Dipartimento di Informatica dell'Università di Bologna

Corso di laurea Magistrale in Informatica

Appunti digitali di

Basi di Dati

Autore:
Lorenzo Balugani (@EvilBalu)

Versione: 231221
Istanza Erre2: erre2next.fermitech.info

Anno Accademico 2020-2021

0 – Prefazione

0.1 – Avvertenze

Questo documento contiene gli appunti personali dell'autore, distribuito tramite licenza CC BY-SA 4.0. L'autore non è parte del corpo docente, o in alcun modo legato ad esso. L'autore è uno studente, e questo testo non deve venire considerato come sostitutivo ad un buon metodo di studio oppure alle spiegazioni e materiale fornito direttamente da parte dei docenti. Questi appunti non sono e non vogliono essere dispense.

Questo testo può venire ridistribuito, in accordo con quanto concesso dalla licenza CC BY-SA 4.0:

- Deve venire fornito credito all'autore originale del documento, e indicare se (e quali) modifiche sono state apportate.
- Eventuali alterazioni di questo documento devono venire ridistribuite sotto la medesima licenza.

In aggiunta:

- Questa pagina deve essere sempre presente in ogni riassunto contenente il logo di Erre2 che viene distribuita tramite quel sistema, e non deve venire modificata in alcun modo.
- Questi documenti non devono essere collocati dietro un paywall.

Al fine di garantire l'accuratezza delle informazioni contenute, si prega l'utente di rendere note eventuali incorrettezze o errori all'interno del documento tramite l'istanza Erre2 da cui si è scaricato oppure tramite eventuali handle lasciati dall'autore.

Si chiede, inoltre, di rendere noti eventuali abusi o utilizzo fraudolento dei riassunti e delle istanze Erre2.

0.2 – Planetario ed Erre2

Erre2 è un servizio per la condivisione, senza paywall, di appunti universitari, sviluppato da Fermitech Softworks e rilasciato sotto licenza GPL v3. Tramite il servizio "Planetario", è possibile tenere traccia di tutte le istanze – verificate – di Erre2, qualora studenti vogliano creare proprie istanze. Tramite il Planetario è possibile, quindi, ispezionare le "Costellazioni" di Erre2, accedendo alle istanze che rispettano determinati criteri di valutazione. Per maggiori informazioni, visitare http://www.planetario.fermitech.info/erre2.

Nel caso in cui istanze di Erre2 vengano utilizzate per scopi illegali, e risultino all'interno del Planetario di Fermitech Softworks, si prega di aprire un ticket con l'assistenza per avviare una verifica. Qualora vi fosse stato chiesto denaro al fine di fruire di questo riassunto, aprire un ticket presso l'assistenza del Planetario.

Data ultimo aggiornamento: 20/9/2021

1 – Introduzione

Il corso ha come obiettivo quello di comprendere e di applicare i modelli di gestione di dati strutturati e semi-strutturati (e relativi linguaggi di query), e comprendere le tecniche analitiche di analisi di dati strutturati (data warehousing, data mining).

Sistemi di gestione di dati relazionali sono impiegati da numerose applicazioni, e il modello relazionale nasce dall'esigenza di gestire dati di tipo amministrativo. Negli ultimi 10 anni si è registrato un incremento di dati non relazionale (video, immagini), e questa crescita è legata a nuovi dispositivi (fotocamere digitali, cellulari...) e ad internet, che consentono all'utente non solo di essere fruitore di contenuti ma anche produttori.

Questi dati hanno proprietà che non hanno una corrispondenza con dati relazionali (nessuno si sogna di salvare foto e video all'interno di un database relazionale). Contenuti di questo tipo, inoltre, sono quelli più prodotti e consumati dagli utenti.

Per **dato strutturato** si fa riferimento ad un informazione "formattata" in un certo modo, che si rifà ad uno schema di riferimento (ad esempio, il modello di una tabella in una base di dati).

Al fine di poter gestire questa mole di dati, si rende quindi necessario la costruzione di sistemi per la gestione di dati semi-strutturati (dati con struttura parziale, non è presente una struttura rigida nella formattazione ma si sa che campi possono avere, il linguaggio più noto è l'XML). Dati non strutturati sono, invece, dati che non hanno alcuna formattazione che permetta di analizzarli o gestirle con pratiche standard, e se ne occupa l'Information Retrieval.

Nel modello relazionale vi è una netta distinzione tra schema e dati (nei semi-strutturati si ha uno schema parziale con le stesse proprietà dei dati), è basato sul concetto di insieme (piuttosto che su liste), non sono ordinati (piuttosto che ordinati, ovvero c'è sempre un concetto di ordine all'interno di una lista) e non sono incapsulabili (a differenza dei semi-strutturati, che non sono 'piatti').

1.1 – XML

XML è un linguaggio di markup che nasce come una involuzione di SGML (creato da IBM per garantire la comunicazione tra applicazioni), che rende più facile e leggero il progetto originale (e per questo si affermerà come standard de facto), e viene usato per rappresentare dati semi-strutturati, usando la sua flessibilità nel rappresentare dati e metadati.

Il problema si pone quando si cercano altri spazi (oltre che ad un filesystem) in cui memorizzare queste informazioni: le informazioni perdono la capacità di essere comprensibili ad esseri umani e applicazioni (senza ulteriori elaborazioni), e alcune interrogazioni non possono essere scritte in sql senza essere inefficienti o ricorsive. Le strade percorribili sono quindi rendere sistemi relazionali "compatibili" con XML tramite estensioni o costruire sistemi ad hoc per XML.

1.2 – Limiti del modello relazionale

Uno dei limiti del modello relazionale sta nel fatto che lo schema diventa immutabile (o quasi), e per i dati semi-strutturati (e non strutturati) sono soggetti a non adattarsi ad un determinato schema: dati che sono soliti a subire variazioni nella loro struttura non si abbinano bene al modello relazionale.

Nei **dati semi-strutturati** si definisce contemporaneamente il formato e il contenuto (in un certo senso la definizione dello schema, o per meglio dire **data guide**, viene terminata quando il dato è stato completamente inserito, quindi non noto a priori), e di conseguenza possono sorgere problemi quando si tratta di inserirli all'interno di un modello relazionale.

Le data guide sono prone ad evolvere quindi rapidamente, insieme ai dati, e sono irregolari e parziali, con la separazione tra dati e "schema" che è praticamente inesistente (le interrogazioni possono riguardare anche lo schema), e che non impone alcun vincolo sui contenuti.

1.3 – Dati non strutturati

XML è un linguaggio di mark-up per dati semi-strutturati, e pur essendo vago in questi dati esiste uno schema. Nel caso di file multimediali (ad esempio), lo schema non esiste, e i dati risultano essere non strutturati. La disciplina che studia dati non strutturati prende il nome di Information Retrieval.

Una possibile "interrogazione" può riguardare una ricerca di tutti i documenti che contengono una specifica parola, molto diversa da un'interrogazione SQL (sintassi semplice vs sintassi complessa dell'SQL).

Le interrogazioni relazionali esprimono precisi requisiti, e la costruzione di una risposta segue un modello booleano (una tupla è presente o meno nel risultato), e questo modello nell'IR non si applica in quanto data la natura delle domande, la vastità dei dati e delle risposte (e alcuni documenti possono rispondere alla stessa domanda). Dal momento che i dati sono ambigui e c'è una differenza tra dati ed informazioni, non è sempre possibile determinare se un risultato sia completamente (o non) rilevante: i risultati vengono ordinati in base a un rank di rilevanza, e sta all'utente tenere in mente la possibilità di errori. I risultati più in alto vengono in oltre "cliccati" più facilmente, e sono quindi estremamente importanti.

Rispetto al relazionale, dati non strutturati non hanno schema, usano un linguaggio di ricerca, fanno uso di un sistema di ranking ed eseguono aggiornamenti totali al posto di parziali. Per migliorare le performance dell'algoritmo di ranking si può eseguire profilazione dell'utente, al modo di riuscire a fornire con maggiore efficacia elementi che rispondano alle esigenze del richiedente.

I DBMS includono tecnologie in grado di eseguire operazioni di indexing di colonne contenenti solo testo (CLOB) o colonne per dati multimediali (BLOB).

2 – XML

XML nasce come formato per lo scambio dati, derivato da SGML, che viene utilizzato anche per stoccare informazioni. Un file XML è composto da una serie di costrutti, i quali definiscono i metadati all'interno dei quali vengono inseriti dati. Il documento è diviso in due parti, il prologo (informazioni relative all'opzionale dichiarazione del formato xml, una grammatica opzionale e commenti e informazioni per applicativi che usano il documento, dette Processing Instructions) e il corpo, costituito dall'elemento <doc></doc>.

Un elemento può essere scritto come <nome [lista_attributi]> [contenuto] </nome>, oppure <nome [lista_attributi]/>, ed è fondamentale che ogni elemento sia contenuto dentro un tag di apertura e uno di chiusura (i tag vanno sempre chiusi, e sono case-sensitive). Altro elemento importante è

ricordarsi di usare le virgolette quando si specifica il valore di attributi. Se rispetta queste condizioni, il documento è ben formato.

Un elemento caratteristico di XML è il DTD (document type declaration), ovvero una dichiarazione che contiene una grammatica che consente di limitare e completare il documento: è composto anch'esso da una serie di dichiarazioni di markup che indicano cosa può e non può essere scritto nel documento (oltre che valori predefiniti e altri vincoli). Un documento XML è valido se contiene un DTD e se è aderente alle specifiche (un documento ben formato può non essere valido).

```
<!DOCTYPE test [
<!ELEMENT test (#PCDATA)>
<!ATTLIST test id ID #REQUIRED>
]>
```

È possible, inoltre, archiviare il DTD in un altro file e richiamarlo altrove (<!DOCTYPE test SYSTEM "test.dtd">).

XML Schema consente di definire vincoli più complessi rispetto al DTD, e i vincoli dello schema vengono espressi in XML. Al fine di costruire XML per immagazzinare dati, bisogna:

- I dati devono essere immagazzinati nel contenuto degli elementi, non come loro attributi;
- Vanno evitati metadati in attributi o nomi di elementi;
- Vanno utilizzate correttamente le gerarchie;

2.1 Dati relazionali e XML

Nel combinare dati relazionali e XML è necessario definire come da SQL si passa a XML e viceversa. L'uso principale di XML è quello di consentire lo scambio di informazioni tra applicazioni, le quali utilizzano a loro volta un database di tipo relazionale.

Un primo approccio di interfacciare i due mondi è SQL/XML, che consente di eseguire il passaggio da SQL a XML (da strutturato a semi-strutturato): fare questo è relativamente semplice, dal momento che perdere struttura è più facile che imporne una (tutto ciò che viene memorizzato in una tabella può venire rappresentato in XML). Se soluzioni possibili sono:

1. Si esegue il mapping della tabella in xml (si estraggono i dati e vengono rappresentati in xml, e con lo schema si stabiliscono i vincoli) e si usa XQuery per farci ricerche sopra;
2. Si estraggono i dati dalla tabella con SQL e poi il risultato viene trasformato in XML tramite SQL/XML (tramite XMLElement), che è un'estensione di SQL;

Per fornire la funzionalità di salvataggio in DB relazionali di XML, diversi sistemi usano diverse soluzioni:

- Usare colonne Object-Relational per memorizzare XML in un unico campo;
- Immagazzinare diversi elementi del documento XML in campi diversi (Shredding);

2.1.1 SQL/XML

SQL/XML è un'estensione costituita da costruttori e funzioni in grado di supportare manipolazione e archiviazione di XML in un database SQL. Definisce diversi costruttori, che vanno posti alla fine di una clausola SELECT (inizialmente viene elaborato il select, poi viene costruito per ogni tupla l'xml):

- XMLElement
    - Consente di creare elementi XML, richiedendo come argomenti il nome dell'elemento, una lista di attributi e il contenuto dell'elemento.
    - SELECT i.id, XMLELEMENT(NAME "emp", i.name) AS result FROM EMPLOYEES i;
- XMLAttributes
    - Consente di creare una lista di attributi di un elemento, e può venire messo all'interno di un XMLElement;
- XMLForest
    - Consente di creare una lista di elementi, comportandosi come XMLAttributes;
- XMLConcat
    - Concatena elementi;
- XMLAgg
    - Usato per aggregare tuple;
- XMLGen
    - Crea XML inserendo variabili tra {};

Il contenuto di un elemento può venire costruito come concatenazione di stringhe (eseguita con ||).

2.2.2 – Xquery

XQuery è un linguaggio di query per dati nel formato XML, e può essere usato per accedere a documenti strutturati e semi strutturati. XQuery opera su sequenze, che possono contenere valori atomici oppure nodi: una espressione XQuery riceve zero o più sequenze e produce una sequenza.

Le sequenze sono ordinate, non incapsulate (1, (2,3) è uguale a (1,2,3)) e non c'è differenza tra un elemento della sequenza e l'elemento stesso ((1) = 1). La virgola e il to consentono di definire sequenze (1, 2, 3) = (1 to 3), mentre union (|), intersect e except permettono di eseguire operazioni a livello di insiemi (unione, intersezione, esclusione). Le sequenze oltre agli atomi possono anche contenere nodi, che vanno a comporre l'albero che rappresenta il documento XML. Due nodi testuali non possono essere adiacenti, quindi il contenuto di <name> Lorenzo Balugani </name> verrà messo come figlio del nodo "name".

Una **Path Expression** è un espressione che può essere usata per estrarre valori da nodi XML, e controllarne le proprietà, e anch'esse sono sequenze, i cui step sono separati da /.

Ogni step è valutato in uno specifico contesto, che dipende dal passo precedente. Lo step di una path expression è costituito da tre parti: una sezione (axe) che seleziona i nodi in base alla posizione (ad esempio, child::), un test che filtra i nodi in base al nome e al tipo (name) e uno o più predicati per filtrare ulteriormente. Ad ogni passo è possibile avere uno o più predicati inclusi tra parentesi quadre.

Le path expression possono anche iniziare con dei prefissi (/ corrisponde ad una sequenza di input che contiene la radice dell'albero, // corrisponde ad una sequenza di input dell'espressione che contiene tutti i nodi del documento)

Espressioni FLWOR

Una espressione FLWOR è composta da 5 parti:

- For: associa una o più variabili all'espressione;

- Let: crea un alias dell'interno risultato di un'espressione;
- Where: filtra le associazioni;
- Order By: ordina le associazioni;
- Return: crea il risultato dell'espressione

I commenti possono venire lasciati se inseriti tra (: :).

```
for $b in doc("file.xml")/bib/book return
<book year="{$b/@year}">{$b/title}</book>
for $i in $employees
order by $i/title
return $i
```

2.2.3 XQuery e DBMS

I DBMS che vengono presi in considerazione sono DB2, Oracle XML DB e MS SQL Server 2012. DB2 consente di inserire colonne di XML, e si possono inserire dati usando INSERT combinato con XMLPARSE (prende in input una stringa UTF-8 e prova a convertirla in xml), e si può usare XMLQUERY per eseguire query si queste colonne.

```
CREATE TABLE employees (data XML); //Permette di creare tabelle contenenti dati XML
CREATE TABLE departments (data XML);
INSERT INTO departments(data) VALUES (XMLPARSE(document cast('<depts><dept deptno="10"
dname="Administration"/>…</depts>' as clob) preserve whitespace)); //Inserimento e parsing dell'XML,
con mantenimento degli spazi Bianchi
SELECT XMLQUERY ('for $d in $list//dept return <deptName>{ $d/@dname }</deptName>' passing dtable.data
as "list") FROM departments dtable;
SELECT XMLQUERY ('for $e in $list//emp
                  let $avgsal := avg($list//emp[@deptno = $e/@deptno]/@salary) return
<averages>{$e/@ename}{$e/@salary}{$avgsal}</averages>' passing etable.data as "list")
FROM employees etable;
```

Oracle DB fornisce supporto completo a XQuery, condividendo molte caratteristiche con DB2. Non sono supportate le funzionalità del version encoding, l'xml:id, l'xs:duration, la validazione dello schema e la funzionalità del modulo. L'XMLType è un costrutto astratto, che viene implementato o come LOB (Large Object, simile ad una stringa) oppure come Structured Storage (il DBMS costruisce tabelle e viste che seguono lo schema XML).

In Oracle Berkeley l'oggetto fondamentale è il container (paragonabile alle tabelle), e una volta creato è possibile inserirvici documenti XML (e da questi si possono estrarre informazioni con query apposite). I dati nel container possono venire memorizzati per documento oppure per nodo (se il documento ha tante modifiche parziali conviene la seconda). Per inserire documenti nel container si usa l'operazione "putDocument book '[XML]' param", dove i parametri possono essere f (al posto di xml ci si mette il nome di un file xml) o s (XML è una stringa xml). Per recuperare documenti in un container, si devono scrivere query XQuery, con la sintassi di seguito riportata:

```
query 'for $book in collection("esempio.dbxml") where $book/price > 100 return $book/title
```

Durante la creazione di un container è possibile associare uno schema XML per fare enforcing di validità dei documenti presenti in esso, ed è possibile specificare diversi vincoli per diversi documenti.

3 – NoSQL

Una base di dati relazionale è basata su un modello relazionale altamente strutturato (colonne, righe, tipi di dati), con un modello di query standardizzato, le possibilità di fare join creando così nuove viste e le proprietà ACID (atomicità, consistenza, isolamento, resistenza).

Con il ruolo dell'utente che cambia da consumatore a produttore di informazioni, lo scenario è cambiato portando alla nascita dei Big Data (Volume, Velocità, Varietà, Variabilità, Veracità), che origina la necessità di flessibilità (per variabilità e velocità), di sistemi per distribuire i dati (volume) e che garantiscano altra disponibilità (velocità). In questo scenario, i join non sono scalabili, le transazioni risultano lente a causa di risorse bloccate, lo schema risulta essere troppo stringente con dati altamente variabili (risulta complesso creare report basati su dati semi (o non) strutturati). NoSQL non indica delle caratteristiche, semplicemente indica che un certo DBMS non è SQL (in parte o completamente). Sistemi NoSQL possono (almeno una delle due opzioni) non utilizzare ACID (si dice quindi BASE, ovvero tollera fallimenti parziali, con lo stato del db che può cambiare e diventa consistente nel lungo periodo) oppure non possedere uno schema (modello dati non relazionale).

Il sistema si dice **Basically Available** se non garantisce la disponibilità dei dati, con l'idea che ad ogni richiesta ci sarà una risposta, che possibilmente fallirà con dati inconsistenti e in cambiamento.

Viene definito **Soft State** uno stato in cui il sistema potrebbe cambiare nel tempo, anche come conseguenza di una ricerca di inconsistenti.

Con **Eventual Consistency** si fa riferimento al fatto che un sistema raggiungerà la consistenza non nell'immediato, e che questi cambiamenti verranno propagati: il sistema non controlla la consistenza di ogni transazione prima che si muova alla successiva.

Non è possibile avere le proprietà ACID e BASE contemporaneamente, bisogna al limite fare dei compromessi (triangolo CAP, scegli 2 tra consistenza/disponibilità/tolleranza ai guasti).

Il meccanismo di Aggregazione consente di aggregare oggetti del dominio di utilizzo come una sola unità (un documento visto come un aggregato di elementi, provenienti magari da "tabelle relazionali" diverse).

I modelli dati possono essere a documento (MongoDB), a grafo (Neo4j), a colonna (Cassandra) o a chiave/valore (Redis). Ogni modello serve per risolvere un problema specifico, ad esempio un sistema chiave/valore può essere usato per gestire la sessione di navigazione, mentre la gestione degli ordini può venire fatta da un sistema a documento, mentre l'inventario viene gestito da un normale DBMS relazionale.

Nel modello chiave/valore ad una chiave viene associato un corpo che può contenere JSON o altro (ad esempio un BLOB), e possono essere memory oppure disk resident: dal punto di vista implementativo operano come una tabella hash e i dati non hanno formato predefinito, e possono venire inseriti, letti aggiornati o cancellati.

Nel modello document si basano su documenti XML, JSON e hanno una struttura ad albero: possono essere visti come sistemi chiave/valore, dove la chiave è un nodo. Sistemi come MongoDB permettono

di passare da relazionale a documenti rapidamente. Il beneficio del sistema è che il modello dati è naturale, i dati possono venire aggregati ad una sola struttura e lo schema è dinamico.

Nel modello a colonne, le tabelle vengono memorizzate per le colonne al posto che per righe, supportando set variabili di colonne e ottimizzato per operazioni tra colonne, consentendo di memorizzare molteplici colonne per chiave.

Nel modello a grafo viene fatto uso di nodi, relazioni e proprietà dei grafi (modello dati property graph), permettendo di usare algoritmi per grafi sui dati, garantendo le proprietà ACID.

4 – Information Retrieval

L'Information Retrieval (IR) è l'operazione di trovare materiali di tipo non strutturato che soddisfano un bisogno informativo all'interno di grandi moli di dati (ricerca tra email, ricerca tra file, librerie digitali, ricerca web ...). L'IR definisce il documento come un file non strutturato, una collezione come un insieme di documenti (statici, non ipertestuali per ora), una query come un insieme di parole chiave che esprimono l'informazione e l'obiettivo è quello di ottenere i documenti contenenti informazioni rilevanti con la query, che viene passata al motore di ricerca, che presenta risultati e dai risultati viene raffinata la query, che viene ripassata al motore di ricerca per produrre risultati migliori.

Un documento può essere una pagina web, email, libri, pdf ..., e tutti contengono testo, e sono o completamente non strutturati o solo vagamente: questo non le rende compatibile con DBMS in cui i dati seguono una struttura ben definita, ed è necessario un sistema in grado di comprendere query in linguaggio naturale. Il linguaggio di query di un IR può offrire risultati diversi a seconda di come viene implementato, e propone una lista (ordinata per rank, ovvero dal più al meno pertinente) di documenti. Comparare il testo della query, lettera per lettera, non è sufficiente, ed è importante usare un sistema in grado di comprendere la rilevanza dei documenti, al posto di un match esatto.

I modelli IR sono:

- Modelli classici
    - Modello booleano (no rilevanza)
        - La query è un'espressione booleana composta da keyword legate da operatori logici, e ogni documento e richiesta viene visto come un insieme di parole. Una ricerca produce una matrice di incidenza, che indica quali documenti contengono una certa parola (ogni documento ha un vettore associato). Per migliorare i problemi legati allo spazio, vengono memorizzate solo le posizioni nella matrice diverse da 0. Sistemi booleani sono considerabili buoni solo se l'utente è esperto, e non funzionali per la maggior parte degli utenti. Query booleane potrebbero dare o troppi risultati o nessuno (l'AND è troppo stringente, OR troppo lasco).
    - Modello a spazio vettoriale
        - È il sistema di ranking più noto, dove un sistema di ranking consente di ordinare il risultato (in ordine decrescente di rilevanza), in modo da rimuovere i problemi di dimensione dei risultati (mostrando solo la top k, ovvero i primi 10 risultati), con la premessa che il modello di ranking funzioni.

        - I documenti (bag of words) sono rappresentati come un vettore di "pesi di termini", e fa la stessa cosa con una query, all'interno di uno spazio che ha tante dimensioni quanti i termini (spazio $|V|$ dimensionale, V numero dei termini). È un sistema ad alta dimensionalità e ci sono molti vettori sparsi.
            - Quando si vuole fare il ranking, questo viene fatto tramite una nozione di prossimità tra query e documenti dello spazio (vengono restituiti solo i documenti "vicini" alla query), e viene stabilita tramite la prossimità dei vettori (minore è la distanza tra due vettori, più sono simili). I risultati vengono poi rankati in base a questo valore.
            - Fanno uso della nozione di term frequency, dove i gli elementi di ogni vettore sono "pesati", ovvero gli viene assegnato un peso $tf_{t,d}$, ovvero la frequenza del termine t nel documento d.
        - Per stabilire la similarità si usa l'angolo tra i vettori passato per la funzione coseno, in quanto funzione monotono decrescente nell'intervallo $[0,180°]$. Viene usata quindi la similarità cosenica $\cos(\boldsymbol{q},\boldsymbol{d}) = \frac{\boldsymbol{q}\cdot\boldsymbol{d}}{|\boldsymbol{q}||\boldsymbol{d}|} = \frac{\boldsymbol{q}}{|\boldsymbol{q}|}\cdot\frac{\boldsymbol{d}}{|\boldsymbol{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$, dove q_i è il peso (tf) del termine i nella query, e d_i è la stessa cosa ma per il documento.
        - I termini comuni sono meno informativi di quelli rari, quindi si può utilizzare l'inverso della frequenza dei termini in un documento (inverse document frequency al posto della document frequency). Con df_t si fa riferimento al numero di documenti che contengono t (riguarda quindi tutti i documenti, e indica quanto sia comune quel termine: è quindi l'inverso dell'informatività di t). La IDF è definita come $idf_t = \log_{10}\left(\frac{N}{df_t}\right)$, dove N è il numero di documenti nella collezione (il logaritmo è usato per appiattire il rapporto). Idf non ha effetti se il ranking viene fatto su query di un solo termine (devono esserci almeno 2), e viene usato insieme al TF per avere una migliore stima della rilevanza di un documento. In particolare, il peso tf.idf è il prodotto del suo peso e del suo peso idf: $w_{t,d} = tf_{t,d} \times \log_{10} N/df_t$, il peso aumenta con il numero di occorrenze nel documento e con la rarità del termine nella collezione.
- Modelli probabilistici
    - La probabilità viene utilizzata per fornire il ranking dei documenti, e viene misurata a partire dai dati disponibili (ranking decrescente di probabilità di rilevanza), la qualità dei risultati sarà la migliore sulle basi dei dati forniti.
        - BM25 (Okapi), fa uso di frequenza dei termini, rarità dei termini (simile a idf) e la lunghezza dei documenti. Lo score di similarità è compreso tra 0 e 1, e la frequenza dei termini ad un certo punto non impatta più sul punteggio. Ha due parametri, ovvero k1 (controlla quanto velocemente la term-frequency si satura) e b (questo parametro controlla quanto pesa la normalizzazione della lunghezza del documento)
        - Modelli basati su linguaggi, dove un modello statistico assegna una probabilità a una sequenza di m parole P tramite una distribuzione. Ogni documento è associato ad un modello statistico e il ranking viene fatto sulla probabilità della query Q nei vari modelli dei documenti. Il modello può essere di tipo

Unigram (probabilità distribuita tra le parole di una lingua) o N-gram (usati dove la probabilità dipende sulle parole precedenti).

- Modelli combinati
    - Network inferenziali
    - Sistema di apprendimento a Rank

4.1 – Inverted index

Per ogni termine t, è importante immagazzinare una lista di tutti i documenti che lo contengono (ogni documento viene identificato da un id di documenti), e dato che il vettore non può essere di dimensione fissa viene utilizzata una posting list (ogni keyword ha determinati "posting" che sono i documenti in cui compare). L'inverted index viene costruito facendo passare i documenti da indicizzare per un tokenizzatore (che esegue anche operazioni di normalizzazione dei token), per poi moduli linguistici (per confrontare il token con il dizionario della lingua, tentando di ottenere ad esempio il plurale o il singolare), e una volta svolti questi passaggi le posting list vengono aggiornati dall'indicizzatore.

Il preprocessing di un testo consiste nella tokenizzazione (rilevazione delle parole), normalizzazione (rimozione di lettere puntate, trattini, rimozione delle maiuscole), lemmatizzazione (riduzione di forme variante a forma base), stemming (rilevazione di radici comuni tra parole), rilevazione delle stopword (ed eventuale rimozione delle parole più comuni, come the, or, a..., che vanno però a ridurre la precisione).

Per quanto riguarda i moduli linguistici, vengono utilizzati i dizionari specifici per quella lingua o per quel dominio.

Elaborare la rilevanza usando modelli non booleani può essere un'attività estremamente pesante, e ci sono strategie per controllare l'efficienza (spesso non si è interessati ai documenti meno rilevanti, e il modo in cui si accede ai documenti cambia molto le cose). Si sceglie di non calcolare lo score completo di tutti i documenti che non arriveranno in top k (a causa di un budget temporale estremamente risicato). La strategia TAAT assegna un punteggio a tutti i documenti, un termine di query per volta, mentre la DAAT valuta il punteggio di un documento per volta, di tutti i termini della query: ognuna delle due strategie implica in che modo l'indice deve venire strutturato.

Con safe ranking si fa riferimento ai metodi che garantiscono che i K documenti restituiti siano quelli che sono certamente i K migliori in termini di score (in un sistema non-safe su usano approssimazioni, come il metodo di index elimination e di champion list).

I motori di ricerca, quando l'utente cerca qualcosa, scorrono il loro indice, che viene popolato tramite dei crawler, ed esegue il ranking. Il motore di ricerca ha a disposizione il bisogno informativo dell'utente, la query dell'utente e l'intento della query.

Google non usa solo la frequenza dei termini per il ranking delle pagine, ma tiene conto anche di altri fattori (diversi pesi per titoli e sottotitoli, "freschezza" di una pagina..., ed è un meccanismo vizioso in cui pochi siti dominano i top rank). Dei 10 risultati prominenti, sullo schermo vengono automaticamente mostrati solo i primi 5 (nella SERP, search engine result page), gli altri si trovano

invece sotto. I primi risultati sono quelli più cliccati (il 40% dei click va al primo risultato, al secondo il 10%, e via dicendo).

4.2 – Web Information Retrieval

Il web crawling è il processo tramite è possibile ottenere, velocemente ed efficientemente, pagine dalla rete (inclusa la struttura di link che le interconnette). Partendo da "pagine seme", la frontiera del web esplorato viene estesa dal crawler, anche se non verrà mai esplorato al 100%: non tutto il web è visibile/indicizzabile, ed è anche estremamente vasto. Un crawler deve essere robusto (evitare trappole, spam, essere distribuito) e deve essere educato (il webserver può avere politiche che indicano ogni quanto possono venire visitati da un crawler), in modo esplicito (robots.txt specifica quali porzioni possono essere mappate) o implicito (evitare a prescindere di visitare troppe volte una pagina).

Una strategia possibile quando si esplorano pagine nella frontiera è la BFS, oppure utilizzare la DFS (nel primo caso si ha un'esplorazione ampia e non approfondita, nel secondo esplorazione non ampia ma approfondita). Cercare all'interno del web spinge a necessitare, insieme alla nozione di rilevanza, una nozione di popolarità del sito a cui la pagina appartiene: il sistema esegue ranking combinando il punteggio del contenuto (quanto è inerente) e il punteggio di popolarità (quanto è importante la pagina). Il contenuto può venire valutato tramite i meccanismi di ranking visti, mentre quello della popolarità può venire calcolato analizzando la struttura dei collegamenti ipertestuali eseguendo uno o più modelli di analisi dei collegamenti.

Il meccanismo "The good, the bad and the unknown" prevede the i nodi buoni non punteranno mai a nodi cattivi (se il nodo lo fa, allora il nodo è cattivo), e se il nodo viene puntato da un buono allora il nodo è buono. Un meccanismo di analisi di citazione può essere anch'esso un modo per valutare la popolarità di una certa pagina. L'idea del PageRank (inizialmente) assegna un punteggio tra 0 e 1 ad ogni nodo del grafo che rappresenta la rete, e questo punteggio dipende dalla struttura dei collegamenti del grafo.

Si consideri un navigatore A, che si muove in modo casuale da pagina a pagina: A è estremamente annoiata, e il suo browser le permette di accedere ad una pagina casuale. Ogni volta che una pagina web viene caricata, A sceglie se navigare in modo casuale usando un link sulla pagina oppure spostarsi su una pagina casuale e non legata al sito. A quindi sceglie un numero casuale r, e se r supera una certa soglia clicca sul pulsante "random", altrimenti clicca un link casuale sulla pagina: in questo modo, prima o poi, tutte le pagine verranno visitate. La probabilità che A stia visitando una pagina X è il PageRank di X.

Il **PageRank** corrisponde a trovare la probabilità di una random walk (una particolare catena di Markow, dove lo stato successivo dipende solo dallo stato attuale). Se, definita una rete con solo 3 pagine (A,B,C), il PageRank di C dipende sul PageRank di A e B: $PR(C) = \frac{PR(A)}{2} + \frac{PR(B)}{1}$. Si inizia assumendo che il peso di tutte le pagine sia lo stesso, e si procede ad iterare i calcoli per le altre pagine: dopo qualche iterazione, il PageRank converge a valori stabili (PR(C)=0.4, PR(A)=0.4, PR(B)=0.2).

Il PageRank è solo uno dei tanti valori che determinano il punteggio finale di una pagina su Google, che usa altri 200 segnali (caratteristiche del linguaggio, della query, del tempo, della posizione dell'utente...). Quando si esegue una certa query, al posto di restituire una lista ordinata di pagine, vengono trovati due insiemi: hub pages (buone liste di link su un argomento) e authority pages (pagine che sono presenti su buone hub per l'argomento). Questo meccanismo è ideale per gestire query che riguardano temi vasti, e una buona hub page punta a numerose pagine autoritative (e viceversa). La ricerca semantica è l'esecuzione di una ricerca su grafo su conoscenza al posto di ricerca standard su testi e documenti.

4.2.1 – Stabilire la rilevanza

I metodi per stabilire la rilevanza risalgono agli anni '60, e richiedono 3 elementi:

- Un benchmark (collezione di documenti);
- Un insieme di query da eseguire sul benchmark (su cui viene trasferito l'user need);
- Un sistema di valutazione umano che stabilisce se un risultato è (o meno) rilevante (il giudizio è relativo all'user need, e non alla query);

Il giudizio di rilevanza può venir fatto in molti modi, anche se il principale è dividere i documenti in rilevante e non rilevante. Tuttavia, se per ogni query consideriamo tutti quanti i documenti, l'operazione risulta essere estremamente costosa: quindi, si fa di solito l'analisi dei top-k documenti (**depth-k pooling**). Esistono delle collezioni di test già fatte (la più importante è la TREC GOV2), con query ad esse associate: per dimostrare che un sistema IR funziona meglio di un altro, vengono eseguite quelle query sui documenti nella collezione.

Un metodo utilizzato è quello del "Turco meccanico": al posto di usare una persona per classificare un documento, se ne usano numerose. Queste persone accederanno ad una certa piattaforma su cui sono presenti i documenti, e con il feedback degli utenti si rende possibile la costruzione del sistema di rilevanza. Questo funziona, ma è presente una grande quantità di rumore e la varianza nei risultati è elevata.

Un metodo meno sperimentale consiste nell'accertare l'efficacia di un sistema IR, ovvero tener conto di due parametri importanti:

- Precisione (frazione dei documenti restituiti che è rilevante al bisogno di informazione, R);
  - $Precisione = \frac{RRD}{RD}$
- Richiamo (frazione dei documenti rilevanti che sono stati inviati dal sistema, RD);
  - $Richiamo = \frac{RRD}{R}$

Documenti che sono sia rilevanti e ritrovati dal sistema (RRD) sono coloro che sono di maggior interesse per l'utente (non tutti saranno rilevanti e non tutti i rilevanti saranno stati estratti dal sistema).

Un sistema perfetto ha Precisione e Richiamo a 1 e per molti sistemi si accetta di avere una precisione entro le prime k risposte (**Precision@K**), quindi stabilire una soglia di ranking a k e calcolare la percentuale delle risposte rilevanti fornite nella top-k (e ignorare tutte le altre). Allo stesso modo viene definito **Recall@K**.

Altra misura è la Precisione Media (Average Precision, AP), che è una misura aggregata dei risultati rankati: al posto di impostare un K arbitrario, la valutazione della precisione si ferma solo quando tutti i documenti rilevanti sono stati trovati (ci si ferma quando la recall arriva a 1), e si calcolano le Precision@K solo per quelle k dove viene restituito un risultato rilevante: la precisione media viene calcolata da questi valori. Dopo aver calcolato la precisione media di ogni query nel test, è possibile computare la Mean Average Precision (MAP, "media precisione media"), la media della precisione tra tutte le query. Se un documento rilevante non viene mai prelevato, la precisione per la rilevanza del documento è 0, e la MAP è una macro-media (ogni query pesa uguale), dà per scontato che l'utente sia interessato a trovare molti documenti rilevanti (non il più rilevante) e richiede molti giudizi di rilevanza in collezioni di testi.

Finora è stata data per assodata l'esistenza di una nozione binaria di rilevanza (e così è normalmente), anche se teoricamente un documento può essere parzialmente rilevante.

4.2.2 – Metodi avanzati

In molte raccolte lo stesso concetto potrebbe venire espresso usando termini diversi: questo fenomeno è chiamato sinonimia. Un'idea può essere quella di "espandere" la query, arricchendola di parole chiave sinonime (questi saranno in or tra di loro), o ancora meglio aumentarla con termini ad essa legate. Sono state studiate numerose tecniche automatiche, il cui scopo è quello di migliorare l'efficacia del sistema aggiungendo termini legati alla query, e semi-automatiche, in cui l'input dell'utente consente di selezionare i termini migliori.

La tecnica di query suggestion consente di presentare all'utente query alternative, spesso usate da altri utenti nello stesso ambito di interesse. I termini vengono riconosciuti come "correlati" alla ricerca tramite vocabolari (non troppo precisi, in quanto non viene tenuto conto del contesto) oppure raccolte di testi, all'interno dei quali possono esserci termini frequentemente co-occorrenti e che compaiono all'interno di documenti rilevanti o restituiti all'utente.

Esistono diversi modi di misurare la co-occorrenza e sono basate su documenti o su sezioni di documenti (frasi, paragrafi):

- Coefficiente di Dice, data una parola la correlazione con un'altra è determinata da $\frac{T}{S}$, dove T è il numero di volte in cui appaiono insieme e S quante volte appaiono singolarmente. Questa misurazione viene usata dagli anni 60, ed è definita formalmente come $2n_{ab}/(n_a + n_b)$ dove a e b sono termini, n_a sono il numero dei documenti che contengono a, n_b idem ma che contengono b e n_{ab} sono i documenti che contengono entrambe.
- Mutua informazione, date due parole a e b è definita come $\log\frac{P(a,b)}{P(a)P(b)}$, dove P(a) è la probabilità che a sia in un documento (idem per P(b) e P(a, b)). Il problema del metodo è che tende a favorire termini poco frequenti (a parità di co-occorrenza, un termine che compare molte volte avrà un minore valore di mutua informazione).
- Mutua informazione attesa, risolve il problema della mutua informazione $P(a,b) * \log\frac{P(a,b)}{P(a)P(b)}$.
- χ^2 di Pearson, che misura il numero di co-occorrenze con il numero atteso di co-occorrenze se le due fossero indipendenti $\frac{\left(n_{ab} - N*\frac{n_a}{N}*\frac{n_b}{N}\right)^2}{N*\frac{n_a}{N}*\frac{n_b}{N}}$, dove le aree evidenziate sono il numero atteso di co-occorrenze.

Usando questi metodi, si può ricavare una lista delle parole più correlabili, di cui si può restituire solo il top-k, e si nota come il Chi-Quadro di Pearson faccia emergere parole bizzarre.

L'espansione delle query può venire eseguita tenendo in considerazione del feedback dell'utente per stabilire la rilevanza, con l'idea che è basata sull'esistenza di sessioni di ricerca all'interno della quale vengono eseguite le query e con l'utente che indica quali sono i più rilevanti, in modo da migliorare la precisione del sistema. Il sistema può venire automatizzato (Pseudo relevance feedback), in modo che all'utente vengano direttamente forniti i risultati della query espansa senza feedback dell'utente.

Se si volessero considerare, combinandole, la frequenza del termine nel documento/titolo, la lunghezza del documento, la popolarità del documento, ..., come se fossero delle feature per stimare la rilevanza di un documento, è necessario un modello di apprendimento per stabilire i pesi da dare ad ogni feature: l'idea è quella di ricorrere al machine learning per costruire un classificatore che divide i documenti in rilevanti e non rilevanti.

Questa soluzione è stata resa possibile dalla grande quantità di dati contenenti log di query su cui eseguire analisi: usare giudizi di rilevanza di una collezione di test per allenare il classificatore è detto offline learning, mentre usare il feedback di rilevanza per migliorare la precisione del classificatore è un esempio di online learning.

5 – Analisi dei dati

Con il termine Data Analytics si intende generalmente l'analisi dei dati. I dati grezzi devono venir compresi (informazione), avere un contesto (conoscenza, sistemi di apprendimento automatico) e venire "applicati" (saggezza, deep learning): Data Analytics si pone tra informazione e conoscenza, insieme a data warehousing e sistemi per supporto alle decisioni.

L'obiettivo della Data Analysis è quello di estrarre informazioni basilari da insiemi di dati: queste informazioni possono essere informazioni sintetizzate (media di valori numerici) oppure associative (presi due insiemi, si cerca di determinare quale sia la relazione tra i due).

Una **popolazione** è un insieme di oggetti verso cui siamo interessati.

Una **variabile** è il nome di un valore di un **record** (tupla di valori che caratterizza un elemento della popolazione), e ha significato comune per tutti gli elementi della popolazione. Possono essere quantitative (numeriche, se è possibile applicarci operazioni aritmetiche) o qualitative (categoriali). Le prime possono essere discrete (se è possibile contarne i valori) o continue, le altre possono essere ordinali (se esiste un ordine naturale tra i possibili valori) o nominali.

Un **dataset** è un insieme di record.

5.1 – Statistica descrittiva

La statistica descrittiva forniscono indicatori che consentono di identificare con un solo valore le proprietà statistiche di una popolazione rispetto ad una sola variabile (media aritmetica, moda mediana, varianza, deviazione standard, ...) o a più variabili (covarianza, correlazione).

- Indicatori a singola variabile
    - Indicatori di centralità, consentono di riassumere un set di valori in un unico valore numerico.

        - La media aritmetica è definita come $\sum_{i=1}^{n} X_i / n$. Nel caso in cui alcuni record non abbiano tutti informazioni nella variabile per cui si sta facendo la media, è possibile sostituire quel dato mancante con la media stessa. È sensibile alle anomalie.
        - La mediana è il valore collocato nella posizione centrale della lista dei valori ordinata. È un indicatore robusto, in quanto variazioni molto grosse o molto piccole non impattano il valore mediano.
        - La moda, dato un insieme di variabili, è definita come il valore con la maggior frequenza nell'insieme delle osservazioni. A differenza delle precedenti, la moda è utilizzabile anche per dati categorici, utilizzabile in sistemi di voto. E' robusta come la mediana, ed ha senso usarla quando non esiste ordine lineare sui valori possibili;
    - Indicatori di variazione
        - Deviazione quadratica, definita come $\sum_{i=1}^{n}(x_i - \bar{x})^2$, dove $\bar{x}$ rappresenta la media. È influenzata dal numero di osservazioni (più sono, maggiore sarà la deviazione).
        - Varianza, definita come $s^2 = \frac{1}{n}\, dev$. Normalizza la deviazione quadratica con il numero di osservazioni.
        - Deviazione standard, definita come $s = \sqrt{s^2}$. Consente di esprimere la deviazione nelle stesse unità dei dati.
    - Altri: minimo, massimo, range (differenza tra massimo e minimo).
- Indicatori a variabili multiple, usati per descrivere quale sia la relazione tra due variabili.
    - Covarianza, definita come $cov(X,Y) = \frac{\sum_{i=1}^{n}(X-\bar{X})\times(Y_i-\bar{Y})}{n}$, e quantifica la forza della relazione tra X e Y. È la media dei prodotti delle deviazioni, e si basa sull'idea che la somma dei prodotti di due deviazioni è alta se, ogni volta che X devia dalla sua media, lo fa anche Y. Se la covarianza è positiva, X e Y sono direttamente proporzionali, diversamente sono inversamente proporzionali. La covarianza è limitata dall'unità di misura, con valori molto grandi che portano ad un alta covarianza (anche se X e Y non sono legati) e viceversa con valori piccoli.
    - Correlazione, definita come $corr(X,Y) = \frac{cov(X,Y)}{stdev(X)\times stdev(Y)}$, che risolve il problema della covarianza inserendo il risultato tra -1 e 1. Se vale intorno a -1 sono inversamente proporzionali, se vale circa 1 sono direttamente proporzionali e se vale quasi 0 sono indipendenti (questo solo per relazioni lineari).
    - È utile visualizzare i valori delle variabili su un piano XY: mentre misure di associazione dicono se esiste o meno un'associazione, modelli a regressione stimano la funzione che lega le due variabili.

5.2 – Data Warehouse

Per capire cosa sia un DW (Data Warehouse) è necessario riprendere la definizione di database: un sistema che supporta le operazioni di applicativi, creato per essere aggiornabile in tempo reale e tramite il quale si interagisce con query complesse per eseguire analisi dati (con join costosi, e potrebbe non avere dati storici) che richiedono conoscenza dello schema dei dati. In un contesto di business, per guidare decisioni, i dati dovrebbero essere una grande sorgente di informazioni senza bisogno di una conoscenza profonda dello schema da parte del decision maker, con la possibilità di

integrare dati da più sorgenti: qui entrano in gioco i DW. Integrare database operazionali è estremamente complesso, dal momento che hanno diverse definizioni di dati e contenuti.

Un **Data Warehouse** è una collezione di dati integrati (provenienti da diverse sorgenti), non volatili (accumula dati per molto tempo, la modifica e la rimozione di dati non sono consentiti), varianti nel tempo (la DW tiene traccia di come i dati sono evoluti nel tempo) e orientata al soggetto (orientata rispetto ad una certa analisi) in grado di supportare decisioni amministrative.

Mentre la progettazione dei database viene fatta seguendo 4 fasi (specifica dei requisiti, progettazione concettuale, progettazione logica, progettazione fisica) che portano ad un database relazionale, per i DW il modello relazionale non è appropriato (necessaria una buona performance in query complesse) in quanto serve una tecnica di progettazione che:

- Possa supportare l'utente finale nella creazione di query;
- Sia orientato verso la comprensione del risultato senza conoscenza pregressa dello schema;
- Sia orientato alle performance, accettando la ridondanza se porta a un boost di performance;

Il modello impiegato prende il nome di modellazione multidimensionale (MD), che vede i dati come un insieme di fatti legati a dimensioni, dove:

- Un fatto rappresenta l'obiettivo dell'analisi;
- Una misura quantifica fatti;
- Una dimensione viene usata per osservare misure da diverse prospettive, come ad esempio il tempo. Le dimensioni includono attributi che formano gerarchie (anno/mese/giorno), che consentono di selezionare diversi livelli di dettagli;

L'aggregazione di misure si verifica ogni volta che una gerarchia è attraversata (muoversi da mese ad anno aggregherà i valori per ogni anno). Al livello logico il modello multidimensionale viene rappresentato da tabelle relazionali organizzate in schemi a stella e a fiocco di neve, che pongono la tabella del fatto al centro, collegata alle diverse tabelle delle dimensioni (gli schemi a stella usano una tabella denormalizzata unica per ogni dimensione anche in presenza di gerarchie, mentre gli altri usano diverse tabelle normalizzate in presenza di gerarchie dimensionali).

Il processo per popolare un DW è chiamato ETL:

1. **E**xtract: i dati da diverse sorgenti vengono estratti;
    a. È statica quando il DW deve venire popolato per la prima volta;
    b. È incrementale per aggiornare il DW, catturando solo le modifiche;
2. **T**ransform: i dati vengono puliti e trasformati per rientrare nel modello del DW (normalizzazione, standardizzazione e correzione), complicato quando sono presenti testi liberi e diversi formati per lo stesso dato;
    a. La pulizia serve per rimuovere dati duplicati, inconsistenze, valori mancanti o errati, uso erroneo di un campo...;
3. **L**oad: i dati trasformati vengono caricati;

Un ETL è molto complesso, ed è l'80% del costo di un DW. Mentre un database tradizionale è un OLTP (online transaction processing), un data warehouse è un OLAP (off line analytical processing):

- OLTP è concentrato sulle transazioni, OLAP su query analitiche, le quali non sono facili da normalizzare;
- Database OLAP sono in grado di gestire un carico pesante di query;
- Le tecniche di indexing OLTP non sono efficaci in OLAP;
- Le query OLAP generalmente includono aggregazione;

5.2.3 – Architettura delle DW

Una possibile architettura per un DW è quella a tier, dove esiste una divisione in livelli:

- Data source;
- Livello di elaborazione, esegue estrazione, trasformazione (pulizia/integrazione/aggregazione) e caricamento nel data warehouse dei dati (caricamento iniziale e ricorrente);
- Data warehouse, un DW e diversi data marts (piccoli DW che misurano un soggetto ben preciso) con numerosi metadati (sia quelli che descrivono la semantica dei dati sia quelli che descrivono la struttura del dato e il loro stoccaggio);
- OLAP, che consente l'analisi dei dati;
- Frontend, che fa uso di diversi tool per analisi e visualizzazione (tool OLAP per l'esplorazione, tool di reporting per la creazione di report, tool statistici e tool di data mining);

Per tier si fa riferimento a un modo di astrarre un problema, dividendo i compiti per scopo (1 scopo = 1 tier) e dando per scontata l'esistenza di un qualche supporto che esegua quelle operazioni.

Il modello multidimensionale vede i dati in uno spazio n-dimensionale, composta da dimensioni e fatti (dove le dimensioni sono le prospettive usate per analizzare i dati). Il livello di dettaglio delle misure per ogni dimensione del cubo è detto granularità del dato (minore la granularità, maggiore la precisione, maggiore i dati da raccogliere, maggiore il costo), mentre istanze di una dimensione sono detti membri (Cibo e Bevande sono membri di Prodotto con granularità Categoria). Una gerarchia consente di vedere dati a diverse granularità, dove il livello basso è detto figlio e quello superiore prende l nome di genitore, e a struttura gerarchica di una dimensione prende nome di schema di dimensione.

Un cubo dati potrebbe essere estremamente denso oppure sparso (in base alle scelte delle dimensioni). Le misure possono essere additive (sommate tra tutte le dimensioni), semi-additive (ha senso sommarle solo su alcune dimensioni), non additive (non possono essere sommate tra alcuna dimensione). Altre classificazioni per le misure possono essere:

- Distributive (definite tramite funzioni di aggregazioni computabili in modo distribuito);
- Algebriche (misure definite da una funzione di aggregazione che può essere definita come una funzione scalare di funzioni distributive);
- Olistiche;

5.2.4 – Query

Le operazioni eseguibili su un DW sono:

- Roll-up
    - Si passa da un livello inferiore a un livello superiore della gerarchia di una dimensione;
- Drill-down

    - Si passa da un livello superiore a un livello inferiore della gerarchia di una dimensione;
- Sort
    - Consente di ordinare la visualizzazione del cubo in base ad una certa dimensione;
- Pivot
    - Consente di ruotare il cubo, cambiando l'ordine delle dimensioni per una migliore visualizzazione;
- Slice
    - Esegue un "taglio" dei dati in base ad una dimensione;
- Dice
    - Esegue una selezione di un sottoinsieme del cubo n-dimensionale;

5.3 – Data Mining

Il data mining consiste nel processo di elaborazione che consente di scoprire pattern di regolarità in grandi dataset usando metodi che riguardano 3 campi: Machine Learning, statistica e basi di dati. L'obiettivo è estrarre un pattern (e conoscenza) da una grande quantità di dati, che possa venire usato come indicatore.

Un **pattern** è una regolarità all'interno di dati che si ripetono per tutte le osservazioni in modo predicibile.

Una volta che un pattern viene appreso, è possibile studiarlo per capirne la logica (descrivendola) e prevedere futuri fenomeni. Per poter trovare i pattern, è necessario esaminare grandi quantità di dati in forma omogenea, prima di procedere con il data mining:

1. Definire il problema (cosa vogliamo ottenere);
2. Identificare i dati necessari (di che dati abbiamo bisogno);
3. Preparazione e pre-elaborazione, selezionare e pulire i dati;
4. Modellare l'ipotesi, selezionando algoritmi di Machine Learning e regolarne i parametri;
5. Allenare il modello e testare l'efficacia;
6. Verificare il modello finale e rilasciarlo;

Il ML è necessario in quanto i pattern su quantità così grandi su dati sono difficili da rilevare per un umano, e in particolare se esistono relazioni non lineari tra variabili. I pattern e la conoscenza estratte sono solo approssimazioni: sono, probabilmente (il modello del mondo reale dovrebbe essere corretto), approssimazioni (il modello dovrebbe avere un basso errore di generalizzazione) corrette (teoria PAC, probably approximately correct). Si prende, quindi, un segnale da una collezione di dati (per quanto grande comunque limitata) e si proietta sul generale.

Esempio: un'osservazione (informaizoni su una singola casa, ad esempio);
Variabili di input: variabili date in input per ogni esempio (i metri quadri, ad esempio);
Variabile di output: variabile che si vuole prevedere in base agli input;
Ipotesi: il modello, la funzione che viene appresa dall'algoritmo di ML;
Etichetta: in apprendimento supervisionato, la collezione di dati è "etichettata", ovvero per ogni esempio nel training set viene indicato il valore della variabile di output;
Predizione: il valore della variabile di output indovinato dalla funzione;

La notazione standard indica X come la matrice dati di input, y come vettore della variabile di output, m come il numero di esempi e n come il numero di variabili negli esempi.

Nelle tecniche di apprendimento supervisionato, il modello viene addestrato con un set di esempi, e le predizioni vengono comparate con il risultato etichettato: si tratta quindi di un problema di regressione nel caso in cui sia un valore numerico l'output mentre si tratta di classificazione se si deve distinguere l'input in più categorie (binario, ternario, ...).

Nell'apprendimento non supervisionato, non è presente un valore per la variabile d'output (oppure non esiste proprio la variabile): non sempre esiste un goal, e si vogliono solo cercare pattern all'interno di dati. Le tecniche principali sono:

- Clustering: dividere esempi in diversi cluster (gruppi);
- Regole associative: scoprire relazioni tra variabili;
- Rilevazione anomalie: trovare casi estremi in un insieme di dati;
- Modelli generativi: dati molti esempi viene creato un nuovo esempio con caratteristiche simili;
- Estrazione di feature: le feature degli esempi vengono ridotte in numero usando nuove caratteristiche generate dall'algoritmo;

Mentre in apprendimento supervisionato è facile capire quanto sia preciso il sistema, in apprendimento non supervisionato è molto più complicato.

5.3.1 – Pre-elaborazione dei dati

Lo step di pre-elaborazione è qualcosa di comune a tutti i progetti di data mining, e consente di risolvere problemi legati a dati del mondo reale (incompletezza, rumore, inconsistenza). Le fasi principali sono:

- Pulizia: inserimento informazioni mancanti, riduzione del rumore, rimozione casi estremi, risoluzione inconsistenze;
    - I dati possono essere incompleti per numerosi motivi (problemi software, differenze da quando i dati sono stati acquisiti e analizzati, ...) e allo stesso modo sono rumorosi (raccolta dati difettosa, errori di trasmissione, ...), inconsistenti (diverse sorgenti, ...).
    - Non sempre le informazioni sono disponibili, a causa di malfunzionamenti, dimenticanze o incomprensioni, e queste informazioni possono essere dedotte (o si può semplicemente ignorare la tupla):
        - Questo può essere fatto automaticamente (costante globale, usare un valore medio, più probabile, ...), oppure manualmente.
    - In caso di rumore, si può ricorrere al binning (appiattire un valore in un set ordinato considerando il suo vicinato), regressione e clustering;
- Integrazione: integrazione di dati provenienti da altre sorgenti;
    - Le differenze semantiche e struttura di dati sono grossi ostacoli, e il problema di incrociare schema e oggetti da diverse sorgenti prende il nome di "Entity Identification Problem". Questo può essere superato usando i metadati (nome degli attributi, tipo di dati, range dei valori) e facendo valutazioni su di essi.
    - Si possono raggiungere accuratezze elevate con sistemi automatici, ma è comunque necessario verificare manualmente.

- Trasformazione: codifica di testo e categorie su vettori di numeri reali e normalizzazione (per poter far funzionare gli algoritmi di ML);
    - E' necessaria in quanto non tutte le variabili potrebbero essere sumeriche, e numero molto grossi non sono ideali nel machine learning (algoritmi di ottimizzazione convergono velocemente con valori normalizzati);
    - Un testo può essere vettorizzato nello stesso modo precedentemente visto (bag of words), o in alternativa si può categorizzare (one hot encoding);
    - La normalizzazione viene fatta con lo scaling lineare $scale(value) = \frac{value - \min(values)}{\max(values) - \min(values)}$;

X – Seminari

X.1 – Temporal Information Retrieval

All'interno di testi, apparentemente non strutturati, sono in realtà strutturati secondo espressioni temporali, quindi potrebbe essere possibile estrarre la un documento la sua dimensione testuale, e questa informazione (presente anche nelle query) può venire impiegata per migliorare l'efficacia dell'IR. È necessario determinare quali siano le proprietà temporali dei documenti e delle query, che verranno trattate come una normale keyword dal sistema IR. Il tempo è una dimensione ubiqua, e può essere interno (una data in un documento o query) oppure esterno (data di pubblicazione oppure data di esecuzione della query). Query e documenti avranno un temporal scope su cui fare ricerche, che consente di migliorare la precisione del sistema IR.

Il problema risulta estrarre informazioni legate a espressioni temporali (in particolare quando ce n'è più di una), e riuscire a elaborare correttamente il tutto a livello di informazioni, che necessitano di venire normalizzate. Al fine di gestire il tutto, è necessario modellare la similarità temporale, andando a correlare eventi temporali (sia in query che documenti) costruire una distanza tra espressioni temporali/intervalli.

X.2 – Analisi del parlato per rilevazione di malattie cognitive

L'aspettativa di vita è aumentata a livello globale, e questo ha portato ad alcuni problemi legati all'età. La prevalenza di demenza tra anziani è in crescita, ed è una grande sfida: il WHO stima che entro il 2050 in 150 milioni soffriranno di demenza. Il problema è aggravato dal fatto che nessuna terapia riesce a risolvere il problema, solo a rallentarlo: prima la malattia viene rilevata, più efficaci sono le contromisure. Un sistema automatico e standardizzato in grado di rilevare i segnali dell'Alzheimer è essenziale per ottenere risposte rapide e sistemi di screening.

La demenza impatta diversi domini (memoria, attenzione, ...) ed è stato dimostrato come i tool di screening più comuni siano inadeguati per una diagnosi prematura. Studi recenti individuano nel linguaggio un sistema promettente per rilevare segni di declino cognitivo, esaminando il linguaggio sotto diversi aspetti. Molti dei metodi proposti, tuttavia, riguardano test manuali e molto time-consuming.