di IR, invece, potrebbe restituire anche oggetti non esatti (risultati non pertinenti); l'importante è che siano piccoli errori accettabili per l'utente.

Information Retrieval



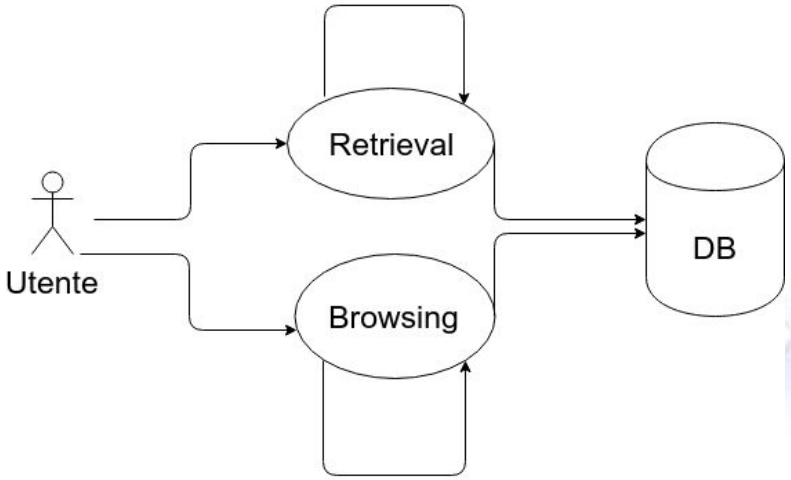
Information Retrieval

Data Retrieval
trovare oggetti che soddisfino condizioni chiaramente specificate mediante una espressione regolare **(FIND person WHERE age>40 AND...OR..)**.
Un errore è segnale di totale fallimento del sistema.

Information Retrieval
richiede una interpretazione della richiesta dell'utente. I documenti recuperati non possono essere classificati tout court come "buoni" o "cattivi", ma vanno associati ad una ***misura di rilevanza*** rispetto alla richiesta dell'utente (*all'interpretazione della richiesta!*)

La nozione di rilevanza è centrale in IR

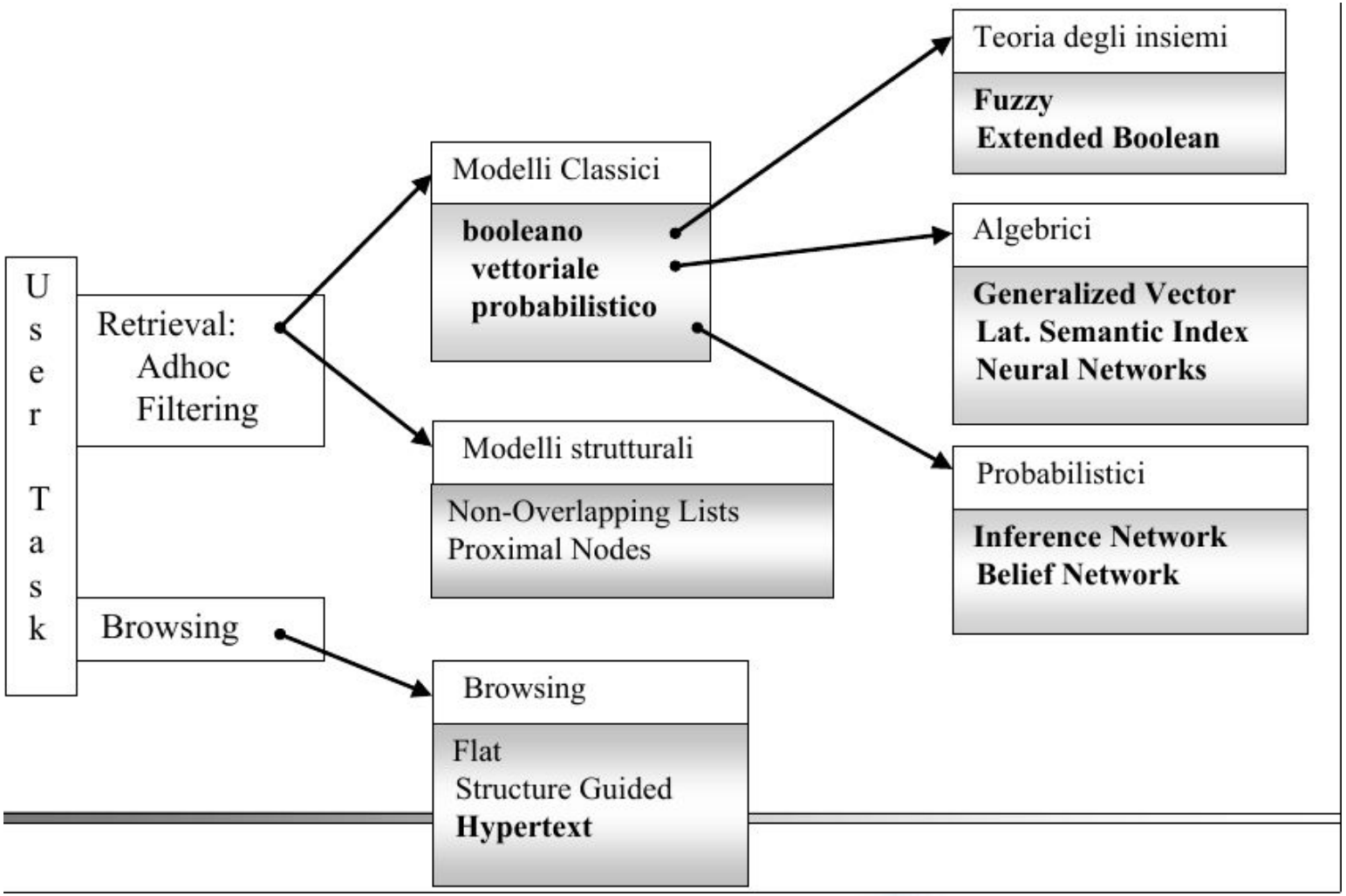
Interazione Utente



Una sessione di Retrieval comporta la specifica degli interessi dell'utente e la sua trasformazione in una query (usualmente, un insieme di parole chiave o keywords)

Se l'interesse dell'utente è mal specificato, o è molto vasto, l'utente può utilizzare una interfaccia interattiva, visualizzare alcuni documenti proposti, seguire hyperlinks a partire da documenti che più lo interessano, o dettagliare meglio la sua query. Si parla allora di una sessione di Browsing.

Information Retrieval



Information Retrieval

Retrieval e Browsing vengono classificate come azioni di pulling:
l'utente "tira fuori" le informazioni richieste

Alcuni sistemi sono in grado di prendere iniziative, cioè di spingere (push) documenti verso l'utente.

Ad esempio, filtrando notizie periodicamente sulla base di profili di utente (web assistant, information filtering and extraction).

Information Retrieval

Come le richieste dell'utente vengono trasformate in query, così un sistema di IR genera una rappresentazione più generale e astratta dei documenti, detta **logical view**, o **rappresentazione**.

Quando la rappresentazione di un documento comprende l'intero insieme delle parole che lo compongono, si parla di **full text logical view**.

Tuttavia, anche nei moderni computers sono necessarie alcune operazioni di compressione dell'informazione.

Queste operazioni possono essere finalizzate non solo alla compressione (es. eliminazione di articoli, congiunzioni., lemmatizzazione...) ma anche alla generalizzazione (non rappresentare termini (ambigui) ma concetti)

Websearch

L'IR è nata per gestire collezioni statiche e ben conosciute: testi di legge, enciclopedie ecc., ma quando la collezione di riferimento diventa il Web, le cose cambiano completamente:

- ➔ La collezione è dinamica, molto variabile nel tempo;
- ➔ Le dimensioni sono enormi;
- ➔ I documenti non sono sempre disponibili;
- ➔ Le query degli utenti sono ancora più imprecise e vaghe.
- ➔ Le tecniche utilizzate dai motori di ricerca possono quindi differire, ad esempio **Pagerank** (una cui variante è usata da Google) utilizza criteri diversi dalla IR standard

Structured IR

L’IR nasce per dati non strutturati. Tuttavia, negli ultimi anni sta emergendo la necessità di applicare tecniche di IR anche a dati semistrutturati, in particolare a documenti XML; si parla allora allora di Structured Information Retrieval (SIR).

Molte cose cambiano. Ad es. in IR la risposta ad una query è un elenco di documenti; in SIR, la risposta è un elenco di documenti XML? O frammenti di documenti XML? O singoli elementi?

Ultimamente sono state avanzate proposte per estendere XQuery con capacità di ricerca full-text.

Tecniche di IR: inverted index

I sistemi di IR non operano sui documenti originali, ma su una vista logica degli stessi. Tradizionalmente i documenti di una collezione vengono rappresentati tramite un insieme di keyword.

La capacità di memorizzazione dei moderni elaboratori permette talvolta di rappresentare un documento tramite l'intero insieme delle parole in esso contenute; si parla allora di **vista logica full text**.

Per collezioni molto grandi **tale tecnica può essere inutilizzabile**; si utilizzano allora tecniche di modifica del testo per ridurre la dimensione della vista logica, che diventa un insieme di **index term**.

Il modulo di gestione della collezione si occupa di creare gli opportuni indici, contenenti tali termini.

Tecniche di IR: inverted index

Le tecniche di indicizzazione studiate per le basi di dati relazionali (ad es. B-Tree) non sono adatte per i sistemi di Information Retrieval.
L'indice più utilizzato è l'indice invertito **(inverted index)**:

- ➔ Viene memorizzato l'elenco dei termini contenuti nei documenti della collezione;
- ➔ Per ogni termine, viene mantenuta una lista dei documenti nei quali tale termine compare.

Tale tecnica è valida per query semplici (insiemi di termini); modifiche sono necessarie se si vogliono gestire altre tipologie di query (frasi, prossimità ecc.).

Tecniche di IR: inverted index

Definizione: un inverted file è un meccanismo orientato alla manipolazione di parole, per indicizzare una collezione di documenti in modo da velocizzare il processo di ricerca.

Struttura di un inverted file:

- ➔ Vocabolario: l'insieme di parole diverse nel testo (NOTA: la taglia di V dipende dalle operazioni effettuate sui testi)

- ➔ Occorrenze: liste che contengono tutte le informazioni necessarie per ogni parola del vocabolario (posizione nel testo, frequenza, documenti nei quali appaiono, ecc.)

Tecniche di IR: inverted index

Il numero di termini indicizzati viene ridotto utilizzando una serie di tecniche, tra cui:

- **Eliminazione delle stopword**: articoli, congiunzioni ecc.;
  - "Il cane di Luca" -> "cane Luca"
- **De-hyphenation**: divisione di parole con trattino;
  - "Sotto-colonnello" -> Sotto colonnello
- **Stemming**: riduzione alla radice grammaticale;
  - "Mangiano, mangiamo, mangiassi" -> "Mangiare"
- **Thesauri**: gestione dei sinonimi, omonimi, ipernonimi
  - "Casa" -> "Casa, magione, abitazione, ..."

L'utilizzo di tali tecniche non sempre migliora la qualità delle risposte ad una query.

Tecniche di IR: inverted index

Thesauri, Iperonimi e iponimi

Gli **iperonimi** sono parole che indicano l'insieme più ampio di cui fanno parte altre parole.

Gli **iponimi**: rosa, viola, papavero sono iponimi di fiore, che è il loro iperonimo;

➔ felino è iperonimo di gatto, che a sua volta è iperonimo di soriano, siamese ecc.

Possono essere considerati iperonimi anche i sinonimi che hanno un significato un po' più ampio di altre parole, come dire rispetto a dichiarare, affermare, sostenere, asserire, esporre, rispondere, replicare ecc.

Tecniche di IR: inverted index

Thesauri, Iperonimi e iponimi

Gli **iperonimi** servono per evitare ripetizioni, per generalizzare (I partigiani lottavano per ideali di libertà, democrazia, progresso sociale) o per definire il significato degli iponimi (Il soriano è un gatto dal pelo corto).

Gli **iponimi** servono per spiegare il significato degli iperonimi, attraverso esempi più concreti (Le labiate comprendono basilico, origano, menta, salvia, timo, rosmarino, lavanda ecc.), ma soprattutto per indicare qualcosa in modo più specifico e preciso
(Chiese se sarebbe venuto e lui disse (errato per rispose) che non poteva)**.**

Tecniche di IR: inverted index

Thesauri, meronimia e olonimia

La **meronimia** (dal greco μέρος = parte e ὄνομα = nome) è una relazione semantica utilizzata in linguistica. Un meronimo indica un costituente o un membro di qualcosa.

- ➔ X è meronimo di Y se X è parte di Y,
- ➔ X è meronimo di Y se X è un membro di Y.

Ad esempio, 'dito' è meronimo di 'mano' in quanto un dito è parte della mano. In modo simile 'ruota' è meronimo di 'automobile'.

La meronimia è la relazione opposta di **olonimia**. Un concetto correlato è la mereologia che tratta relazioni parte/intero in logica.

Nei linguaggi di rappresentazione della conoscenza, la meronimia è spesso espressa come relazione "**part-of**" (parte di).

Tecniche di IR: inverted index

Thesauri, meronimia e olonimia

Olonimia (dal greco holon = intero e onoma = nome) è una relazione semantica.

L'olonimia indica la relazione tra un termine che indica l'intero e un termine che ne rappresenta una parte, o un membro. Ovvero:

- ➔ 'X' è una olonimia di 'Y' se Y è parte di X, o
- ➔ 'X' è una olonimia di 'Y' se Y è membro di X.

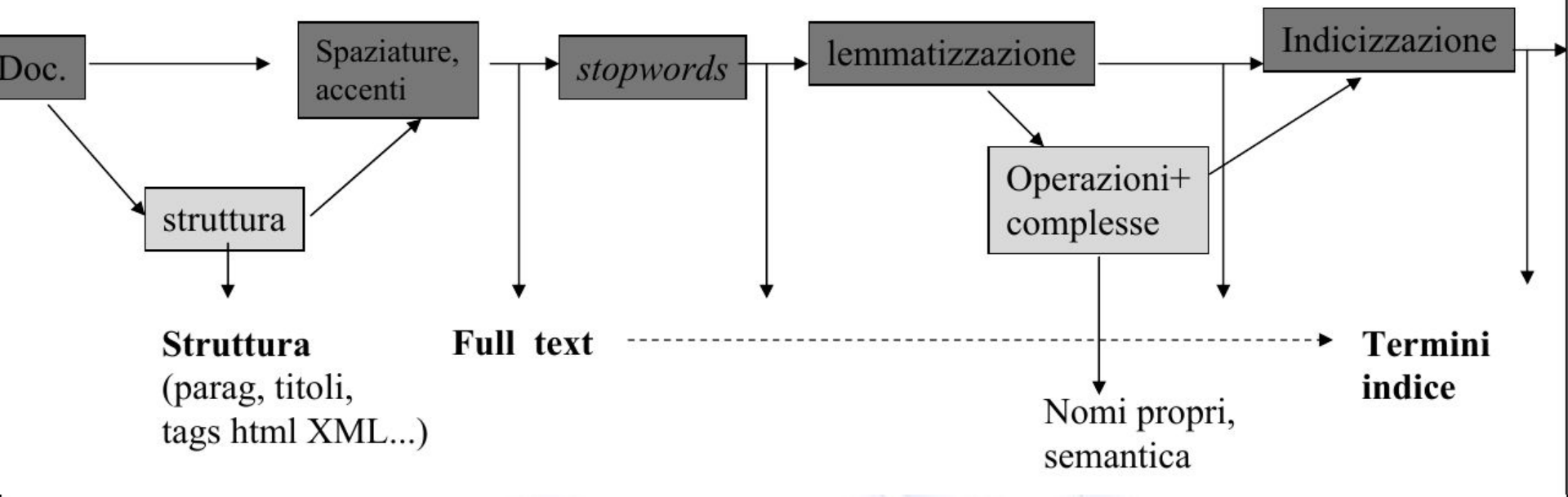
Ad esempio, l'albero è una olonimia di 'corteccia', o 'tronco'.

La Princeton University mantiene un un ***Lexical Database*** chiamato:

WordNet

WordNet contiene tutte le relazioni semantiche tra i termini di un linguaggio.

Tecniche di IR: inverted index



Tecniche di IR: inverted index

Avendo quindi non i documenti ma i loro inverted index, il processo di ricerca di informazioni viene riformulato:

1. L'utente specifica un bisogno informativo.

2. La query viene eventualmente trasformata…

3. ...per poi essere eseguita, utilizzando indici precedentemente costruiti, al fine di trovare documenti rilevanti;

4. I documenti trovati vengono ordinati in base alla (**presunta**) **rilevanza** e ritornati in tale ordine all‟utente;

5. L'utente esamina i documenti ritornati ed eventualmente raffina la query, dando il via ad un nuovo ciclo.

Tecniche di IR: inverted index

Formalmente un **modello di Information Retrieval** è una quadrupla **(D, Q, F, R)**, dove:

- ➔ **D** è un insieme di viste logiche dei documenti della collezione;
- ➔ **Q** è un insieme di viste logiche (dette query) dei bisogni informativi dell'utente;
- ➔ **F** è un sistema per modellare documenti, query e le relazioni tra loro;
- ➔ **R(qi, dj)** è una funzione di ranking che associa un numero reale non negativo ad una query qj e un documento dj, definendo un ordinamento tra i documenti con riferimento alla query qj.

I due modelli classici di Information Retrieval sono il Modello **booleano** e quello **vettoriale**.

Tecniche di IR: inverted index

Ogni documento viene rappresentato mediante un insieme di parole-chiave o termini indice

Un termine-indice è una parola ritenuta utile per rappresentare il contenuto del documento

In genere, nei sistemi di IR “classici”, gli indici sono nomi, perché maggiormente indicativi del contenuto

Tuttavia, nei motori di ricerca vengono considerati tutti i termini (full text representation)

Tecniche di IR: inverted index

Un ranking è un ordinamento dei documenti recuperati che dovrebbe riflettere gli interessi dell'utente

E' basato su :

- ➔ Identificazione di gruppi di termini comuni
- ➔ Condivisione di termini pesati
- ➔ Probabilità di rilevanza

La classificazione dei modelli di IR è basata su diversi criteri di ranking

Tecniche di IR: inverted index

Trascurando, per ora, il problema del passaggio dal “full text” alla visione logica, supponiamo di avere a disposizione tale visione logica, e questa sia data da un insieme di parole-chiave o keywords.

Un banale matching di parole chiave fra documenti e query spesso fornisce risultati modesti, gli utenti sono insoddisfatti.

Inoltre, spesso gli utenti non sono in grado di esprimere i loro interessi mediante un elenco appropriato di keywords.

Il problema è reso più grave se siamo nell’ambito Web IR

Un ranking appropriato dei documenti ha un effetto notevole sulle prestazioni (vedi i motori di ricerca)

Indicizzazione full-text: inverted indexes

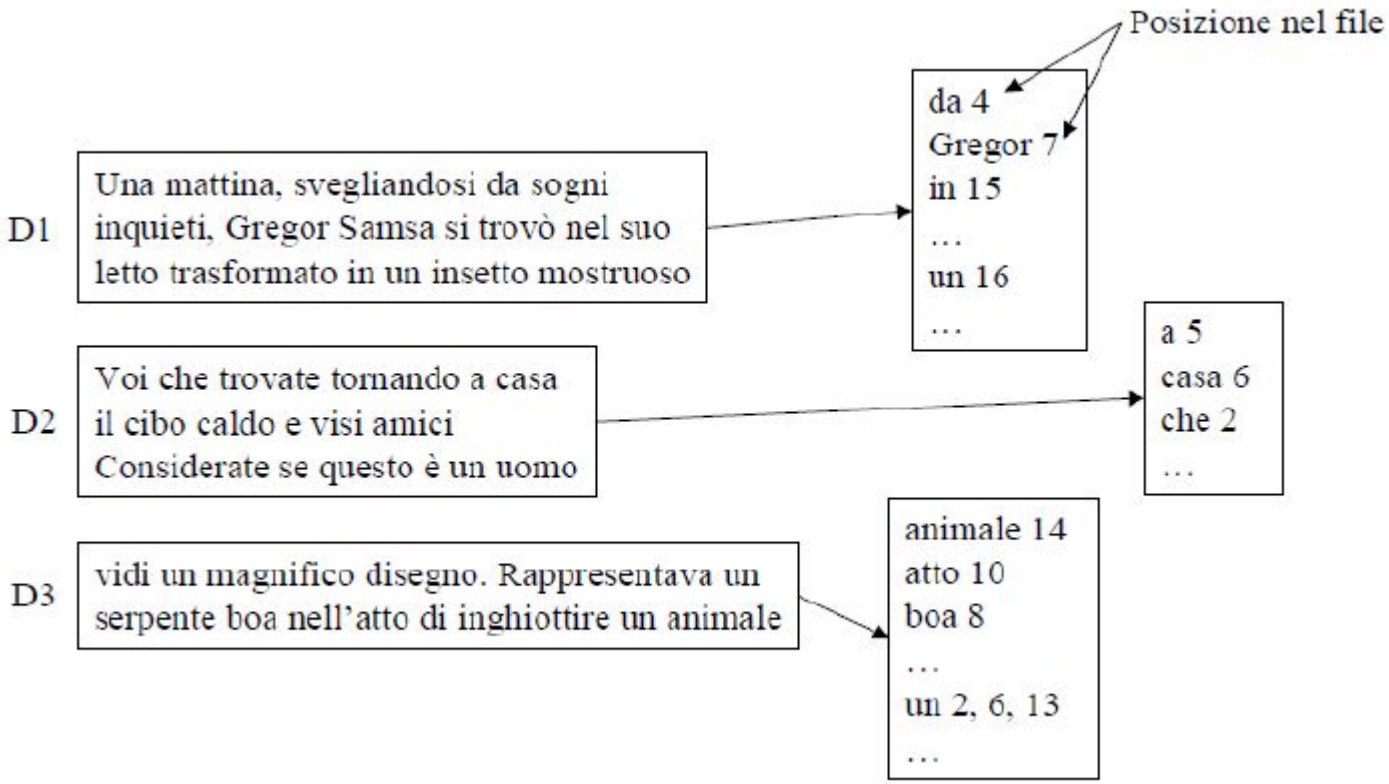
Nei sistemi relazionali gli indici permettono di recuperare efficientemente da una relazione i record contenenti uno o più valori di interesse. Analogamente, possiamo utilizzare indici per recuperare efficientemente documenti testuali di interesse. In questo caso, le interrogazioni che vogliamo supportare sono del tipo:

- ➔ Ritornare i documenti che contengono l'insieme di keyword k1, ..., kN.
- ➔ Ritornare i documenti che contengono la sequenza di keyword k1, ..., kN.

Indici che supportano queste operazioni si **chiamano invertiti**, in quanto invece di rappresentare tutte le parole presenti in documento (indici) rappresentano tutti i documenti in cui appare una specifica parola.

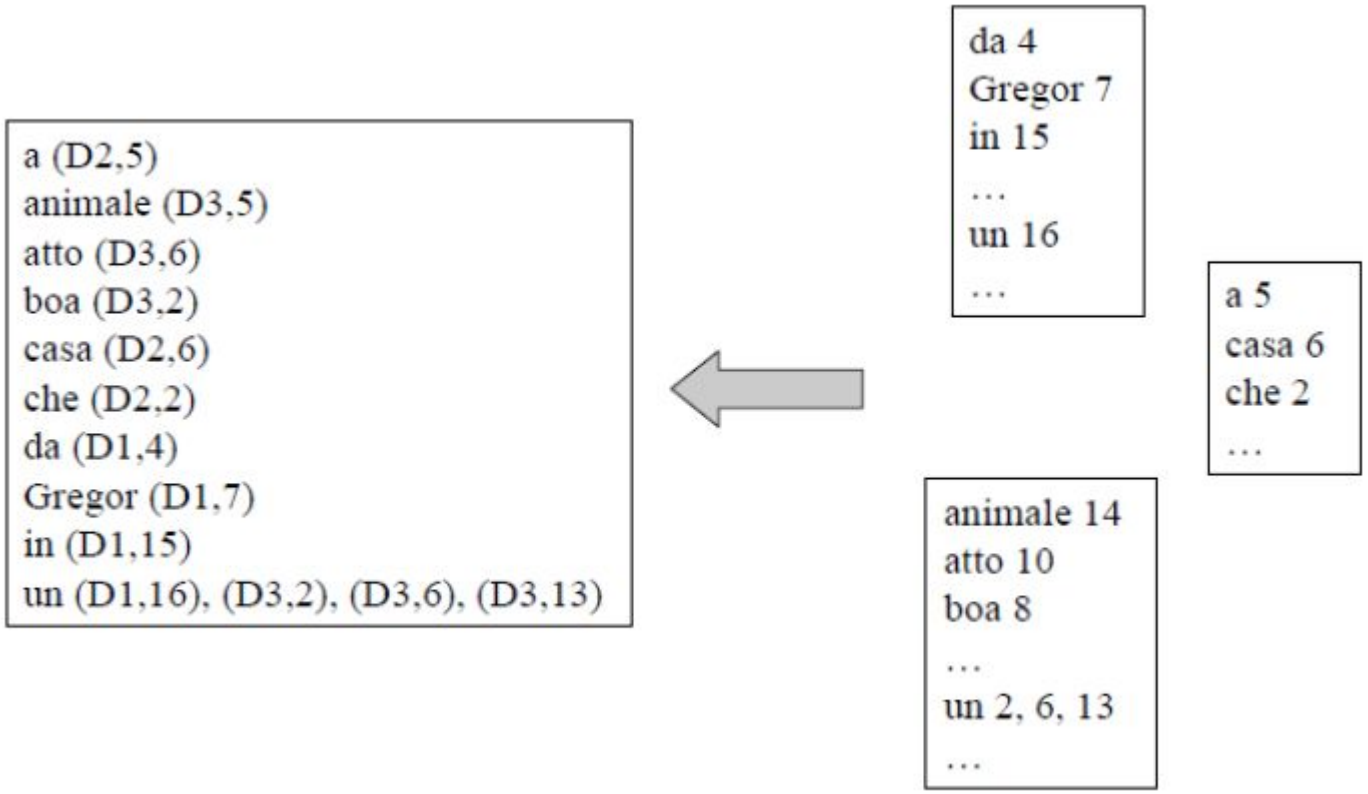
Indicizzazione full-text: inverted indexes

Generazione degli indici: rilevamento della posizione dei terms



Indicizzazione full-text: inverted indexes

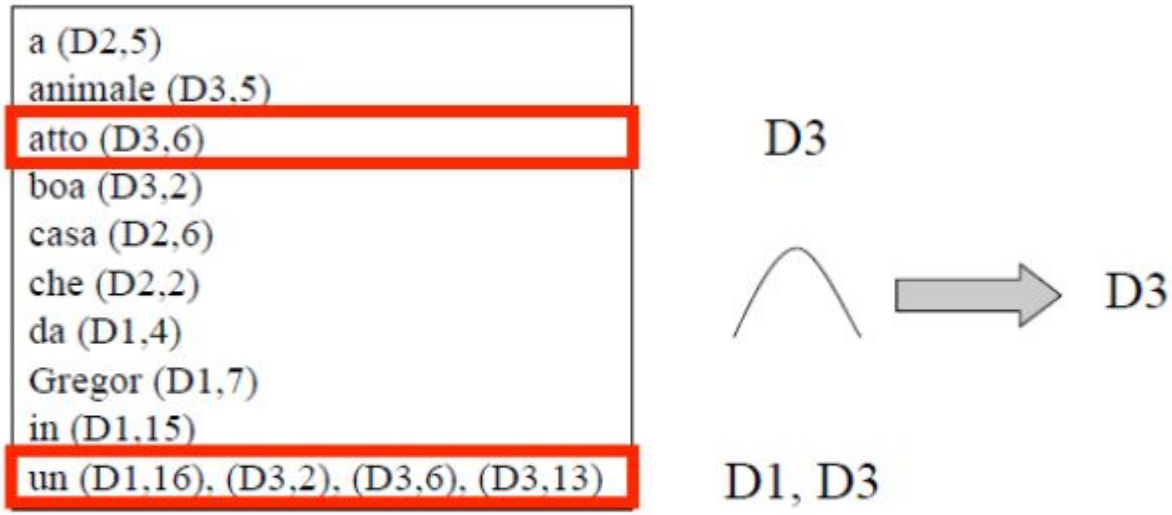
Generazione degli indici: creazione dell'inverted index



Indicizzazione full-text: inverted index

E’ successivamente possibile effettuare delle interrogazioni:

- Ritornare i documenti che contengono “un” e “atto”.

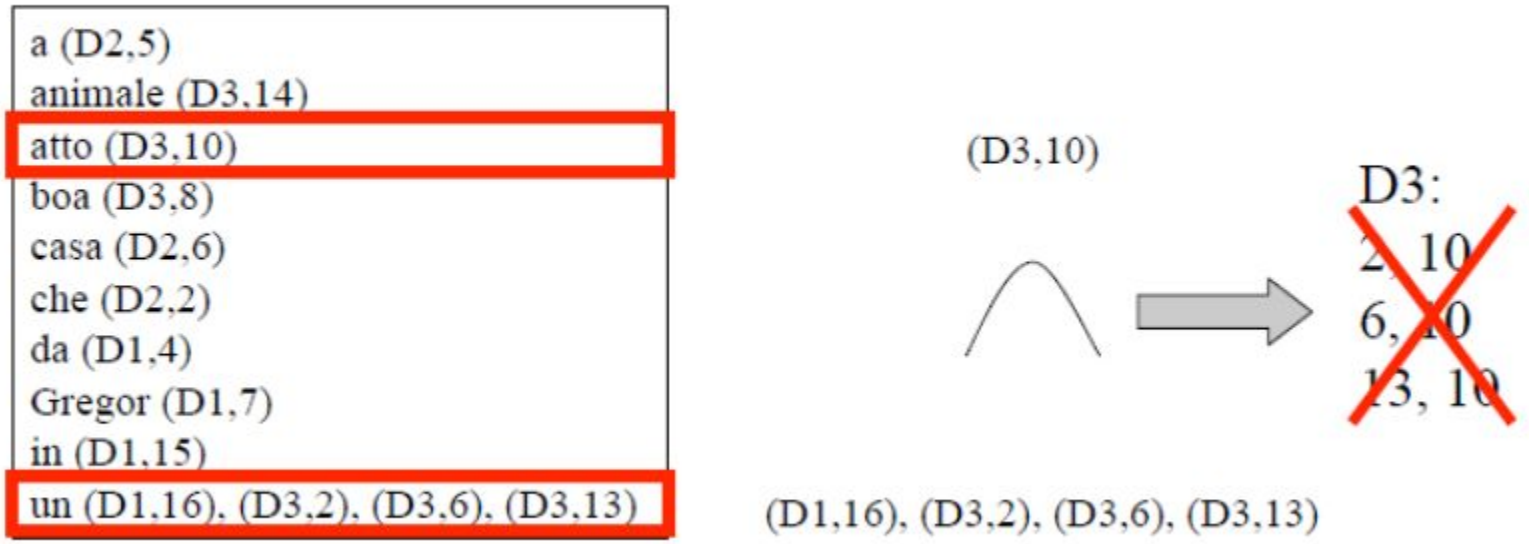


D3 | vidi **un** magnifico disegno. Rappresentava **un** serpente boa nell’**atto** di inghiottire **un** animale

Indicizzazione full-text: inverted index

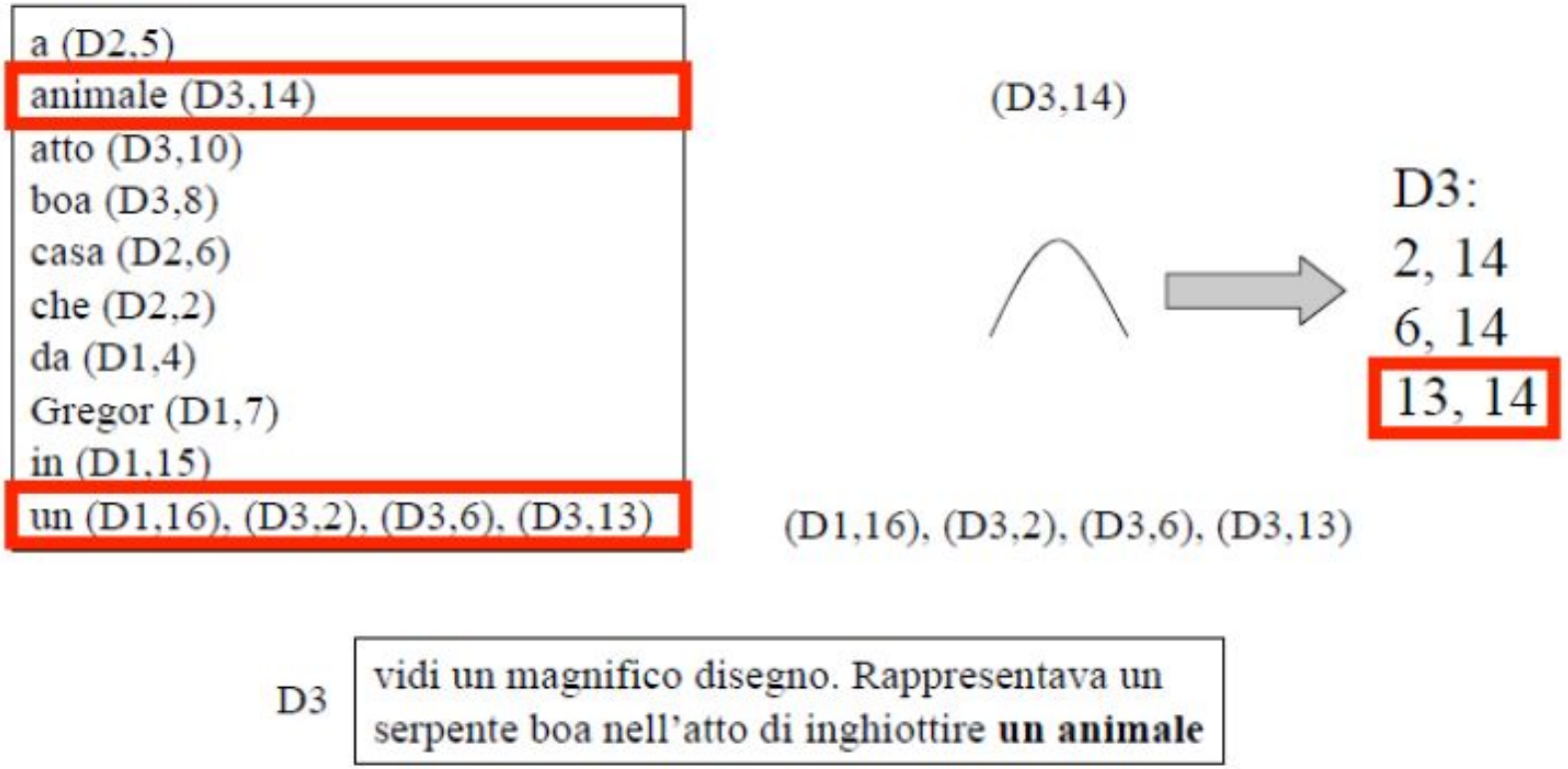
E’ successivamente possibile effettuare delle interrogazioni:

- Ritornare i documenti che contengono “un atto”.



Tecniche di IR: inverted index

- Ritornare i documenti che contengono la sequenza di parole “un animale”.



Indicizzazione full-text: inverted index

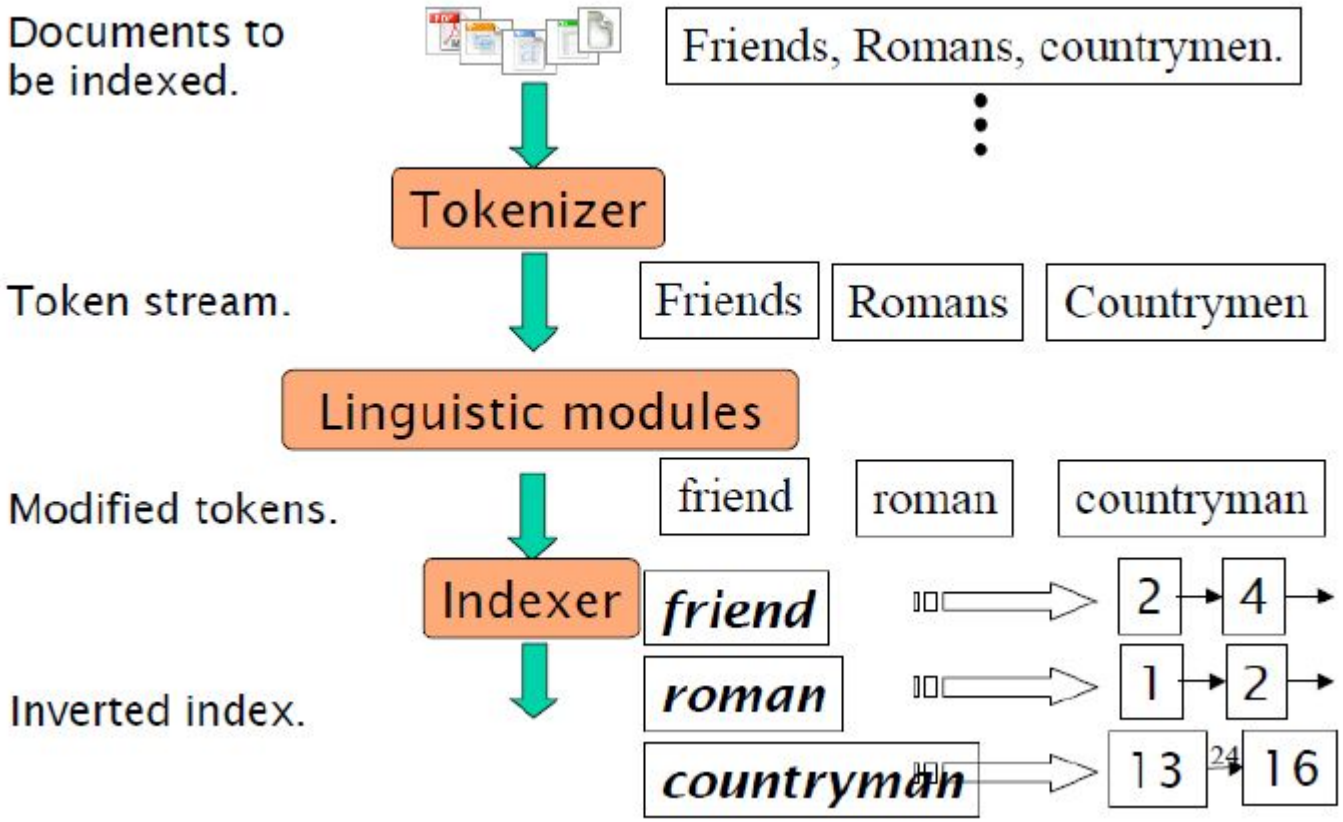
La **costruzione dell inverted index è più complessa**, come visto infatti, gli indici invertiti lavorano su "termini", **ma cos'è un termine**?

- ➔ Principi → Prìncipi? Princìpi?
- ➔ Cordon Bleu → un termine o due?
- ➔ Semi-structured → Semistructured? Semi-structured? Semi e Structured?
- ➔ La, e, and, or, 192.168.0.1, 25/12/2004... → vanno indicizzati?
- ➔ Auto e Automobile → lo stesso termine o due termini diversi?
- ➔ Sono, Siamo, E' → lo stesso termine o tre termini diversi?

Prima della fase di costruzione dell'indice, si devono quindi trasformare i documenti della collezione in un elenco di termini.

Indicizzazione full-text: inverted index

Il processo di indicizzazione consta di diverse fasi



Indicizzazione full-text: inverted index

La prima fase è quella di tokenizzazione, in cui il tokenizer trasforma uno stream di testo ("Friends, Romans, Countrymen") in un elenco di token ("Friends", "Romans", "Countrymen") **candidati** a diventare entry dell'indice. **Tipiche trasformazioni** effettuate durante la tokenizzazione:

- ➔ Eliminazione delle parole contenenti cifre;
- ➔ Divisione in più parole dove è presente un trattino (de-hyphenation);
- ➔ Trasformazione delle maiuscole in minuscole;
- ➔ Eliminazione della punteggiatura.

Ma è necessario **gestire alcune eccezioni**:

- ➔ Trattini che sono parti integranti di una parola (es. B-49);
- ➔ Parole che se scritte in maiuscolo assumono un diverso significato (es. MIT vs la parola tedesca mit);
- ➔ Punteggiatura che è parte integrante di una parola (es. 510D.C.).

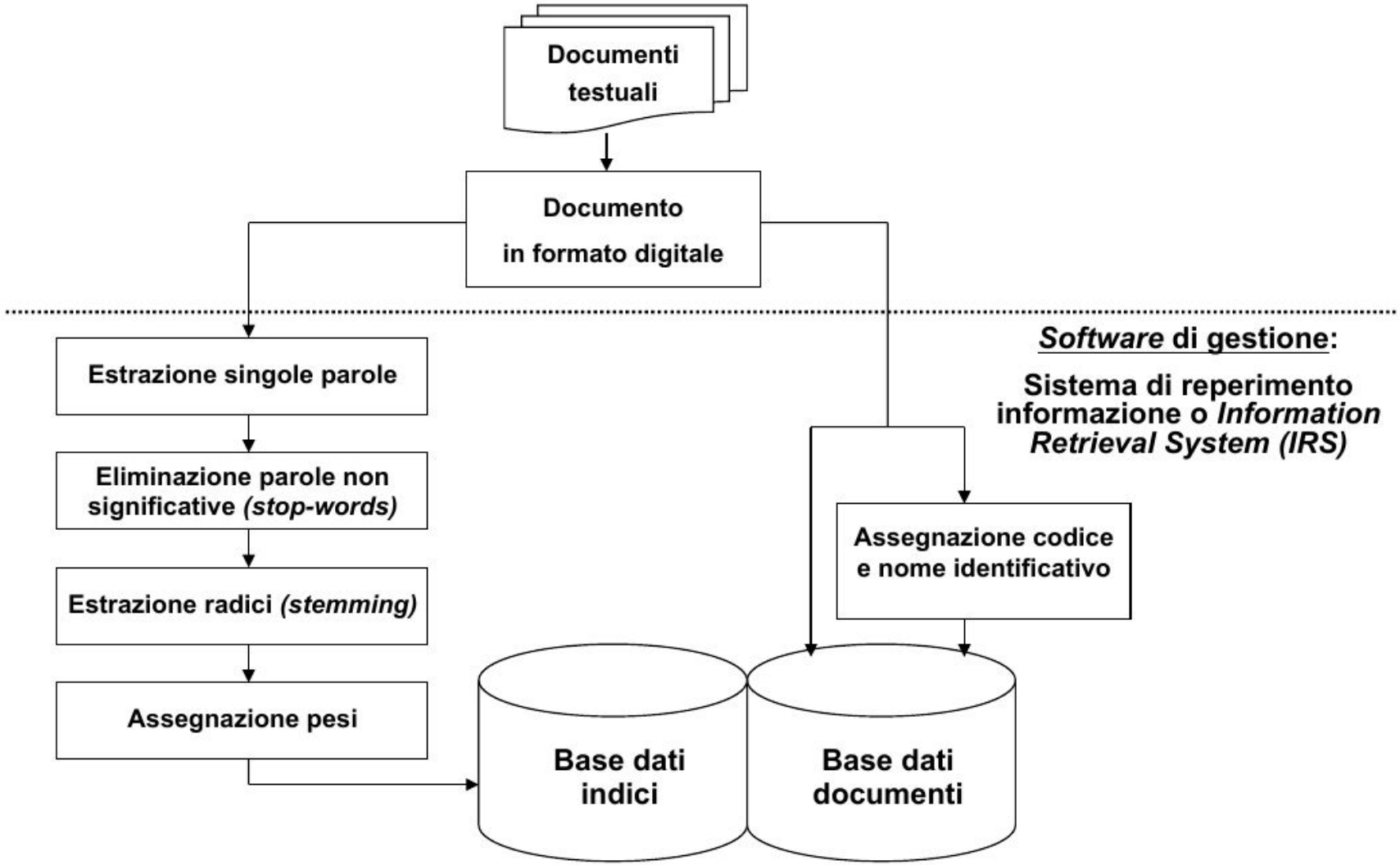
Indicizzazione full-text: inverted index

Successivamente, entrano in azione i moduli linguistici, il cui scopo è prendere i token e validarli, operazione che dipende dalla lingua utilizzata (anche se alcuni criteri sono generali).

Le operazioni effettuate dai moduli sono:

- ➔ Eliminazione delle stopword
- ➔ Stemming
- ➔ Thesauri
- ➔ Lemmatization

Information Retrieval



Indicizzazione full-text: inverted index

Eliminazione delle stopwords: alcune parole sono più importanti di altre per comprendere il contenuto di un documento. Le parole che non hanno un significato proprio e sono eliminate sono le stopwords: articoli, congiunzioni, particelle pronominali, verbi frequenti ecc. Eliminare le stopwords attenua il rumore che disturba la ricerca di informazioni e riduce la dimensione dell'indice.

Lo stemming riduce i termini alla loro "radice", rimuovendo prefissi e suffissi, ad esempio:

- ➔ automate, automatic, automation → automat;
- ➔ "for example compressed and compression are both accepted as equivalent to compress" → "for example compres and compres are both accept as equival to compres".

Esistono vari algoritmi di stemming; il più comune per l'inglese è l'algoritmo di **Porter**, che opera trasformazioni come:

- ➔ sses → ss (witnesses → witness);
- ➔ s → Ø (cars → car);
- ➔ tional → tion (national → nation).

Indicizzazione full-text: inverted index

Tipicamente vengono eliminate parole ad elevata frequenza in un linguaggio, dette **stopwords** (es. parole funzionali, in inglese: “a”, “the”, “in”, “to”; o pronomi: “I”, “he”, “she”, “it”, ecc.).

Le Stopwords sono ovviamente dipendenti dalla lingua. **Vector Space Retrieval** (VSR) utilizza un set standard di circa 500 parole inglesi.

Le stringhe di stopwords vengono memorizzate in una **hashtable** per essere **riconosciute in tempo costante**.

Stemming

Si intende per stemming il processo di riduzione di un termine al lemma o alla radice, in modo da riconoscere variazioni morfologiche della stessa parola.

➔ “comput-er”,“comput-ational”, “comput-ation” → “compute” L’analisi morfologica è specifica di ogni lingua, e può essere molto complessa (ad esempio in Italiano ben più che inInglese)

I sistemi di stemming più semplici si limitano ad identificare suffissi e prefissi e ad eliminarli.

Porter Stemmer

E’ una procedura semplice, che iterativamente riconosce ed elimina suffissi e prefissi noti senza utilizzare un dizionario (lemmario).

Può generare termini che non sono parole di una lingua:

➔ “computer”, “computational”, “computation” diventano tutti “comput”

Vengono unificate parole che in effetti sono diverse (matto e mattone).

Non riconosce deviazioni morfologiche (vado e andiamo).

Errori del Porter Stemmer

Errori di “raggruppamento”:

- ➔ organization, organ → organ
- ➔ police, policy → polic
- ➔ arm, army → arm

Errori di “omissione”:

- ➔ cylinder, cylindrical
- ➔ create, creation
- ➔ Europe, European

Operazioni più complesse sui testi

Analisi morfosintattica

➔ I have been → be

Analisi dei Nomi Propri, date, espressioni monetarie e numeriche, terminologia

➔ Medical Instrument inc

➔ April 15th, 2003. 15-4-03...

➔ 5 millions euros

➔ Consiglio di amministrazione, week end, ..

Analisi della struttura del testo

➔ Es: I termini nel titolo o nei paragrafi hanno un peso maggiore, i termini in grassetto o sottolineati..

Analisi semantica: associare a parole singole o a porzioni di testo dei concetti di una **ontologia**

➔ “L’albergo dispone di piscina..” Is_a hotel_facility

➔ “L’albergo si trova a Milis..” Is_in Campidano Sardegna ..

Indicizzazione full-text: inverted index

Con i **thesauri** si gestiscono i sinonimi tramite classi di equivalenza predefinite, ad esempio car = automobile. Due possibili tecniche:

- ➔ Espansione dell'indice: se un documento contiene car, lo inseriamo nei posting sia di car che di automobile;
- ➔ Espansione della query: se una query contiene car, cerchiamo anche i documenti contenenti automobile (preferibile).
- ➔ L'utilizzo delle classi di equivalenza può però portare a risultati scorretti; ad es. puma e jaguar sono sinonimi, ma rischio di trovare informazioni relative alle automobili piuttosto che all'animale...

La **lemmatization** riduce una parola alla sua radice grammaticale, ad esempio:

- ➔ am, are, is → be;
- ➔ car, cars, car's, cars' → car;
- ➔ the boy's cars are different colors → the boy car be different color.

Indicizzazione full-text: inverted index

Come visto, i moduli linguistici sono dipendenti dal linguaggio usato nel testo.

Possono crearsi problemi se il testo contiene parole scritte in diversi linguaggi. Alcuni linguaggi creano inoltre problemi aggiuntivi: pensiamo al giapponese, cinese, arabo etc.

Danno dei benefici in termini di precisione della ricerca e dimensione dell'indice.

Ma trasformare il testo può rendere più difficile la ricerca all'utente: pensiamo alla ricerca "to be or not to be”.

Per questo motivo non tutti sono concordi sull'opportunità di usarli (molti Web Search Engine non lo fanno).

Dimensioni di un Inverted File in termini di % (approssimate) della taglia dell'intera collezione

Percentuale di riduzione	Piccole collezioni (1Mb)		Collezioni medie (200Mb)		Collezioni grandi (2Gb)	
In % delle parole	45%	73%	36%	64%	35%	63%
In % dei documenti~10k	19%	26%	18%	32%	26%	47%
In % di blocchi da 256 K	18%	25%	1.7%	2.4%	0.5%	0.7%

Ricerca su inverted indexes

L’algoritmo di ricerca basato su indici inversi opera in tre fasi successive:

- ➔ Ricerca nel vocabolario: le parole presenti nella query vengono ricercate nel vocabolario
- ➔ Recupero delle occorrenze: viene estratta la lista delle occorrenze di tutte le parole incluse nella query
- ➔ Manipolazione delle occorrenze: la lista delle occorrenze viene elaborata, per generare una risposta alla query

Ricerca su inverted indexes(2)

La ricerca parte sempre dal vocabolario, che dovrebbe essere memorizzato separatamente

I vocabolari vengono memorizzati in strutture di tipo hashing o B-trees

Alternativamente, le parole possono essere memorizzate in ordine alfabetico (si risparmia in spazio, maggior tempo di ricerca)

Modelli di IR: modello Booleano

Non tutti i termini sono ugualmente importanti per la rappresentazione di un testo (o query).
Nella selezione di termini indice è importante assegnare un peso di rilevanza alle varie parole.

L'importanza di un termine indice è rappresentata da un valore che ad esso viene associato
Sia

- ➔ k_i un termine indice
- ➔ d_j un documento della collezione
- ➔ w_{ij} un peso associato a (k_i,d_j)

Il peso wij quantifica l'importanza dell'indice ki per descrivere il contenuto del documento

Modelli di IR: modello Booleano

ki è un termine indice

dj è un documento

t è il numero totale di documenti in D

K = (k1, k2, ..., kt) è l'insieme dei termini indice

wij >= 0 è un peso associato con (ki,dj)

wij = 0 indica che un termine non appartiene al documento

vec(dj) =dj =(w1j, w2j, ..., wtj) è un vettore pesato associato a dj

gi(dj) = wij è una funzione che restituisce il peso associato alla coppia (ki,dj)

Modelli di IR: modello Booleano

Un modello molto semplice basato sulla set theory

Le query vengono rappresentate mediante espressioni booleane

➔ $q = ka \wedge (kb \vee \neg kc)$

Es: **(automobili ∧ (vendita ∨ ¬fabbricazione)**

I termini sono presenti o assenti. Dunque $wij \in \{0,1\}$

Modelli di IR: modello Booleano

Il **modello booleano** è il modello più semplice; si basa sulla teoria degli insiemi e l'algebra booleana.

Storicamente, è stato il primo ed il più utilizzato per decenni.

I documenti vengono rappresentate come insiemi di termini.

Le query vengono specificate come espressioni booleane, cioè come un elenco di termini connessi dagli operatori booleani AND, OR e NOT.

La strategia di ricerca è basata su un criterio di decisione binario, senza alcuna nozione di grado di rilevanza: un **documento è considerato rilevante oppure non rilevante**.

Modelli di IR: modello Booleano

Nel modello booleano, un documento soddisfa le condizioni oppure no.

Questo modello può essere ragionevole solo per degli utenti esperti che conoscano perfettamente la collezione di documenti e le proprie necessità.

Una query può ritornare migliaia di risultati, ma la maggior parte degli utenti non vogliono scorrere migliaia di voci.

Inoltre una eventuale riformulazione della query provoca il ricalcolo dell'intero risultato, con evidenti problemi prestazionali.

Per questa serie di motivi, il modello booleano di fatto NON viene utilizzato, preferendo in sua vece quello vettoriale

Modelli di IR: modello Booleano

Il retrieval è basato su criteri di decisione binari, non esiste la nozione di corrispondenza parziale.

Non viene fornito un ordinamento parziale dei documenti (non c'è ranking!!)

Gli utenti trovano difficile trasformare le loro richieste informative in una espressione booleana

Gli utenti formulano spesso query booleane troppo semplicistiche (congiunzioni di termini)

Di conseguenza, le query booleane restituiscono o troppo pochi o troppi documenti

Esempi di interrogazioni booleane

D_1 = L'enorme quantità di informazioni presenti nelle pagine Web rende necessario l'uso di strumenti automatici per il recupero di informazioni
D_2 = I presenti hanno descritto le fasi del recupero dell'enorme relitto ma le informazioni non concordano su tipo e quantità di strumenti in uso
D_3 = E' stato presentato nel Web un documento che informa sulle enormi difficoltà che incontra chi usa uno strumento informativo automatico

recupero AND Web	→	D_1
recupero OR Web	→	D_1, D_2, D_3
recupero AND NOT relitto	→	D_1
(Web OR uso) AND strumenti	→	D_1, D_2
(Web OR uso) AND NOT strumenti	→	D_3
informazioni AND relitto AND studente	→	$\emptyset$
informazioni OR relitto OR Internet	→	D_1, D_2
bologna OR NOT padova	→	D_1, D_2, D_3

Modelli di IR: modello Vettoriale

Il **modello vettoriale** è giustificato dall'osservazione che assegnare un giudizio binario ai documenti (1=rilevante, 0=non rilevante) è troppo limitativo.

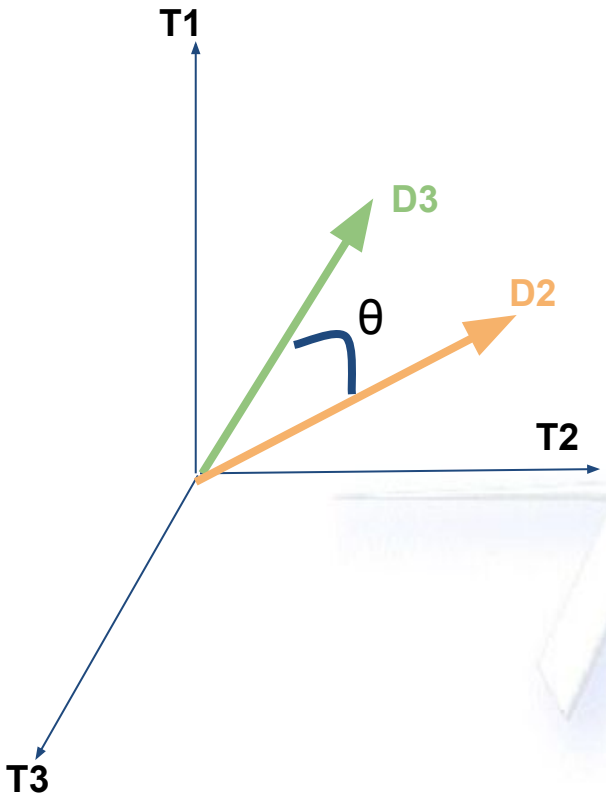
Nel modello vettoriale ad ogni termine nei documenti o nelle query viene assegnato un peso (un numero reale positivo).

I documenti e le query sono rappresentati come vettori in uno spazio n-dimensionale (n = # di termini indicizzati).

La ricerca viene svolta calcolando il grado di similarità tra il vettore che rappresenta la query e i vettori che rappresentano ogni singolo documento: i documenti con più alto grado di similarità con la query hanno più probabilità di essere rilevanti per l'utente.

Il grado di similarità viene quantificato utilizzando una qualche misura, ad esempio il coseno dell'angolo tra i due vettori (che esprime la **"vicinanza"** fra i vettori).

Modelli di IR: modello Vettoriale



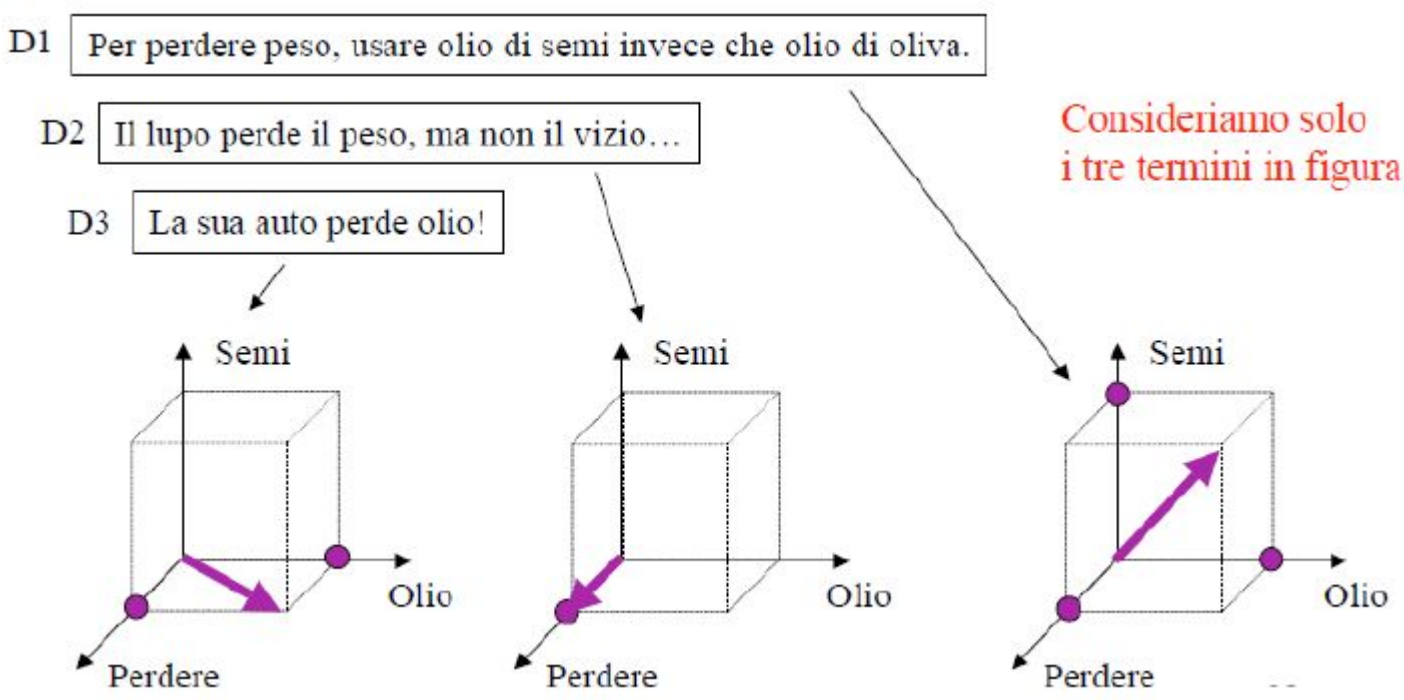
query e documenti sono rappresentati mediante vettori pesati in uno spazio vettoriale

Lo spazio ha una dimensionalità pari al numero di termini nel vocabolario.

La similarità tra la query e il documento impiega misure geometriche di similarità tra vettori.

Modelli di IR: modello Vettoriale

Ogni documento può essere visto come un vettore di valori in uno spazio vettoriale (n-dimensionale), i cui assi sono i termini, contenente i documenti. Le query possono essere viste come dei brevi documenti, e quindi anch'esse sono dei vettori appartenenti a questo spazio, esempio:



Modelli di IR: modello Vettoriale

Modello vettoriale:

D1 Per perdere peso, usare olio di semi invece che olio di oliva.

Semi	Perdere	Olio
1	1	1

D2 Il lupo perde il peso, ma non il vizio…

Semi	Perdere	Olio
0	1	0

D3 La sua auto perde olio!

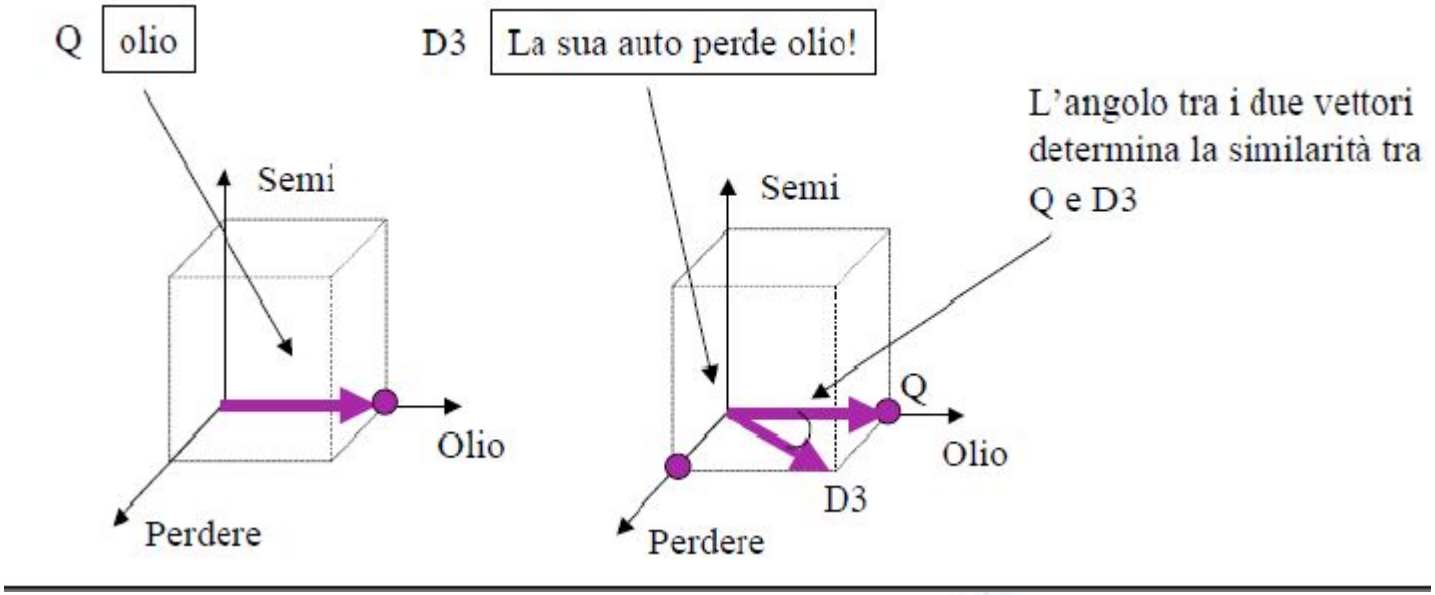
Semi	Perdere	Olio
0	1	1

Q olio

Semi	Perdere	Olio
0	0	1

Modelli di IR: modello Vettoriale

Per esprimere il concetto di similitudine fra documenti e la query, quindi per rispondere ad una query, si introduce una **metrica**, ossia una **distanza fra vettori**, che banalmente potrebbe essere quella degli spazi lineari, ossia il **coseno** dell'angolo fra i vettori, che di fatto viene calcolato usando le coordinate. I documenti saranno poi ordinati (**ranking**) in base alla minore distanza dalla query.

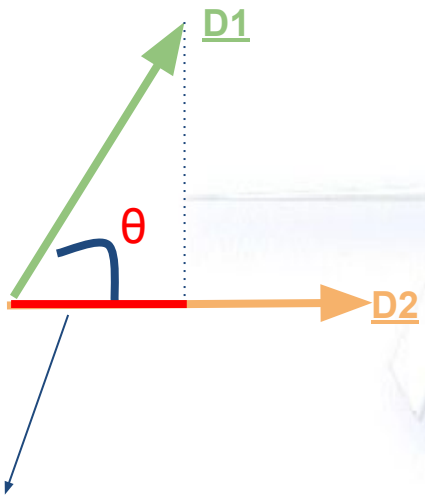


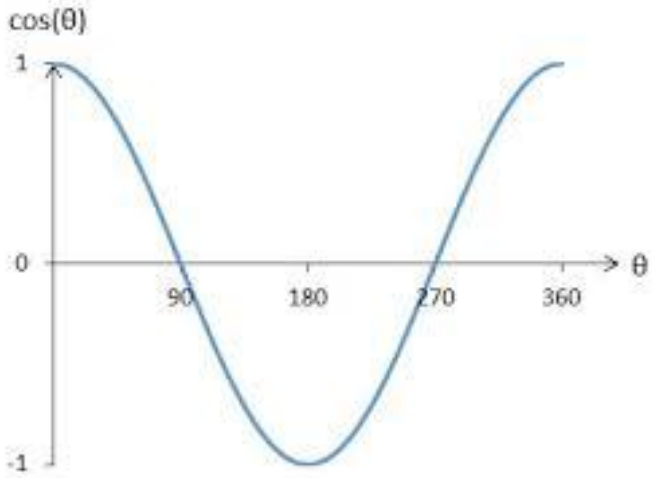
Modelli di IR: modello Vettoriale

Similarità fra i documenti dj e dk (uno dei quali solitamente è la query):

Prodotto Scalare

$$\underline{D1} * \underline{D2} = |\underline{D1}| * |\underline{D2}| * \cos(\theta) = \sum_i (w_{1,i} * w_{2,i})$$

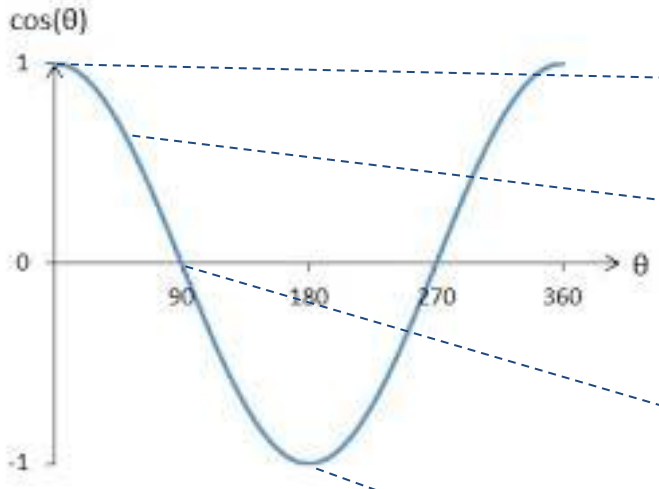


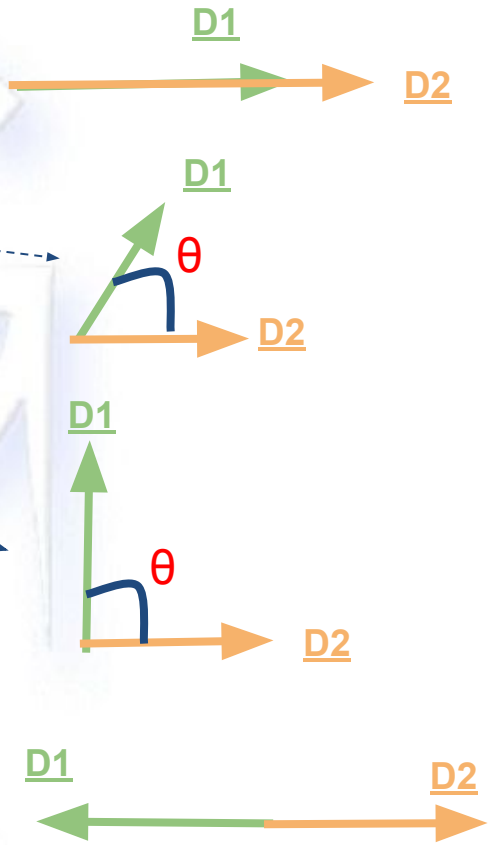


Prodotto Scalare = D1 * D2

Modelli di IR: modello Vettoriale

Similarità fra i documenti dj e dk (uno dei quali solitamente è la query):
Prodotto Scalare

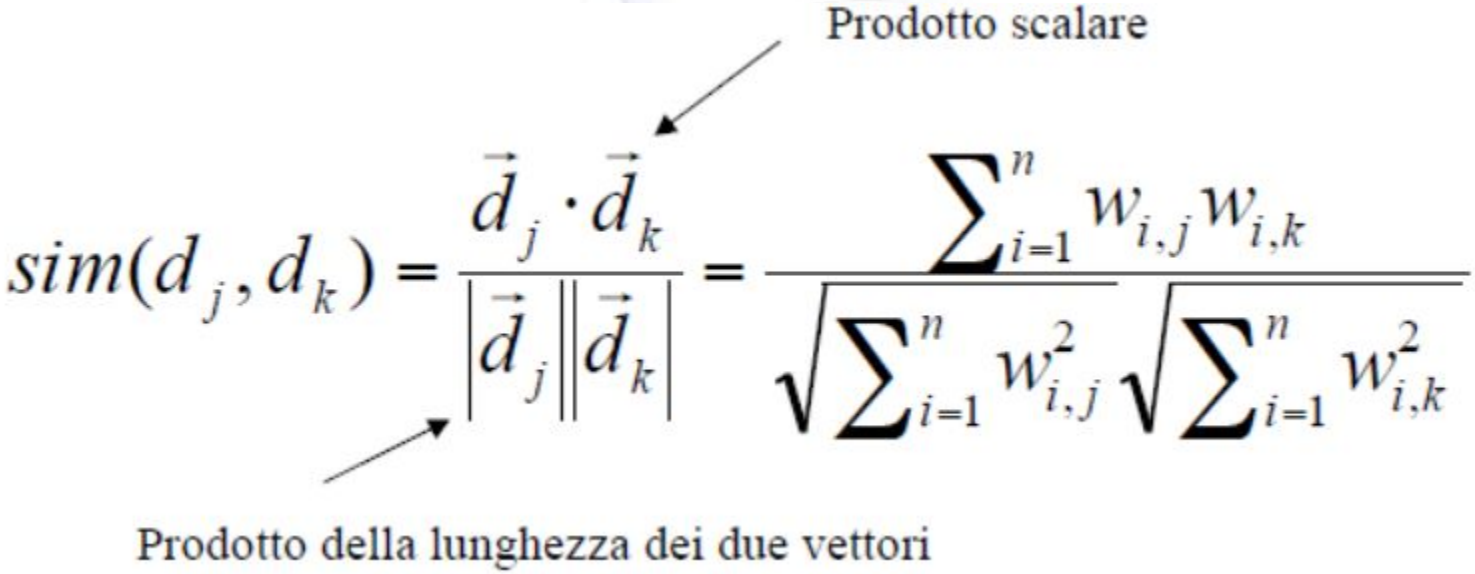




$$cos(\theta) = \sum_i (w_{1,i} * w_{2,i}) / | \underline{D1} | * | \underline{D2} |$$

Modelli di IR: modello Vettoriale

Similarità fra i documenti dj e dk (uno dei quali solitamente è la query):



Modelli di IR: modello Vettoriale

$$|Q \cap D|$$

Simple matching

$$2\frac{|Q \cap D|}{|Q| + |D|}$$

Dice's Coefficient

D=[1,2,3,4]

Q=[3,4,9]

$$\frac{|Q \cap D|}{|Q \cup D|}$$

Jaccard's Coefficient

|D ∩ Q| = [**3,4**]

|D ∪ Q| = [**1,2,3,4,9**]

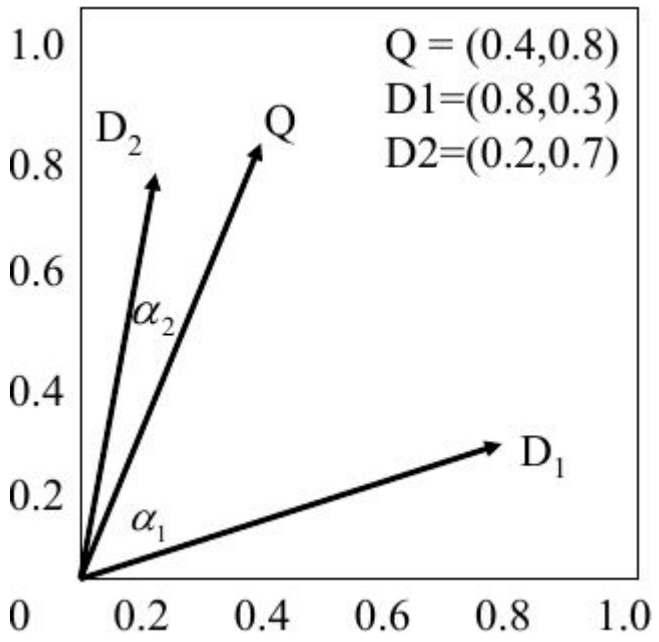
$$\frac{|Q \cap D|}{|Q|^{1/2} \times |D|^{1/2}}$$

Overlap Coefficient

$$\frac{|Q \cap D|}{\min(|Q|, |D|)}$$

Cosine Coefficient

Modelli di IR: modello Vettoriale



$$D_i=(d_{i1},w_{di1};d_{i2},\ w_{di2};\ldots;d_{it},\ w_{dit})$$
$$Q=(q_{i1},w_{qi1};q_{i2},\ w_{qi2};\ldots;q_{it},\ w_{qit})$$

$$sim(Q,D_i)=\frac{\sum_{j=1}^{t} w_{q_j} w_{d_{ij}}}{\sqrt{\sum_{j=1}^{t}(w_{q_j})^2 \sum_{j=1}^{t}(w_{d_{ij}})^2}}$$

$$sim(Q,D2)=\frac{(0.4\cdot 0.2)+(0.8\cdot 0.7)}{\sqrt{[(0.4)^2+(0.8)^2]\cdot[(0.2)^2+(0.7)^2]}}$$

$$=\frac{0.64}{\sqrt{0.42}}=0.98$$

$$sim(Q,D_1)=\frac{.56}{\sqrt{0.58}}=0.74$$

Modelli di IR: modello Vettoriale

Esempio

	Semi	Perdere	Olio
D1	1	1	1
D2	0	1	0
D3	0	1	1
Q	0	0	1

$$\text{Sim}(D1, Q) = \frac{1*0 + 1*0 + 1*1}{\sqrt{3} * \sqrt{1}} = .577$$

$$\text{Sim}(D2, Q) = \frac{0*0 + 1*0 + 0*1}{\sqrt{1} * \sqrt{2}} = 0$$

$$\text{Sim}(D3, Q) = \frac{0*0 + 1*0 + 1*1}{\sqrt{2} * \sqrt{1}} = .707$$

Modelli di IR: modello Vettoriale

I problemi principali di questo approccio sono:

- ➔ Olio appare due volte nel primo documento, il che può indicare una maggiore importanza della parola.

- ➔ Perdere appare in tutti i documenti, per cui non serve a discriminare tra più o meno rilevanti.

Invece di segnalare la presenza o meno di un termine in un documento, vogliamo dunque assegnare un peso.

Un approccio tipico consiste nell'utilizzare pesi ottenuti dal prodotto tra la **term frequency** (**tf**, ovvero, quante volte o in che percentuale un termine appare nel documento) e l'**inverse document frequency** (***idf***, ovvero, quanto è rara l'occorrenza di un termine).

Modelli di IR: modello Vettoriale

Per **Document Frequency** si intende il numero di documenti che contengono un certo termine.

L'inverso del Document Frequency (**Inverse Document Frequency, idf)**, cioè la rarità di un termine all'interno della collezione, è una buona misura della significatività di un termine.

Solitamente viene usata la seguente formula: $idf_i = log \frac{N}{n_i}$

Dove ***N*** è il numero totale di documenti della collezione e ***ni*** è il numero di documenti che contengono il termine ***i***.

Ad ogni termine della query viene assegnato un **peso** in base ad una misura combinata di ***tf*** e ***idf***

Modelli di IR: modello Vettoriale

Una misura del peso di un termine in un documento deve tenere conto di due fattori :
Quantificazione del peso che il termine ha nel documento:

$$tf(i,j) = \frac{freq_{i,j}}{\max_k \left(freq_{k,j} \right)}$$

Quanto il termine aiuta a discriminare il documento dj dagli altri documenti in D
➔ Fattore idf (inverse document frequency)

$$idf_i = \log \frac{N}{n_i}$$

Dove: ***N*** numero di doc. nella collezione, n_i numero dei documenti in cui il termine k_i appare

$$wij = tf(i,j) * idf(i)$$

Modelli di IR: modello Vettoriale

Salton e Buckley suggeriscono:

$$w_{i,q} = \left(0,5 + \frac{0,5\, freq_{i,q}}{\max\limits_{k}(freq_{k,q})} \right) \times log \frac{N}{n_i}$$

Modelli di IR: modello Vettoriale

Vantaggi:

- ➔ Il peso dei termini migliora la qualità delle risposte
- ➔ Poiché possono verificarsi matching parziali fra documenti e query, è possibile ottenere risposte che approssimino le richieste dell'utente
- ➔ La formula usata per misurare la similarità tra vettori consente di ordinare i documenti rispetto al grado di similarità con la query
- ➔ È possibile clusterizzare i documenti ed effettuare ricerche gerarchiche

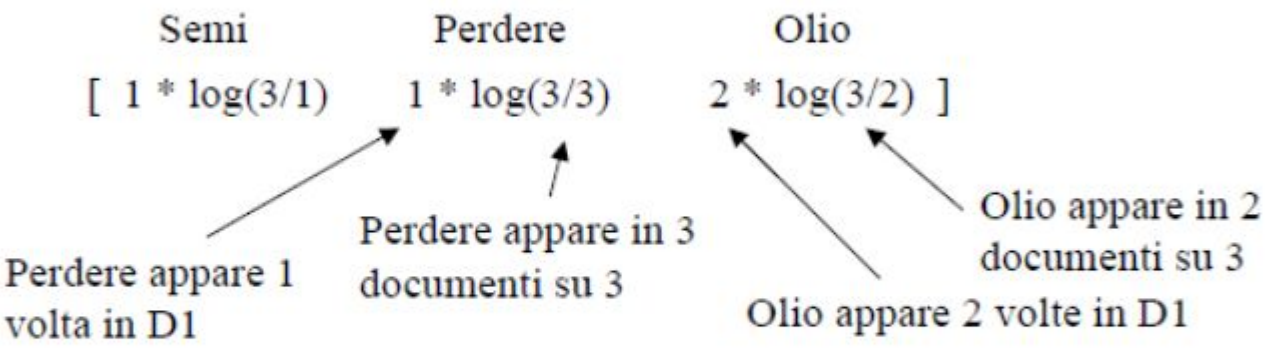
Svantaggi:

- ➔ Si basa sull'assunzione che i termini siano fra loro indipendenti. Ma non è provato che questo sia un vero svantaggio..

Modelli di IR: modello Vettoriale

Esempio:

D1 | Per **perdere** peso, usare **olio** di **semi** invece che **olio** di oliva.



D2 | Il lupo **perde** il peso, ma non il vizio…

D3 | La sua auto **perde olio**!

Modelli di IR: modello Vettoriale

Esempio:

	Semi	Perdere	Olio
D1	.477	0	.252
D2	0	0	0
D3	0	0	.176
Q	0	0	.176

$$\text{Sim}(D1, Q) = \frac{.447*0 + 0*0 + .252*.176}{.512 * .176} = .492$$

$$\text{Sim}(D2, Q) = \frac{0*0 + 0*0 + 0*.176}{0 * .176} = 0$$

$$\text{Sim}(D3, Q) = \frac{0*0 + 0*0 + .176*.176}{.176 * .176} = 1$$

Modelli di IR: modello Vettoriale

Esempio

	Semi	Perdere	Olio
D1	1	1	1
D2	0	1	0
D3	0	1	1
Q	0	0	1

$$\text{Sim}(D1, Q) = \frac{1*0 + 1*0 + 1*1}{\sqrt{3} * \sqrt{1}} = .577$$

$$\text{Sim}(D2, Q) = \frac{0*0 + 1*0 + 0*1}{\sqrt{1} * \sqrt{2}} = 0$$

$$\text{Sim}(D3, Q) = \frac{0*0 + 1*0 + 1*1}{\sqrt{2} * \sqrt{1}} = .707$$

Modelli di IR: modello Vettoriale

Problemi dell'approccio vettoriale:

- ➔ La lunghezza dei documenti incide sul calcolo della rilevanza.
- ➔ Il numero di documenti incide sul calcolo della rilevanza.
- ➔ Non viene considerato l'ordine dei termini.

CRITERI DI VALUTAZIONE

Un sistema tradizionale di **Data Retrieval** può essere valutato utilizzando svariate misure:

- ➔ Velocità di indicizzazione (numero di documenti indicizzati all'ora);
- ➔ Velocità di ricerca (in funzione della dimensione dell'indice);
- ➔ Espressività del linguaggio di interrogazione.

Tutte queste proprietà (performance evaluation) sono misurabili.

Ma la vera e più importante misura delle prestazioni di un motore di IR è un'altra: la "soddisfazione" dell'utente (retrieval performance evaluation), visto il meccanismo ranking based

Come misurare la soddisfazione di un utente? La velocità di ricerca è sicuramente un fattore importante, ma una risposta velocissima ma inutile non renderà felice l'utente!

CRITERI DI VALUTAZIONE

Come è possibile rispondere alla domanda “quale di questi due sistemi di IR funziona meglio”?

Un sistema tradizionale di Data Retrieval può essere valutato oggettivamente, sulla base delle performance (velocità di indicizzazione, ricerca ecc.).

In un sistema di IR tali valutazioni sono possibili, ma a causa della soggettività delle risposte alle query, le cose si complicano. Quello che si vorrebbe in qualche modo misurare è la “soddisfazione” dell‟utente.

Esistono delle misure standard per valutare la bontà delle risposte fornite da un sistema di IR: **precision** e **recall**.

CRITERI DI VALUTAZIONE

Misurare il grado di soddisfazione dell'utente tuttavia **non è cosa facile**, la scelta più valida infatti dipende dal tipo di utente e di applicazione:

Motore di ricerca: se un utente è soddisfatto delle prestazioni di un motore di ricerca tornerà ad utilizzarlo, quindi potremmo misurare la percentuale di utenti che "tornano";

Sito di e-commerce: dal punto di vista dell'utilizzatore del sito, il motore è valido se il tempo necessario per effettuare un acquisto è basso; dal punto di vista del proprietario del sito, il motore è buono se un'alta percentuale di ricerche si concludono con un acquisto;

Azienda: una valida misura può essere il tempo risparmiato dai dipendenti nella ricerca di informazioni.

CRITERI DI VALUTAZIONE

In generale allora il modo migliore per valutare un motore di IR è considerare la **rilevanza dei risultati**. Occorre definire una metodologia ed abbiamo bisogno di una serie di strumenti:

Una **collezione di documenti di test**; esistono diverse collezioni, tra cui ricordiamo la collezione TREC, sviluppata da NIST (National Institute of Standards and Technology): circa 700K documenti, dimensione 2GB

Un elenco di esempi di richieste di informazioni (query), solitamente definite informalmente, in linguaggio naturale (si parla perciò più precisamente di retrieval task), e da esperti del settore.

Una valutazione di rilevanza, cioè un giudizio rilevante / non rilevante per ogni coppia query / documento, definita da esperti del settore.

Data una strategia di IR, la misura della valutazione quantifica la similarità tra l'insieme dei documenti ritornati e l'insieme dei documenti classificati come rilevanti.

CRITERI DI VALUTAZIONE

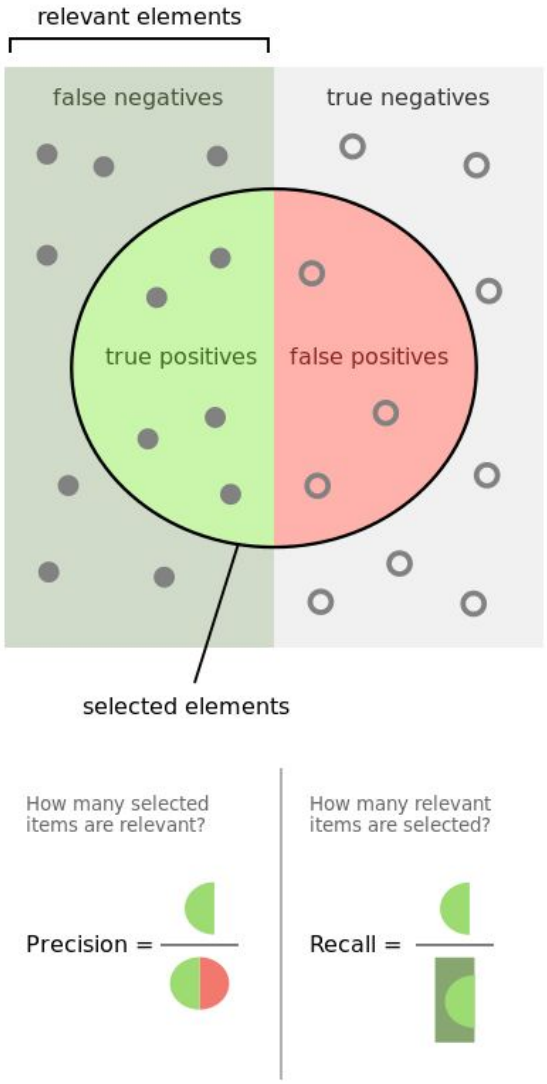
Scopo di un sistema di IR è di accedere efficientemente a tutti e soli i documenti rilevanti per una data interrogazione.
La bontà di un sistema di IR può essere valutata in vari modi. I criteri più accettati includono: **coverage**, **time**, **presentation**, **effort**, **precision** e **recall**.

- ➔ **Coverage** (copertura) rappresenta la completezza della collezione, cioè quanto del materiale disponibile e classificato e quindi potenzialmente ritrovabile.
- ➔ **Time** misura il tempo necessario per calcolare la risposta ad una interrogazione.
- ➔ **Presentation** rappresenta la bontà dell'interfaccia utente.
- ➔ **Effort** rappresenta lo sforzo, misurabile in termini di tempo, necessario all'utente per effettuare una interrogazione.

Precision e recall, sono i due parametri più noti e frequentemente utilizzati.

- ➔ **Precision** valuta la capacità di filtrare documenti non rilevanti in risposta ad una query;
- ➔ **Recall** valuta la capacità di recuperare documenti rilevanti.

CRITERI DI VALUTAZIONE



Precision

Quanti tra i documenti **recuperati** sono **rilevanti**?

Recall

Quanti tra i documenti **rilevanti** sono stati **recuperati**?

CRITERI DI VALUTAZIONE

$$N = T_p + T_n + F_p + F_n$$

$$P = \frac{T_p}{T_p + F_p}$$

$$R = \frac{T_p}{T_p + F_n}$$

	Rilevante	Non-rilevante
Recuperato	T_p	F_p
Non-Recuperato	F_n	T_n

In base ai quattro sottoinsiemi, è possibile calcolare due parametri molto usati per calcolare le prestazioni di un sistema di IR

PRECISION

➔ Rapporto tra il numero di documenti rilevanti reperiti (insieme T_p) e il totale dei documenti reperiti (insiemi T_p e F_p)

RECALL

➔ Rapporto tra il numero di documenti rilevanti reperiti (insieme T_p) e il totale dei documenti rilevanti (insiemi T_p e F_n)

CRITERI DI VALUTAZIONE

$$N = T_p + T_n + F_p + F_n$$

$$P = \frac{T_p}{T_p + F_p} \quad R = \frac{T_p}{T_p + F_n}$$

$$F = \frac{(\beta^2 + 1) * P * R}{\beta^2 * P + R} \quad F_1 = \frac{2 * P * R}{P + R}$$

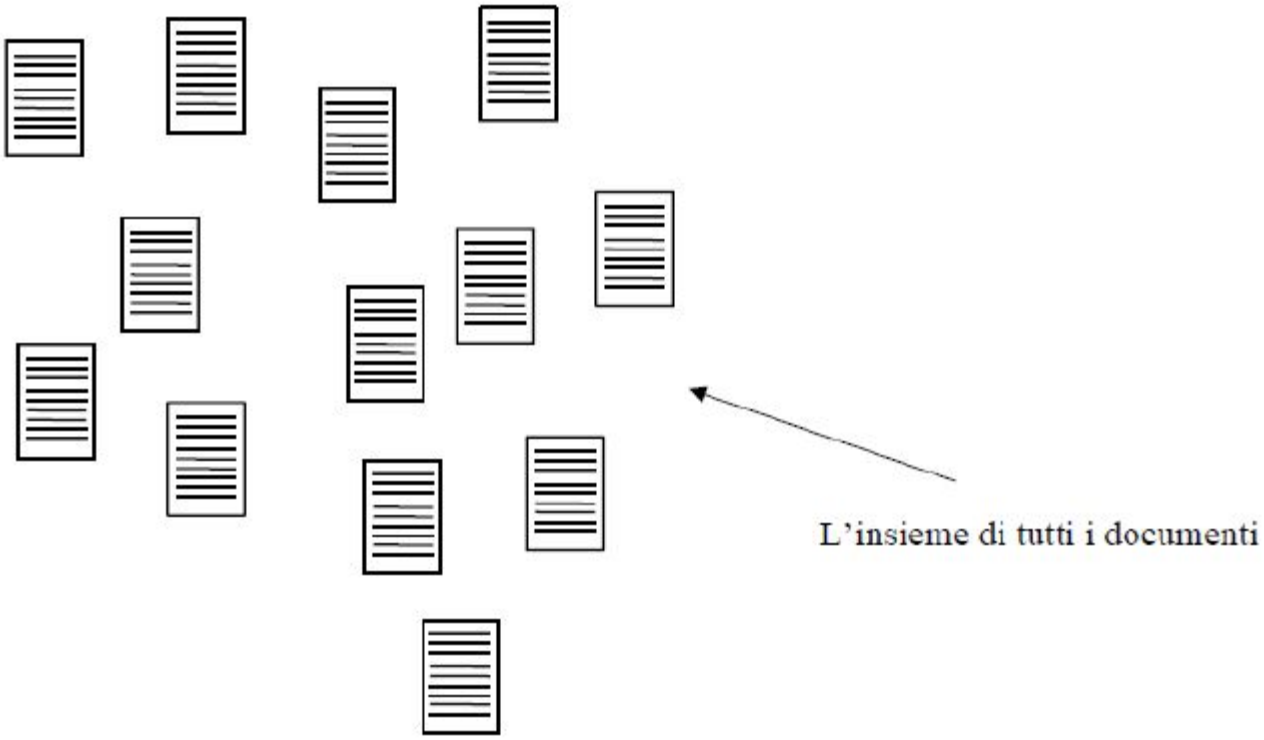
	Rilevante	Non-rilevante
Recuperato	T_p	F_p
Non-Recuperato	F_n	T_n

Combinando precision e recall, si ha **F**, che è una misura combinata che bilancia l'importanza di **Precision** e **Recall**.

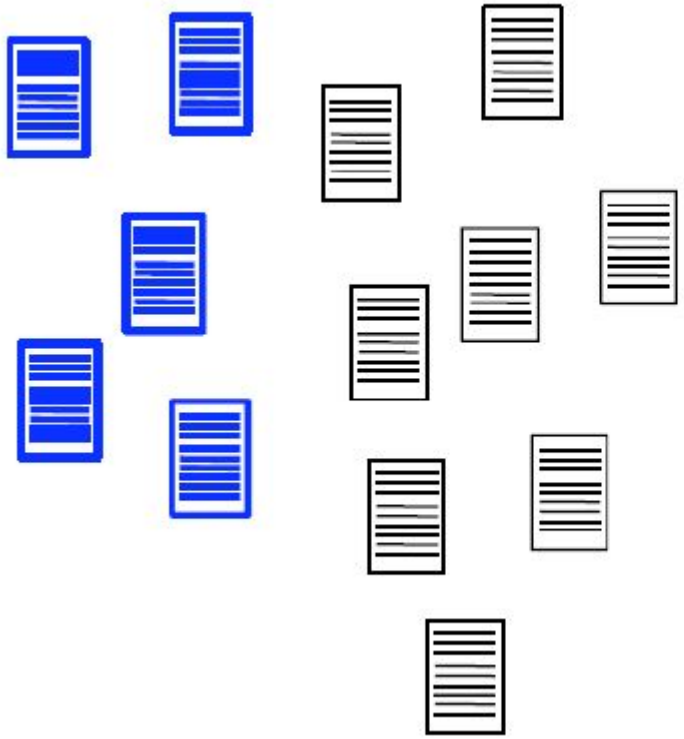
Solitamente viene usata la **media armonica** (il reciproco della media dei reciproci) tra Precision e Recall **F1** (cioè con β = 1).

In questa misura combinata si assume che l'utente bilanci l'importanza di Precision e Recall.

Precision e Recall

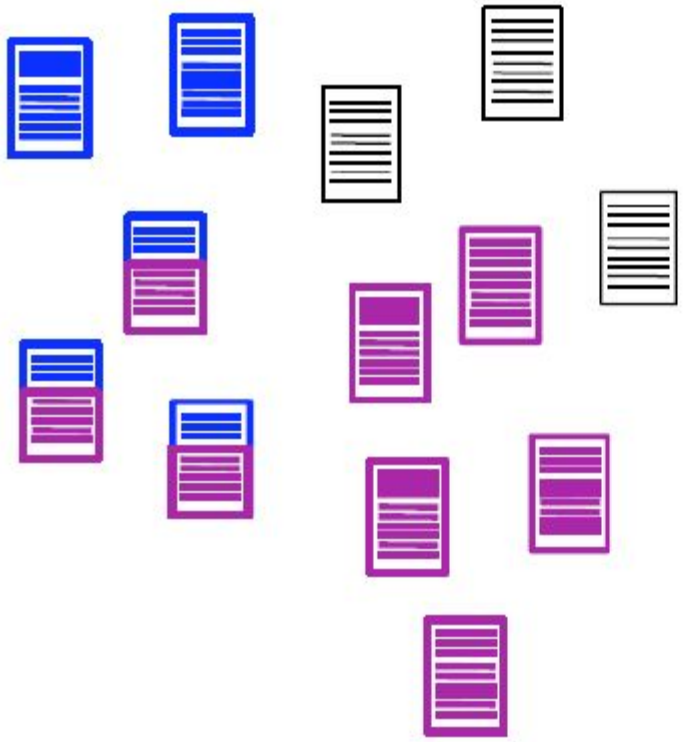


Precision e Recall



DOCUMENTI RILEVANTI

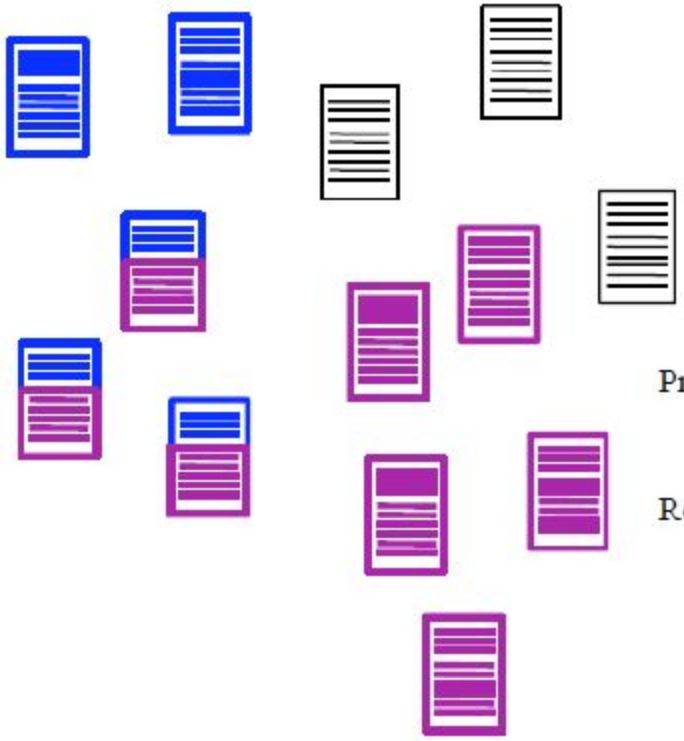
Precision e Recall



DOCUMENTI RILEVANTI

RISULTATO DELLA QUERY
(DOCUMENTI RITORNATI)

Precision e Recall



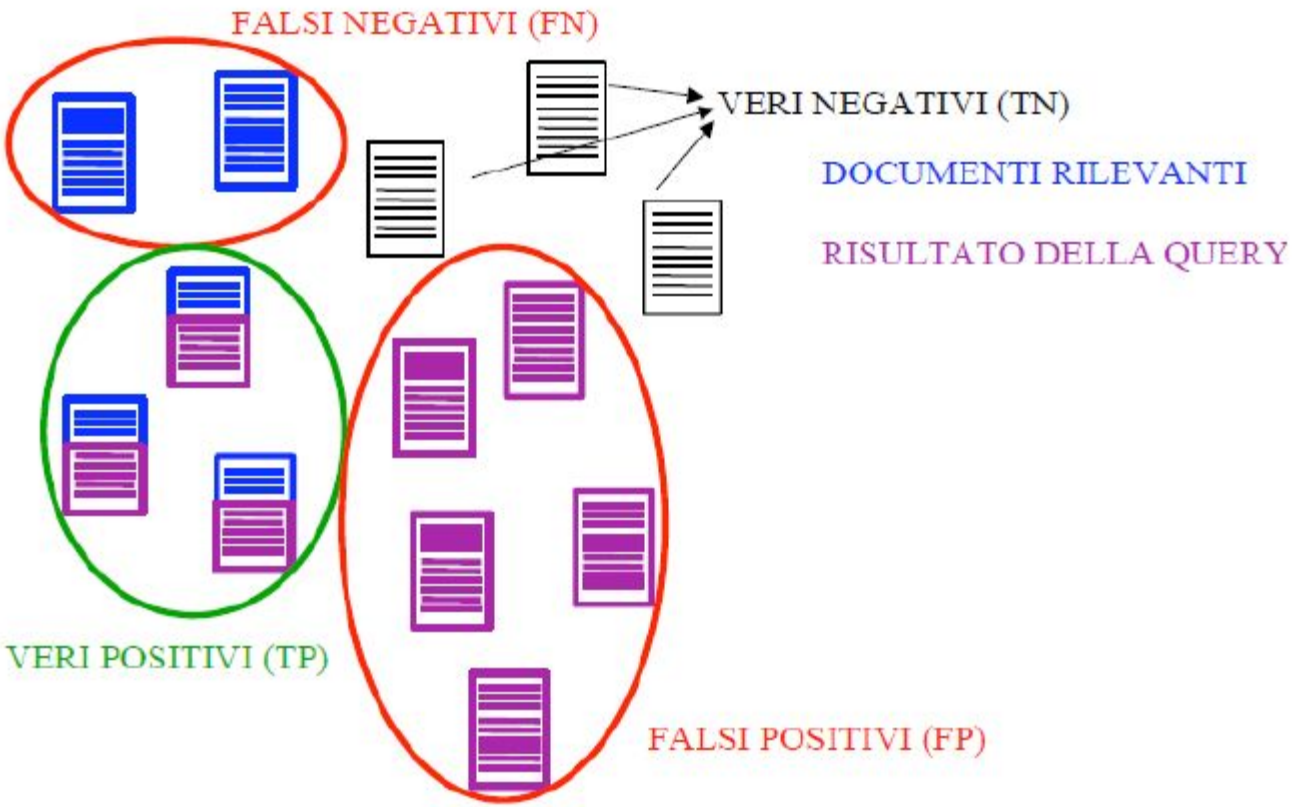
DOCUMENTI RILEVANTI

RISULTATO DELLA QUERY
(DOCUMENTI RITORNATI)

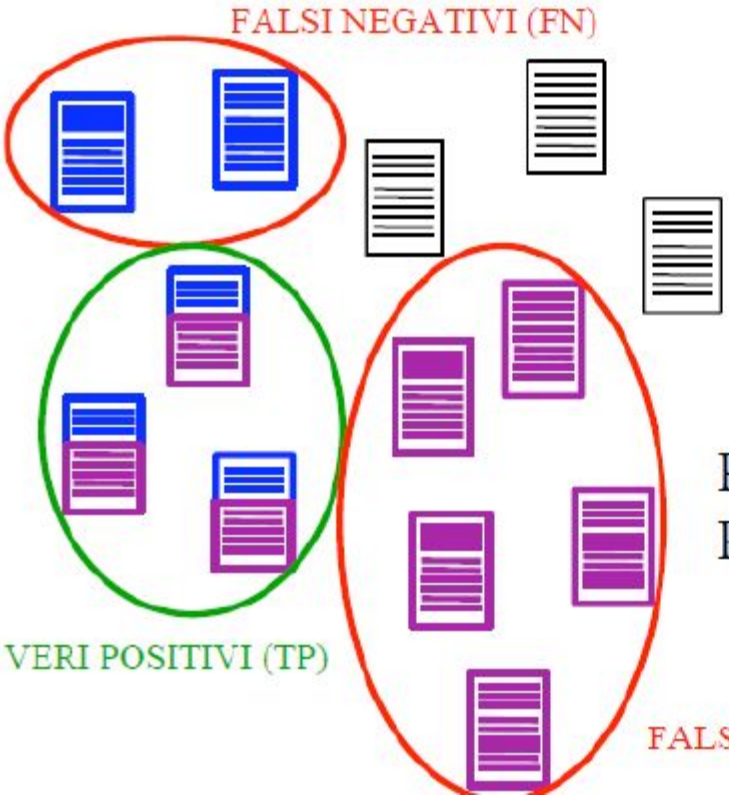
$$\text{Precision} = \frac{|\{\text{Rilevanti}\} \cap \{\text{Ritornati}\}|}{|\{\text{Ritornati}\}|} = 3/8$$

$$\text{Recall} = \frac{|\{\text{Rilevanti}\} \cap \{\text{Ritornati}\}|}{|\{\text{Rilevanti}\}|} = 3/5$$

Precision e Recall



Precision e Recall



DOCUMENTI RILEVANTI

RISULTATO DELLA QUERY

Precision = TP / (TP + FP) = 3/8

Recall = TP / (TP + FN) = 3/5

Precision e Recall

Perché utilizzare due misure?

Se considerassimo solo la misura **Recall**, un motore che (in risposta a qualsiasi query) restituisce tutti i documenti della collection sarebbe considerato perfetto (Recall = 1).
Se considerassimo solo la misura **Precision**, un motore che restituisce solo un documento (che sicuramente viene considerato rilevante) sarebbe considerato perfetto (Precision = 1).
Precision e Recall, considerate come due funzioni del numero di documenti restituiti e hanno un andamento opposto:

- ➔ Recall è non decrescente;
- ➔ Precision è solitamente decrescente.

Usando questi comportamenti, nel valutare le prestazioni di un motore di ricerca che restituisce un elenco di documenti ordinati per similarità con la query, possiamo valutare Precision e Recall a vari livelli di Recall, cioè variando il numero di documenti restituiti.
Otteniamo così delle **curve di Precision e Recall**

CRITERI DI VALUTAZIONE

In conclusione, l'utilizzo di **Precision** e **Recall** per la valutazione di un motore di IR pone alcuni problemi:

- ➔ I documenti della collezione devono essere valutati manualmente da persone esperte: non sempre il giudizio è completamente veritiero;
- ➔ La valutazione dei documenti è binaria (rilevante / non rilevante): non sempre è facile catalogare così nettamente un documento;
- ➔ Le misure sono pesantemente influenzate dal dominio di applicazione, cioè dalla collezione e dalle query: un motore di IR potrebbe avere delle ottime prestazioni in un determinato dominio ma non in un'altro.

Esercitazioni: 1-3

https://github.com/marcoortu/WAAT-2020

branch: *01-esercitazione*
branch: *02-esercitazione*
branch: *03-esercitazione*