

DATA MINING E TEXT ANALYTICS

2^k dove k è numero di bit

Computer a 2 bit potrà avere 4 lettere, rappresentazione

LLM, cioè reti neurali

INTRODUZIONE

- **Rumore** (noise) —> qualcosa che non ci fa leggere bene i dati, disturba la lettura del dato e quindi anche l'interpretazione.
- **Bias** (altro tipo di rumore) —> pregiudizi, per esempio di genere. Preconcetti che possono essere riprodotti sia nei dati sia in chi legge.

Differenza—> noise in generale è qualcosa che disturba il segnale, bias preconcetto che viene riflesso all'interno dei dati

Rumore sale e pepe (pixel bianchi e neri sull'immagine), attraverso il **filtro mediano** è possibile cancellare il rumore. Rumore è un qualcosa che viene aggiunto.

Perchè utilizzare il data mining?

Il data mining è un processo utilizzato per scoprire informazioni preziose all'interno di grandi quantità di dati. Per comprendere meglio questo concetto, immagina di scavare un buco mentre cerchi qualcosa di prezioso. (Scoprire conoscenza)

Il Processo di Ricerca dell'Informazione

Durante il processo di scavo, dovrà attraversare diversi strati di materiale: terra di scavo, rocce, residui di pietra e altri elementi che potrebbero non essere esattamente ciò che stai cercando. Allo stesso modo, nel data mining lavori con grandi volumi di dati grezzi per trovare le informazioni che ti servono.

Strumenti e Tecniche Necessari

Per raggiungere il tuo obiettivo - l'**INFORMAZIONE** - hai bisogno di alcuni strumenti e tecniche specifici che ti guidino nel processo di ricerca e analisi.

I Pattern: I Mattoni dell'Informazione

Prima di estrarre l'informazione finale, è necessario mettere insieme alcuni piccoli pezzi o componenti che, una volta combinati, favoriscono la composizione dell'informazione stessa. Questi piccoli pezzi sono chiamati **PATTERN** (modelli).

Definizione di Pattern

I pattern sono un insieme di elementi presenti nei dati che esprimono:

- Sequenze: ordini specifici di eventi o dati
- Motivi: ricorrenze o schemi ripetitivi
- Codici: rappresentazioni strutturate di informazioni
- E molti altri tipi di strutture informative

Pattern in stock price

Analisi del Prezzo delle Azioni

L'analisi del prezzo delle azioni è un argomento cruciale. Le fluttuazioni nei prezzi possono rivelare eventi di interesse per il mercato.

Esempio di Serie Storica:

Dove:

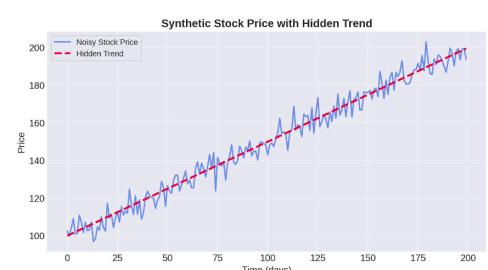
- P_t è il tempo dell'azione al tempo
- ϵ_t è un fattore additivo (rumore o variabilità casuale)

$$P_t = 100 + 0.5t + \epsilon_t$$

Trends and hidden patterns

Punto medio è una linea (rossa)

Trend cercando di eliminare la diminuzione e aumento del prezzo



Esempi di Data Mining

- Identificare correlazioni nei dati finanziari e di business intelligence
- Consentire agli ospedali di rilevare trend e anomalie nelle cartelle cliniche dei pazienti
- Migliorare il posizionamento e il ranking negli engine di ricerca e negli spazi pubblicitari
- Permettere ad agenzie ambientali e di sanità pubblica di individuare schemi e irregolarità nei loro dataset
- Monitorare i consumi energetici degli elettrodomestici di casa
- Analizzare pattern in bioinformatica e in dati farmaceutici
- Scoprire trend in blog, Twitter e altre piattaforme social

Definizione formale di Data Mining

Il data mining, comunemente noto anche come **Knowledge Discovery from Data (KDD)**, è l'estrazione automatizzata o facilitata di pattern che rappresentano conoscenza implicitamente memorizzata o catturata in grandi database, data warehouse, sul Web, in altri vasti repository di informazioni o in flussi di dati.

Definizione informale di Data Mining

Il Data Mining è il processo mediante il quale un programma informatico esplora grandi quantità di dati alla ricerca di pattern o relazioni nascoste.

- Individuazione di combinazioni di sintomi che fungono da indicatori affidabili per una determinata malattia
- Scoperta di quali prodotti i clienti tendono a comprare insieme (analisi del "market basket")
- Raggruppamento di dati che mostrano particolari correlazioni tra variabili
- Qualsiasi scenario in cui serva estrarre insight da enormi moli di informazioni in modo automatizzato

Che cosa rivelano i pattern?

I pattern consentono di individuare regolarità, tendenze e correlazioni nei dati, mettendo in luce relazioni non ovvie tra variabili.

1. Definizione di pattern

Un *pattern* è qualsiasi elemento o sequenza che si ripete con una forma o organizzazione costante all'interno del dataset.

2. Correlazioni mostrate dai pattern

1. Una sequenza di segnali che si manifesta periodicamente
2. Elementi che seguono una progressione lineare

4. Tipologia di dati

I pattern possono emergere sia da dati strutturati (tabelle, database relazionali) sia da dati non strutturati (testi, immagini, flussi di log).

Dati strutturati

Dati con un elevato grado di organizzazione (in modo simile a un foglio di calcolo), dove ogni elemento è descritto da campi e schemi ben definiti.

Dati semistrutturati

Dati con un certo livello di organizzazione (per esempio moduli XML o JSON), che includono tag o marcatori ma non rispettano uno schema relazionale rigido.

Dati non strutturati

Dati privi di un'organizzazione predefinita (come un file di testo libero), senza uno schema o campi stabiliti a priori.

Structured Data	Semi-structured Data	Unstructured Data
Excel spreadsheets; Comma-separated value file; Relational Database Tables.	HTML files; Json files; XML files.	Images (.jpeg or .png files); videos (.mp4); sound files (.wav, .mp3 files); PDF files; Word Files; Plain text files.
Around 20% of worldwide data is structured.	Characterised by hierarchical structure.	Most of data that is created today is unstructured. (Tweets, Facebook posts, social media comments).

Databases

Una database è una raccolta organizzata di dati strutturati, tipicamente memorizzati elettronicamente in un sistema informatico. Ogni elemento di informazione è archiviato secondo uno schema definito che ne facilita l'accesso, la gestione e la modifica.

Il database è normalmente gestito da un Database Management System (DBMS), un software che offre strumenti per:

- Definire e modificare la struttura dei dati
- Inserire, aggiornare ed eliminare record
- Eseguire query per recuperare informazioni
- Garantire integrità, sicurezza e concorrenza nell'accesso ai dati

Data Warehouse: Definizione e caratteristiche

Un data warehouse è un sistema di gestione dei dati progettato per supportare attività di Business Intelligence (BI) e soprattutto analisi.

- È pensato per eseguire query e analisi, spesso su grandi volumi di dati storici.
- I dati contenuti in un data warehouse provengono da una molteplicità di fonti (log applicativi, database transazionali, CRM, ecc.).
- Col tempo costruisce un archivio storico che diventa prezioso per data scientist e analisti aziendali, fungendo da "**single source of truth**" (unica fonte di verità).
- Grazie alla centralizzazione e alla consolidazione dei dati, un data warehouse permette di ottenere viste integrate e coerenti dell'intera organizzazione, facilitando decisioni basate su informazioni complete e affidabili.

Enterprise Data Warehouse

Raccoglie tutte le informazioni relative ai soggetti di interesse dell'intera organizzazione.

È progettato per supportare l'integrazione dei dati a livello aziendale (cross-funzionale) e può contenere da centinaia di gigabyte a terabyte (o oltre) di dati storici.

Data Mart

Contiene un sottoinsieme di dati aziendali di valore per un gruppo specifico di utenti (ad esempio un dipartimento).

Il suo ambito è limitato a determinati soggetti. Per esempio:

- Un *marketing data mart* potrebbe concentrarsi su cliente, prodotto, canali di marketing e vendite.
- Un *risk control data mart* potrebbe focalizzarsi su credito cliente, rischi e frodi.

Data Lake

In alcune aziende esistono enormi quantità di fonti dati complesse con una grande varietà di tipologie, formati e qualità dei dati, come:

- Dati aziendali
- Comunicazioni tra clienti e organizzazione
- Normative e regolamentazioni
- Analisi di mercato

Può essere difficile progettare un Data Warehouse dove i dati vengono integrati, strutturati e caricati secondo criteri specifici, specialmente quando si ha a che fare con questa varietà di fonti eterogenee.

Un **data lake** è un repository unico che contiene tutti i dati aziendali nel loro formato naturale, includendo, utile per i disaster recovery:

- Dati relazionali (database strutturati)
- Dati semistrutturati (file XML, JSON)
- Dati non strutturati (documenti, testi)
- Dati binari (audio, immagini, video)

Un data lake spesso conserva sia: (**key:backup**)

- Copie raw (dati grezzi originali)
- Dati trasformati (elaborati attraverso processi di analisi)

Questi vengono archiviati in repository cloud o distribuiti, permettendo una gestione scalabile e flessibile delle informazioni aziendali.

Database, Data Warehouse e Data Lake

Database (DB)

I **database** sono progettati per l'elaborazione rapida e transazionale di dati strutturati.

Data Warehouse (DW)

I data warehouse sono ottimizzati per le query analitiche su dati strutturati, con un approccio top-down e orientato agli obiettivi.

Data Lake (DL)

I data lake memorizzano dati grezzi, non strutturati o semistrutturati per analisi su larga scala.

Scenari reali con Database, Data Warehouse e Data Lake

Elaborazione transazionale con Database

Un sito di e-commerce utilizza un database relazionale (ad esempio MySQL) per memorizzare e gestire ordini cliente, giacenze di magazzino e account utente. Quando un cliente effettua un ordine, il database elabora la transazione in tempo reale, aggiornando le quantità di prodotto e registrando i dettagli dell'ordine.

Query analitiche con Data Warehouse

Un'azienda retail impiega Amazon Redshift come data warehouse per analizzare le tendenze di vendita degli ultimi cinque anni. Il data warehouse aggrega i dati di vendita provenienti da diversi negozi, consentendo all'azienda di eseguire query complesse per comprendere il comportamento dei clienti, i trend stagionali e la redditività.

Gestione di dati grezzi e non strutturati con Data Lake

Un servizio di video streaming sfrutta Amazon S3 come data lake per immagazzinare grandi volumi di file video grezzi, log di interazione degli utenti e metadati. I data scientist accedono a questi dati "raw" per eseguire algoritmi di machine learning che raccomandano contenuti agli utenti in base alla loro cronologia di visione e alle loro preferenze.

Data and information technology evolution

- "La necessità, che è la madre dell'invenzione."
– Platone
- Enorme crescita del volume di dati disponibili
- Informatizzazione della società

- Rapido sviluppo di potenti strumenti di raccolta e di archiviazione dei dati

- Alcuni dati sulla crescita dei dati

Ogni giorno vengono caricati su YouTube 720.000 ore di video. Ogni giorno vengono condivise su Instagram 1,3 miliardi di immagini.

Esempi di dati prodotti dalle aziende

Transazioni di vendita, record di scambi azionari, descrizioni di prodotti, promozioni commerciali, profili e performance aziendali, e feedback dei clienti sono solo alcuni esempi dei dati generati dalle imprese.

I dati delle vendite al dettaglio vengono attualmente analizzati su più variabili per identificare pattern e tendenze.

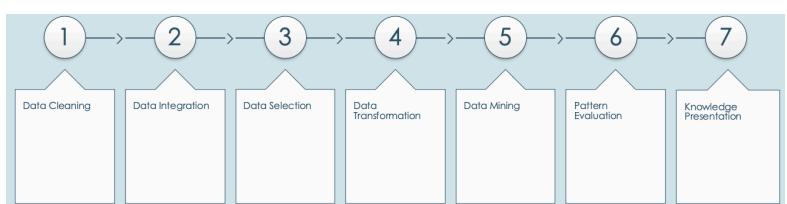
Wal-Mart gestisce ogni settimana centinaia di milioni di transazioni in migliaia di filiali sparse nel mondo.

Studi scientifici dimostrano che il meteo può influenzare le vendite al dettaglio.

Le cinque categorie di prodotti più sensibili alle condizioni atmosferiche coprono un'ampia gamma di tipologie, con gli alimenti salutistici in prima linea per variabilità legata al clima.

Statistiche e Tendenze

- Statistiche e trend possono aiutare a scoprire alcune correlazioni tra i dati.
- Se alcuni pattern sono facili da individuare, altri non sono altrettanto semplici da estrarre.
- Il data mining entra in gioco per scoprire conoscenza a partire dai dati disponibili.
- Il processo di Knowledge Discovery Process è composto da diversi passi. (Passo 0 → raw data)



KDP: Passi principali ed esempi

1. Data Cleaning (Pulizia dei dati)

- Rimozione di rumore e dati inconsistenti.
- Obiettivo: rendere i dati più affidabili eliminando errori, duplicati e valori anomali.

2. Data Integration (Integrazione dei dati)

- Unione di più fonti di dati, integrazione sorgenti.
- Obiettivo: combinare informazioni provenienti da database diversi per avere un dataset completo e coerente.

3. Data Transformation (Trasformazione dei dati)

- Trasformazione e consolidamento dei dati in forme adatte all'analisi.
- Include operazioni di sintesi e aggregazione (ad esempio: calcolo delle medie, somma di valori, raggruppamenti).

4. Data Mining (Estrazione dei dati)

- Applicazione di metodi intelligenti per estrarre pattern significativi o costruire modelli dai dati.
- Esempi: algoritmi di classificazione, clustering, regole di associazione.

5. Pattern/Model Evaluation (Valutazione dei Pattern o Modelli)

- Processo di identificazione dei pattern o modelli davvero utili e interessanti, sulla base di misure specifiche.
- Serve a selezionare la conoscenza realmente rilevante da quella meno significativa.

6. Knowledge Presentation (Presentazione della Conoscenza)

- Visualizzazione e rappresentazione della conoscenza trovata, per facilitarne l'interpretazione da parte degli utenti.
- Si usano grafici, report, cruscotti o altre forme di presentazione chiara ed efficace.

Data Mining Descrittivo e Predittivo

Il **data mining descrittivo** si occupa di analizzare e descrivere le proprietà e le caratteristiche dei dati presenti nel dataset. L'obiettivo principale è individuare pattern, relazioni e riassunti che aiutano a comprendere la struttura dei dati, senza necessariamente fare previsioni. Alcuni esempi di tecniche

descrittive sono il clustering, la ricerca di regole di associazione e l'analisi delle correlazioni.

Il **data mining predittivo**, invece, mira a costruire un modello (attraverso un processo di induzione) che sia in grado di prevedere valori futuri o sconosciuti di una variabile target, utilizzando le altre variabili disponibili nel dataset. Le tecniche predittive più comuni includono la classificazione, la regressione e il forecasting.

Tipi di dati che possono essere analizzati con il data mining

È possibile applicare tecniche di data mining su diverse tipologie di dati:

- **Dati di database:** informazioni organizzate in tabelle relazionali tradizionali (ad esempio, database gestionali).
- **Data warehouse:** insiemi di grandi quantità di dati, spesso provenienti da fonti eterogenee e integrati in un'unica struttura per l'analisi.
- **Dati transazionali:** dati relativi a transazioni effettuate, ad esempio registrazioni di acquisti, vendite, movimenti bancari.
- **Altri tipi di dati:** possono includere dati non strutturati, dati testuali, dati multimediali, dati web, serie temporali e dati spaziali.

Queste categorie coprono quasi tutte le fonti di dati che possono essere oggetto di estrazione di conoscenza mediante tecniche di data mining.

Database Data

Un sistema di database, chiamato anche **Database Management System (DBMS)**, è formato da una collezione di dati correlati (database) e da un insieme di programmi software che permettono di gestire e accedere a questi dati.

Per capire meglio, si può immaginare una struttura molto simile a un foglio di calcolo, con colonne e righe che rappresentano come vengono organizzati e trattati i dati all'interno di un database.

	A	B	C	D
1	Company Name	Profit January 2022	Profit February 2022	Profit March 2023
2	Company_A	20000	35000	28000
3	Company_B	30000	42000	38000
4	Company_C	25000	34000	37000
5	Company_D	20500	23000	27000
6	Company_E	32000	22000	28000
7	Company_F	33000	28000	26000

Quando viene richiesto di trovare l'**attributo** che identifica in modo **univoco** i dati di un'azienda in un database, bisogna individuare la colonna (o attributo) che contiene valori diversi per ciascuna azienda e che quindi permette di distinguerle senza ambiguità.

Questa colonna prende il nome di **chiave primaria (primary key)**: è un attributo, o un insieme di attributi, che permette di identificare univocamente ogni record (ogni azienda, in questo caso) nella tabella del database.

Se si parte dal presupposto che nessuna azienda abbia lo stesso nome, l'attributo "Company Name" permette di identificarla in modo univoco nel database ed assume il ruolo di chiave primaria. Nel caso specifico, quindi, il Company Name rappresenta la chiave primaria della tabella, cioè l'**attributo che distingue in maniera unica ogni azienda**.

Al contrario, attributi come il fatturato di febbraio (o quello di altri mesi) non identificano univocamente l'azienda, poiché valori identici possono essere condivisi da aziende diverse.

Un **database relazionale** è formato da una raccolta di tabelle, ognuna identificata da un nome unico.

Ogni tabella contiene un insieme di attributi (colonne, chiamate anche campi) e memorizza un elevato numero di **tuple** (record o righe), dove ciascuna tupla rappresenta un **oggetto**.

Ogni oggetto (tupla) è identificato in modo univoco da una **chiave primaria** (unique key/primary key) ed è descritto da diversi valori di attributo.

Le relazioni tra le varie tabelle vengono stabilite tramite le **chiavi esterne** (external keys), che permettono di collegare dati tra tabelle diverse all'interno del database.

Consideriamo AllElectronics, un'azienda fittizia.

L'azienda viene rappresentata tramite un database relazionale composto da diverse tabelle di relazione, ciascuna dedicata ad aspetti diversi dell'organizzazione. In particolare, le tabelle principali sono:

- customer (clienti)
- item (prodotti/articoli)
- employee (dipendenti)
- branch (filiali/sedi)

Queste tabelle consentono di modellare i dati fondamentali dell'azienda e di descrivere le sue relazioni interne.

Quando si lavora con i database, è utile considerare gli **schemi logici**. Questi rappresentano la struttura logica del database attraverso diversi elementi chiave:

- **Tabelle (Entità)**: punti di raccolta dati nel sistema.
- **Attributi (Campi)**: proprietà o caratteristiche delle entità.
- **Relazioni**: indicano come le diverse entità sono collegate tra loro (ad esempio, relazioni uno-a-uno, uno-a-molti).
- **Vincoli**: regole che governano come i dati possono essere inseriti o collegati, tra cui chiavi primarie, chiavi esterne, vincoli di unicità.

Gli schemi logici aiutano a progettare il database in modo chiaro e sicuro, garantendo che i dati siano ben strutturati e che le relazioni tra le parti siano esplicite.

Gli schemi logici del database di AllElectronics sono rappresentati dalle seguenti tabelle di relazione, ognuna con i propri attributi:

- customer: contiene dati come ID cliente, nome, indirizzo, età, occupazione, reddito annuo, informazioni di credito, categoria, ecc.
- item: include ID prodotto, marca, categoria, tipo, prezzo, luogo di produzione, fornitore, costo, ecc.
- employee: gestisce ID dipendente, nome, categoria, gruppo, salario, commissione, ecc.

- branch: dati relativi a ciascuna sede (ID sede, nome, indirizzo, ecc.).
- purchases: registra le transazioni d'acquisto (ID transazione, ID cliente, ID dipendente, data, ora, metodo di pagamento, importo).
- items sold: elenca per ogni transazione i prodotti venduti (ID transazione, ID articolo, quantità).
- works_at: tiene traccia di dove lavorano i dipendenti (ID dipendente, ID sede).

In sintesi, ogni riga di queste tabelle rappresenta un record, mentre gli attributi tra parentesi identificano le proprietà di ciascuna entità nel database.

L'accesso ai database relazionali avviene tramite i linguaggi di interrogazione basati su **query**, come **SQL**. Questi consentono di eseguire interrogazioni per recuperare informazioni dai database. Le istruzioni più comuni sono join, selezione e proiezione, che permettono di:

- unire dati da più tabelle,
- selezionare solo i record di interesse,
- visualizzare specifici attributi.

Un esempio di query potrebbe essere "Mostrami la lista di tutti gli articoli venduti da gennaio 2022".

Inoltre, i linguaggi relazionali permettono di usare le **funzioni di aggregazione** (come somma, media, conteggio, massimo, minimo) per svolgere analisi più complesse.

Ad esempio: "Mostrami il totale delle vendite dell'ultimo mese raggruppato per filiale".

Nel contesto dei **database relazionali**, il **data mining** viene utilizzato principalmente per cercare **pattern** (schemi ricorrenti) nei dati.

Ad esempio, analizzando i dati dei clienti si possono prevedere rischi di credito per i nuovi clienti, basandosi su variabili come reddito, età e precedenti informazioni creditizie.

Un altro compito fondamentale è l'**analisi delle deviazioni (deviation)**: il data mining aiuta a individuare prodotti che registrano vendite molto diverse rispetto alle aspettative, ad esempio confrontando le vendite con gli anni precedenti. Que-

ste deviazioni possono essere indagate ulteriormente per scoprirne le cause.

Il data mining può anche rilevare **cambiamenti nascosti**, come modifiche nel packaging di un prodotto, che potrebbero influenzare l'andamento delle vendite o altri indicatori aziendali.

Data Warehouse

Immagina che AllElectronics diventi una grande azienda internazionale con **diverse filiali** sparse nel mondo, e che ogni filiale gestisca i propri database separati.

Se volessi confrontare, ad esempio, l'andamento dei diversi tipi di articoli venduti in ogni filiale, sarebbe molto complicato lavorando solo con database relazionali separati: dovresti raccogliere dati da più fonti e integrarli manualmente.

In questi casi entra in gioco il **Data Warehouse**. Un Data Warehouse permette di **unificare e integrare** i **dati** provenienti da tutte le filiali, rendendo possibile analisi avanzate e centralizzate sull'intera organizzazione, superando i limiti dei semplici database relazionali distribuiti.

Un **data warehouse** è un archivio di informazioni raccolte da diverse fonti, integrate tramite uno schema unificato.

La costruzione di un data warehouse richiede **fasi** come: Data cleaning, Integration, Transformation, Loading e Refreshing.

I dati sono generalmente organizzati per tematiche rilevanti all'analisi decisionale: clienti, prodotti, fornitori, attività, ecc.

Nel data warehouse, i dati vengono salvati anche per **finalità storiche** (ad esempio, sintesi degli ultimi 6-12 mesi).

Struttura multidimensionale:

Il data warehouse possiede una struttura multidimensionale, detta **Data Cube**:

- Ogni dimensione rappresenta un attributo o un insieme di attributi (es.: periodo, prodotto, filiale).
- Ogni cella del cubo contiene un valore aggregato, come la somma delle vendite di un prodotto in una certa filiale in uno specifico mese.

Questo modello permette analisi complesse e flessibili sui dati aziendali, favorendo la visione d'insieme, la comparazione storica, la scoperta di pattern e trend.

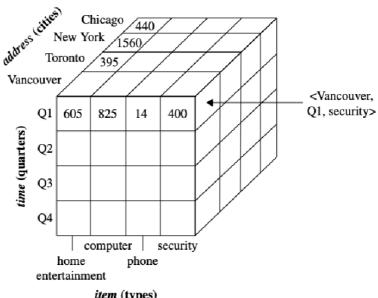
Data Warehouse | Drill-down, Roll-up

Un data warehouse consente di effettuare analisi dettagliate e flessibili sui dati grazie alle funzioni di drill-down e roll-up, che corrispondono a visualizzazioni differenti delle informazioni.

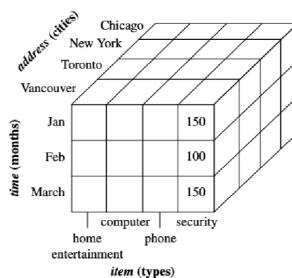
- **Drill-down:** consiste nel "scendere" nel dettaglio dei dati, focalizzandosi su attributi specifici (ad esempio, visualizzare le vendite per singolo prodotto piuttosto che per categoria).
- **Roll-up:** è l'operazione inversa, che permette di "raggruppare" i dati in maniera più compatta, aggregando gli attributi per ottenere una visione d'insieme (esempio: vendite totali trimestrali).

I **data warehouse** mettono a disposizione strumenti di **OLAP (Online Analytical Processing)**, che permettono analisi interattive sui dati multidimensionali a diversi livelli di granularità.

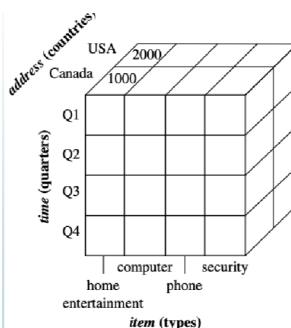
Esempio: le vendite di AllElectronics sono archiviate e analizzabili per tipologia di prodotto, periodo (trimestri) e località (città). Questo consente confronti rapidi e approfondimenti tra diverse dimensioni dei dati aziendali.



Drill-down on Time for Q1



Roll-up on address



Transactional Data

Ogni record in un database transazionale **rappresenta una transazione**, come un acquisto effettuato da un cliente, una prenotazione di volo o i click di un utente su una pagina web.

Di solito contiene un **transaction ID** (identificativo univoco della transazione) e una lista degli articoli coinvolti (prodotti acquistati).

Le **Transactional Databases** possono avere anche altre **tabelle** correlate, per descrivere meglio gli articoli acquistati (ad esempio: **descrizione, informazioni** sul negozio o sul venditore).

AllElectronics Transactional DB

Un esempio concreto di database transazionale per AllElectronics mostra una tabella con due colonne:

- **trans_ID:** identificativo della transazione (es. T100, T200)
- **list_of_item_IDs:** lista degli identificativi degli articoli acquistati in quella transazione (es. 11, 13, 18, 116)

Queste informazioni consentono di capire quali prodotti vengono spesso acquistati insieme (mar-

ket basket analysis), strategia utilizzata per ottimizzare promozioni e aumentare le vendite.

Altri tipi di Dati

Oltre ai dati tabellari e transazionali, il data mining può essere applicato a molti altri tipi di dati:

- **Dati temporali o sequenziali:** serie storiche, dati di borsa, record biologici, sequenze temporali.
- **Dati multimediali e ipertestuali:** testi, immagini, dati audio/video.
- **Dati a grafo e di rete:** dati provenienti da reti sociali o reti di informazione.

Esempi di applicazioni:

- Sulle borse valori si possono individuare trend utili per strategie di investimento.
- I flussi di dati delle reti informatiche vengono analizzati per rilevare anomalie (sistemi di rilevamento intrusioni).
- Le immagini possono essere utilizzate per addestrare modelli di machine learning che classificano oggetti e assegnano etichette semantiche, suddividendo dataset complessi in categorie precise.

Quali pattern si possono estrarre?

Nel data mining è possibile estrarre diversi tipi di pattern dai dati:

- **Classi o concetti:** rappresentano categorie o tipologie di dati (ad esempio, il segmento di clientela).
- **Associazioni e correlazioni:** relazioni che indicano quali elementi compaiono frequentemente insieme (regole di associazione).
- **Classificazione e regressione:** tecniche per prevedere valori (regressione) o assegnare una classe a ciascun dato (classificazione).
- **Cluster analysis:** identificazione di gruppi omogenei di dati all'interno del dataset senza conoscere le categorie a priori.
- **Outlier analysis:** individuazione di dati anomali che si discostano significativamente dal resto del dataset (outlier).

Classi e concetti

I dati possono essere associati a classi o concetti, che rappresentano categorie specifiche o raggruppamenti coerenti all'interno del dataset.

Ad esempio, AllElectronics vende diversi articoli che possono essere raggruppati per categorie di costo (come "Big Spender" oppure "budget Spenders").

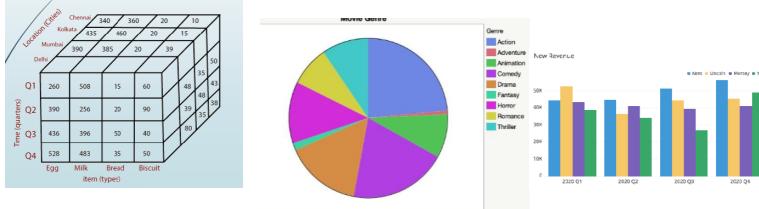
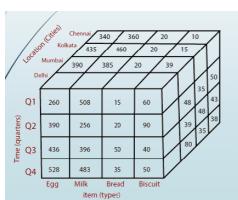
Le **classi** e i **concetti** sono descritte in termini dettagliati e queste descrizioni sono chiamate concetti o classi.

Si possono costruire tramite 2 approcci:

- La **caratterizzazione (Data Characterisation)** consiste nel **sintetizzare** le caratteristiche generali di una classe di dati (es. "strumenti software con vendite aumentate del 10% nell'ultimo anno").
- La **discriminazione (Data Discrimination)** invece mette a **confronto** le caratteristiche tra classi di dati diverse, evidenziandone le differenze principali.

Characterisation & Discrimination

Output examples: pie charts, bar charts, curves, multidimensional data cubes and multidimensional tables



Esempi pratici: Data Characterisation e Data Discrimination

Data Characterisation Task:

Si tratta di riassumere le caratteristiche dei clienti che spendono più di \$5000 all'anno da AllElectronics.

Risultato: si ottiene un profilo generale di questi clienti (per esempio: hanno tra i 40 e i 40 anni di età, sono occupati e possiedono ottimi rating di credito).

Data Discrimination Task:

In questo caso, il manager della customer relationship di AllElectronics vuole confrontare due gruppi di clienti secondo gli articoli acquistati. Più nello specifico, si vogliono confrontare i clienti che acquistano regolarmente gli stessi prodotti con quelli che lo fanno raramente.

Risultato: si ottiene un profilo comparativo dei gruppi di clienti. Ad esempio, l'80% di chi acquista spesso prodotti informatici ha tra i 20 e i 40 anni ed è laureato, mentre il 60% di chi fa acquisti rari degli stessi articoli non ha una laurea ed è più anziano o più giovane rispetto al primo gruppo.

Pattern frequenti

Nel data mining, è possibile individuare **pattern** analizzando la presenza ricorrente di alcuni elementi nei database.

Spesso, insiemi di oggetti appaiono con regolarità nello stesso dataset transazionale. Ad esempio, latte e pane vengono spesso comprati insieme.

L'analisi dei pattern frequenti permette di estrarre associazioni e correlazioni tra gli elementi presenti nei dati, rivelando legami utili per strategie di marketing, ottimizzazione delle vendite e comprensione dei comportamenti dei clienti.

Esempio: Association Analysis

Se vuoi sapere quali articoli vengono acquistati frequentemente insieme da AllElectronics, puoi utilizzare l'Association Analysis.

Un esempio di regola prodotta dall'Association Analysis è:

$\text{Buys}(X, \text{"computer"}) \Rightarrow \text{buys}(X, \text{"software"})$

[support = 1%, confidence = 50%]

Questo significa che ogni volta che X (un cliente) acquista un computer, con una certa frequenza (support) e probabilità (confidence), acquista anche un software.

La regola individua quindi un'associazione concreta tra i prodotti acquistati dai clienti, mostrando come computer e software siano spesso acquistati nello stesso ordine.

Spiegazione dettagliata: Support, Confidence e soglia nelle associazioni

- La regola di Association Analysis nel caso AllElectronics ($\text{Buys}(X, \text{"computer"}) \Rightarrow \text{buys}(X, \text{"software"})$) [support = 1%, confidence = 50%]) indica che il 1% di tutte le transazioni analizzate contiene sia computer che software (support).
- Il valore di confidence del 50% significa che, se un cliente acquista un computer, c'è il 50% di probabilità che acquisti anche un software.
- Le associazioni vengono tipicamente considerate utili solo se superano una certa soglia (threshold) di confidence stabilita dai data analyst.

Questi parametri sono fondamentali per selezionare solo quelle regole che risultano statisticamente significative e davvero rilevanti per le strategie aziendali.

Classificazione per Analisi Predittiva

La **classificazione** è un processo che consente di trovare un modello o una funzione capace di distinguere tra diverse classi o concetti di dati.

Per **addestrare** un modello di classificazione, si utilizza un training set, cioè un insieme di dati etichettati (con etichette già note).

Quando si esegue una classificazione, il modello viene poi utilizzato per fare **previsioni su nuovi dati**, sfruttando la conoscenza inferita durante l'addestramento.

Esistono diversi approcci per realizzare la classificazione:

- **IF THEN ELSE** (classificazione basata su regole)
- **Decision Tree** (rete di nodi che rappresentano diversi test)
- **Neural Network** (è necessario un training set per inferire la conoscenza dai dati etichettati)

Regressione

La regressione, a differenza della classificazione che restituisce valori **discreti** (etichette), produce previsioni su una scala continua.

Viene usata per stimare valori numerici mancanti o non disponibili, invece di assegnare semplici etichette di classe.

L'analisi di regressione è una **metodologia statistica** particolarmente adatta per le previsioni numeriche e per individuare i **trend di distribuzione nei dati**.

Mentre la classificazione necessita di dati etichettati, la regressione si basa su numeri per produrre **stime statistiche** (ad esempio, previsioni su trend futuri partendo da serie storiche di dati).

Esempio: Classificazione vs Regressione

Un sales manager di AllElectronics vuole **classificare** un insieme di articoli dello store in base a 3 tipi di risposta a una campagna di vendita:

- good response
- mild response
- no response

In questo caso, il sistema di classificazione assegna a ogni articolo una delle 3 possibili etichette (label), scegliendola tra quelle indicate.

La classificazione restituisce quindi un valore discreto (buona risposta, risposta moderata, nessuna risposta) per ogni elemento analizzato.

Se ti viene chiesto di stimare il possibile ricavo di una vendita futura di uno specifico articolo, dovrà fornire delle previsioni mese per mese.

In questi casi la classificazione non è adatta, perché serve una previsione numerica e non un'etichetta discreta.

La **regressione**, invece, fornisce una curva su un diagramma: permette di stimare in modo continuativo il valore previsto (ad esempio, ricavi mensili), visualizzando così chiaramente la tendenza dei dati.

Nell'esempio mostrato, la regressione lineare semplificata genera una retta che meglio approssima i punti osservati, rendendo possibile la previsione dei valori futuri.



Analisi dei cluster

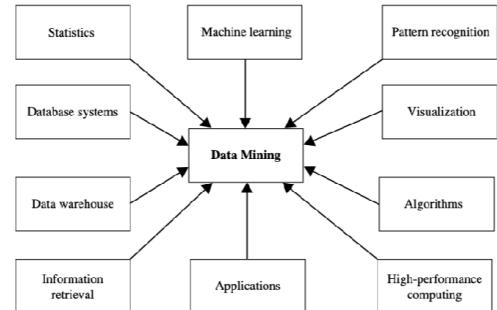
A differenza della classificazione e della regressione, la cluster analysis raggruppa i campioni di dati senza utilizzare etichette predefinite.

Nei casi in cui i dati etichettati non siano disponibili, la cluster analysis consente di suddividere i campioni in gruppi omogenei sulla base delle caratteristiche osservate.

Questa tecnica può anche essere utilizzata per generare nuovi dati etichettati (labels) e facilitare successive fasi di analisi.

Esempio: se ti viene chiesto di identificare dei sottogruppi tra i clienti di un'azienda, la cluster analysis ti permette di suddividere la clientela in gruppi distinti, ad esempio in base all'area geografica di provenienza.

Quali tecnologie sono utilizzate nel Data Mining?



Statistica

Il data mining è strettamente collegato alla statistica, che si occupa della raccolta, analisi e interpretazione dei dati.

Un **modello statistico** è costituito da un set of mathematical functions che descrivono come si comportano le variabili di un dataset.

Il comportamento di queste variabili viene descritto attraverso variabili casuali e le loro distribuzioni di probabilità.

La statistica permette sia di rappresentare il **"rumore"** presente nei dati, sia di sintetizzare e riassumere l'informazione contenuta nei dati raccolti.

Statistica: test, inferenza e significatività

La **statistica inferenziale** modella i dati per trarre conclusioni sulla popolazione oggetto di studio.

Il **Statistical Hypothesis Test** è essenziale per prendere decisioni statistiche sulla base di pattern osservati negli esperimenti. Un risultato è definito **Statistically Significant** quando non è dovuto al caso. Esistono diversi test che permettono di verificare se un risultato è davvero Statistically Significant oppure se potrebbe essere semplicemente frutto di casualità nei dati.

"Se si etichettano i dati per addestrare un modello allora si utilizzano un modello di ML supervisionato.

Informazioni raw (grezzo), dato atomico —> interrogo datawarehouse

Alla base del data management ci sono i singoli database, data warehouse, Enterprise DW e poi data mart per dipartimento"

Dalla slide si vede che le 3 tavole hanno una relazione :

Customer id sia chiave primaria sei esterna (nella prima tabella che si collega con customer id della seconda tabella ma che ha solo una chiave primaria che è order id e poi nella terza con customer come chiave esterna e come shipping id come chiave solamente primaria)"

Fondamenti di Structured Query Language (SQL):

Definizione e Manipolazione di Base dei Dati: La Tabella Customers

Per dimostrare le capacità di SQL, il processo inizia con la definizione di una tabella fondamentale denominata Customers (Clienti). La creazione di questa struttura dati avviene tramite il comando CREATE TABLE.

(VARCHAR(50) —> stringa variabile con una lunghezza massima di 50 caratteri

Successivamente alla definizione della struttura, sono stati inseriti dati campione utilizzando il comando INSERT INTO al fine di popolare la tabella per le successive operazioni di query.

```
• CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    age INT,
    country VARCHAR(50));

-- Insert sample data
INSERT INTO Customers (customer_id, first_name, last_name, age, country)
VALUES
    (1, 'John', 'Doe', 31, 'USA'),
    (2, 'Robert', 'Luna', 22, 'USA'),
    (3, 'David', 'Robinson', 22, 'UK'),
    (4, 'John', 'Reinhardt', 25, 'UK'),
    (5, 'Betty', 'Doe', 28, 'UAE');
```

L'Operazione di Selezione (SELECT) e Ordinamento

L'operazione SELECT è il comando cardine per interrogare il database e recuperare informazioni. Essa permette non solo di filtrare le righe (la vera selezione in termini di algebra relazionale), ma anche di definire quali colonne (attributi) devono essere visualizzate (la Projection).

Un esempio pratico dell'uso combinato di selezione e ordinamento si manifesta nella query che recupera i nomi e i cognomi dei clienti residenti negli Stati Uniti (USA), ordinandoli per età in ordine decrescente. La sintassi di tale operazione è:

```
SELECT first_name, last_name FROM Customers WHERE country = "USA" ORDER BY age DESC;
```

Questa istruzione dimostra come l'utente possa applicare un filtro (WHERE country = "USA") per isolare un sottoinsieme di righe e allo stesso tempo applicare un ordinamento (ORDER BY age DESC) per presentare i risultati.

Customers				
customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Orders			
order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	3
4	Keyboard	400	1
5	Mousepad	250	2

Shipments		
shipping_id	status	customer
1	Pending	2
2	Pending	4
3	Delivered	3
4	Pending	5
5	Delivered	1

ti in base a criteri specifici, evidenziando prima i clienti più anziani tra quelli statunitensi.

L'Arte della Congiunzione (JOIN): Tipi e Semantica Operativa

L'operazione di JOIN (Congiunzione) è cruciale nell'SQL per combinare logicamente righe provenienti da due o più tabelle, basandosi su un attributo comune. Gli esempi forniti presuppongono l'esistenza di una seconda tabella, Orders (Ordini), che viene collegata alla tabella Customers attraverso la corrispondenza dei rispettivi customer_id.

Congiunzione Interna (INNER JOIN)

Il tipo di JOIN più restrittivo e spesso utilizzato è l'INNER JOIN (Congiunzione Interna), che è anche il tipo predefinito se si specifica semplicemente JOIN. L'obiettivo semantico di questa operazione è chiaro: restituire solo le righe dove esiste una corrispondenza tra l'attributo di congiunzione (customer_id) in entrambe le tabelle.

La query di esempio per la Congiunzione Interna seleziona il nome del cliente, l'articolo ordinato (item) e l'importo (amount) combinando i dati dei clienti e degli ordini:

- SELECT c.first_name, o.item, o.amount
- FROM Customers c
- JOIN Orders o ON c.customer_id = o.customer_id;

La clausola ON c.customer_id = o.customer_id definisce la condizione di unione.

Congiunzione Sinistra (LEFT JOIN)

A differenza della Congiunzione Interna, il LEFT JOIN (Congiunzione Sinistra) opera garantendo l'inclusione di tutte le righe della tabella di sinistra, anche se non esiste una corrispondenza nella tabella di destra.

La semantica operativa implica che vengano restituiti tutti i clienti, inclusi quelli che non hanno effettuato alcun ordine. Nel caso in cui un cliente non abbia ordini corrispondenti, i campi relativi all'articolo (item) e all'importo (amount) provenienti dalla tabella Orders risulteranno con valore NULL (che indica l'assenza di dati validi).

```
SELECT c.first_name, c.last_name, o.item, o.amount
FROM Customers c
LEFT JOIN Orders o ON c.customer_id = o.customer_id;
```

Congiunzione Destra (RIGHT JOIN)

Il RIGHT JOIN (Congiunzione Destra) è l'operazione speculare al LEFT JOIN. In questo caso, vengono restituite tutte le righe della tabella di destra (gli ordini nell'esempio), indipendentemente dal fatto che esista un cliente corrispondente nella tabella di sinistra.

Il significato è quello di visualizzare tutti gli ordini registrati. Se un ordine dovesse esistere senza un cliente associato (ad esempio, a causa di un'anomalia nei dati o un cliente non registrato), i campi relativi al cliente (first_name, last_name, ecc.) provenienti dalla tabella Customers risulteranno NULL.

L'istruzione che implementa questo comportamento è:

```
SELECT c.first_name, c.last_name, o.item, o.amount  
FROM Customers c  
RIGHT JOIN Orders o ON c.customer_id = o.customer_id;
```

Congiunzione Completa (FULL JOIN)

Il FULL JOIN (Congiunzione Completa) rappresenta l'operazione di unione più inclusiva, poiché restituisce tutte le righe da entrambe le tabelle, abbinando i record dove possibile.

L'interpretazione operativa è che vengono restituiti sia tutti i clienti che tutti gli ordini. Le regole per la gestione dei dati non corrispondenti sono simmetriche: se un cliente non ha ordini, le informazioni sull'ordine saranno NULL. Analogamente, se un ordine non ha un cliente corrispondente, le informazioni sul cliente saranno NULL. La query che esegue questa congiunzione è:

```
SELECT c.first_name, c.last_name, o.item, o.amount  
FROM Customers c  
FULL JOIN Orders o ON c.customer_id = o.customer_id;
```

La Proiezione (PROJECTION): Isolamento degli Attributi

La Projection (Proiezione) è un concetto fondamentale nell'algebra relazionale che, nell'ambito SQL, si realizza attraverso la specificazione delle colonne nel comando SELECT. Il suo scopo è selezionare solamente determinate colonne (attributi) da una tabella, riducendo così la dimensionalità del set di risultati.

Ad esempio, se l'obiettivo è visualizzare unicamente i nomi dei clienti e i paesi di provenienza, si utilizza una query che isola precisamente questi tre attributi, escludendo età, ID o altri dettagli. La query di esempio fornita dimostra questa pura operazione di proiezione:

```
SELECT first_name, last_name, country  
FROM Customers;
```

Il risultato di tale operazione, come indicato nel materiale, mostra solo le colonne specificate.

In sintesi, mentre l'operazione SELECT nel suo complesso si riferisce all'atto di interrogare e recuperare dati, la Projection si focalizza sulla scelta verticale degli attributi, risultando essenziale per focalizzare l'analisi su dati specifici.

L'Apprendimento Non Supervisionato e le Tecniche di Clustering: Struttura nel Dato Non Etichettato

Dall'Ignoto all'Apprendimento Non Supervisionato (Unsupervised Learning)

Il termine "Ignoto" (*Unknown*), in questo contesto, fa riferimento specifico alla **mancanza di dati etichettati**. Poiché mancano le etichette (*labels*), è necessario esplorare l'"Ignoto" senza alcuna forma di supervisione. Questo percorso introduce l'Apprendimento Non Supervisionato (**Unsupervised Learning**), una branca del *machine learning* dove gli algoritmi mirano a scoprire *pattern* (schemi o modelli) intrinseci nei dati senza l'ausilio di esempi etichettati o di una guida diretta. A differenza dell'Apprendimento Supervisionato (**Supervised Learning**), che opera con dati etichettati (coppie input-output), l'Apprendimento Non Supervisionato lavora con dati non etichettati, scoprendo autonomamente la struttura inherente.

Le **caratteristiche fondamentali** che definiscono l'Apprendimento **Non Supervisionato** sono distinte e cruciali:

- non vengono fornite etichette o *output target* (risultati attesi) durante la fase di *training* (addestramento), il che obbliga l'algoritmo a identificare autonomamente schemi, relazioni e strutture.

Di conseguenza, spesso non esiste una singola risposta considerata "corretta" (**single 'correct' answer**), e algoritmi differenti possono identificare strutture valide, ma diverse. Questo comporta una sfida nella fase di valutazione, poiché manca una Verità di Base (**Ground Truth**) con cui confrontare i risultati.

La Verità di Base e il Valore dell'Apprendimento Non Supervisionato

La Verità di Base (**Ground Truth**), nel campo della scienza dei dati, rappresenta i valori o le etichette veri e verificati utilizzati come riferimento (*benchmark*) per addestrare e valutare i modelli. Essa incarna l'esito o la classificazione attuale che un modello dovrebbe idealmente prevedere.

La **Ground Truth** è indispensabile nell'apprendimento **supervisionato**, dove i modelli imparano dai dati etichettati; esempi tipici includono immagini etichettate come "cat" o "dog" o recensioni di clienti classificate come "positive" o "negative". Durante la valutazione, le previsioni del modello vengono confrontate con la *Ground Truth* per calcolare metriche prestazionali come accuratezza, precisione e *recall* (sensibilità).

L'**Apprendimento Non Supervisionato** risulta particolarmente prezioso in diverse situazioni applicative: quando si dispone di grandi quantità di dati non etichettati, quando l'obiettivo è scoprire *patterns* nascosti senza nozioni preconcette, quando l'etichettatura dei dati risulterebbe proibitivamente costosa o richiederebbe troppo tempo, oppure quando è necessario comprendere la struttura naturale dei dati prima di applicare altre tecniche.

A titolo di **esempio**, un sito di e-commerce potrebbe analizzare i dati di acquisto dei clienti e, grazie all'Apprendimento Non Supervisionato, potrebbe scoprire inaspettatamente che i clienti che acquistano attrezzi da giardinaggio comprano anche frequentemente mangiatoie per uccelli.

Questa è una connessione non prevista dai *marketer* (esperti di marketing) che apre **nuove opportunità** di cross-selling (vendita incrociata).

Nel caso di grandi quantità di dati non etichettati, una piattaforma di social media che traccia milioni di interazioni giornaliere può usare il **clustering** per identificare automaticamente diversi segmenti di utenti basati sui loro schemi comportamentali, rivelando gruppi distinti come "creatori di contenuti", "scorridori passivi" (*passive scrollers*) e "connettori sociali".

Un **esempio** in ambito medico dimostra come l'Apprendimento Non Supervisionato possa ridurre costi e tempi: in un progetto con 10.000 scansioni cerebrali, l'etichettatura manuale da parte di specialisti richiederebbe mesi e costi elevati.

L'Apprendimento Non Supervisionato può raggruppare scansioni simili, permettendo agli esperti di esaminare solo pochi esempi rappresentativi per ogni *cluster* (gruppo). Infine, comprendere la struttura naturale dei dati è utile prima di applicare modelli supervisionati: un'azienda che analizza le chiamate del servizio clienti potrebbe scoprire, tramite l'Apprendimento Non Supervisionato, che esistono sette tipi distinti di problemi, rispetto alle sole tre categorie utilizzate nel loro sistema manuale, aiutando a progettare un modello supervisionato più appropriato.

Fondamenti e Scopi del Clustering

Il **Clustering** è una tecnica fondamentale nell'ambito del *machine learning* e dell'analisi dei dati che ha come obiettivo il raggruppamento di punti dati simili tra loro, basandosi sulle loro caratteristiche o *features* (attributi). Essendo un metodo di Apprendimento Non Supervisionato, non necessita di dati etichettati per identificare gli schemi.

L'**obiettivo principale** del clustering è organizzare i dati in gruppi significativi dove si verificano 2 condizioni essenziali:

- in primo luogo, gli elementi all'**interno dello stesso cluster** (gruppo) devono essere **simili** tra loro;
- in secondo luogo, gli elementi in **cluster diversi** devono essere **dissimili** l'uno dall'altro.

La versione più semplice e basilare di analisi dei *cluster* è il **partizionamento** (*partitioning*).

Nel **clustering a partizionamento**, gli oggetti di un dato *dataset* (insieme di dati) vengono organizzati in diversi gruppi o *cluster* esclusivi. Un requisito fondamentale di questo approccio è che **il numero di cluster (\$k\$) deve essere fornito in anticipo**; esso costituisce il punto di partenza per i metodi di *clustering a partizionamento*.

Formalmente, dato un *dataset* D di n oggetti, e un numero k di *cluster* da formare, un algoritmo di partizionamento organizza gli oggetti in k partizioni, con la condizione che k sia minore o uguale a n.

Esistono diverse **metodologie** di **clustering** nella letteratura scientifica, le più popolari includono:

- **K-means** (Metodo a Partizionamento)
- Hierarchical clustering (Clustering Gerarchico)
- **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise - Clustering Spaziale Basato sulla Densità di Applicazioni con Rumore)

Il Clustering a Partizionamento: L'Algoritmo K-means

Il **K-means** è uno degli algoritmi di *clustering* più rappresentativi, noto per essere relativamente semplice e ampiamente utilizzato in settori quali la segmentazione dei clienti, l'analisi di immagini e il *document clustering* (raggruppamento di documenti). In questo algoritmo, la lettera "K" rappresenta il numero di *cluster* desiderati, che deve essere predeterminato dall'utente.

Il K-means opera attraverso un **processo iterativo**, affinando progressivamente i *cluster* fino al rag-

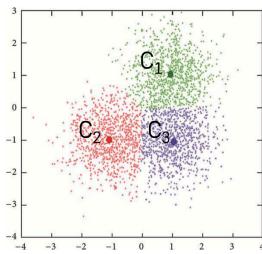
giungimento di una stabilità. L'obiettivo principale è raggruppare punti dati simili.

Pseudo-codice e Funzionamento di K-means

Il processo di K-means si svolge in fasi successive:

1. **Selezione di K:** Si definisce in anticipo il numero K di gruppi che si desidera trovare (ad esempio, K=3).
2. **Inizializzazione dei Centroidi:** Vengono posizionati casualmente K "centri" (detti **Centroidi**), che fungono da stime iniziali per il centro di ciascun gruppo.
3. **Assegnazione dei Punti:** Ogni punto dati nel dataset viene analizzato in base alla sua distanza dai K centroidi. Il punto viene quindi assegnato al gruppo il cui centroide è il più vicino. I punti più vicini al primo centroide formano il Gruppo 1, quelli più vicini al secondo formano il Gruppo 2, e così via.
4. **Ricalcolo dei Centroidi:** Per ogni gruppo appena formato, si calcola la posizione media di tutti i punti che ne fanno parte. Questa media definisce il nuovo centroide, che si spera sia una rappresentazione migliore della posizione effettiva del gruppo.
5. **Iterazione:** I passi di assegnazione (3) e ricalcolo dei centri (4) vengono ripetuti. I punti vengono riassegnati al nuovo centroide più vicino e i centri vengono nuovamente ricalcolati.
6. **Convergenza:** Il processo continua fino a quando i centri non si muovono più in modo significativo, o quando le assegnazioni dei gruppi non cambiano sostanzialmente. A quel punto, l'algoritmo ha trovato un insieme stabile di *cluster*.

Un **esempio visivo** tipico è dato da un diagramma a dispersione (**scatter plot**) che mostra una distribuzione di punti raggruppati con K=3, dove sono chiaramente identificabili i tre centroidi (C1, C2, C3) all'interno dei rispettivi *cluster*.



Vantaggi e Svantaggi di K-means

K-means presenta diversi **punti di forza (Pros)**:

- è concettualmente diretto e facile da implementare (**Simplicity**);
- è efficiente e veloce, con una complessità temporale lineare $O(n \cdot k \cdot d \cdot i)$, dove n è il numero di punti dati, k è il numero di cluster, d è la dimensionalità e i è il numero di iterazioni (**Efficiency**);
- funziona bene su dataset di grandi dimensioni e con molte dimensioni (**Scalability**);
- è garantita la convergenza verso un ottimo locale (**Guaranteed convergence**);
- è adattabile e può essere modificato (come in k-means++ o *mini-batch k-means*) per migliorarne le prestazioni; e i suoi risultati sono semplici da comprendere e spiegare (**Interpretability**).

Tuttavia, K-means ha anche **limitazioni (Cons)** significative:

- richiede obbligatoriamente la specifica del numero **k di cluster in anticipo**;
- è sensibile all'**inizializzazione**, poiché i cluster finali possono variare a seconda del posizionamento iniziale dei centroidi;
- è limitato a **cluster sferici** (*spherical clusters*) e funziona male quando i cluster hanno forme complesse;
- è sensibile agli **outliers** (valori anomali) che possono distorcere significativamente il calcolo della media (del centroide); la sua implementazione standard utilizza la **distanza Euclidea**, che potrebbe non essere appropriata per tutti i tipi di dati;
- a causa dell'**inizializzazione casuale**, non è deterministico, il che significa che diverse esecuzioni possono produrre risultati diversi.

È possibile trovare **soluzioni di clustering K-means** applicate a dataset di acquisti di clienti in repository GitHub specifici.

Il Clustering Gerarchico: Strutture Nidificate e Dendrogrammi

Il Clustering Gerarchico (Hierarchical Clustering) è generalmente considerato l'**opposto** dei metodi di *clustering* a partizionamento come K-means. A differenza del partizionamento, che crea una partizione piatta a livello singolo, il *clustering gerarchico* costruisce una **gerarchia a più livelli di cluster nidificati**.

Una **differenza** cruciale è che il *clustering gerarchico* **non richiede la specifica preventiva del numero di cluster**. Mentre il *clustering* a partizionamento ottimizza una funzione obiettivo globale, il gerarchico prende decisioni **locali** a ogni passo (quale cluster unire o dividere).

Inoltre, nel gerarchico, una volta che un punto è assegnato a un *cluster*, non può essere riassegnato, al contrario del K-means dove i punti possono muoversi durante l'ottimizzazione.

Il risultato principale di questa tecnica è un **dendrogramma** completo che mostra le relazioni tra i cluster a diversi livelli.

Esistono **2 approcci principali** nel *clustering gerarchico*:

1. **Agglomerativo (bottom-up - dal basso verso l'alto):** Inizia trattando ogni punto dati come un *cluster* separato. Procede progressivamente unendo i cluster più vicini tra loro, continuando fino a quando tutti i punti non appartengono a un unico *cluster*.
2. **Divisivo (top-down - dall'alto verso il basso):** Inizia con tutti i punti in un unico *cluster*. Divide ricorsivamente i cluster fino a quando ogni punto si trova nel proprio *cluster* individuale.

Procedura Agglomerativa e Metriche di Distanza

La procedura per il Clustering Agglomerativo si articola in 5 passaggi:

1. Assegnare ogni punto al proprio cluster individuale.
2. Calcolare le distanze tra tutte le coppie di cluster.
3. Unire i due cluster più vicini.
4. Aggiornare le distanze tra il nuovo cluster risultante e tutti gli altri cluster.
5. Ripetere i passi 3 e 4 fino a quando rimane un solo cluster.

La **distanza tra i cluster** stessi può essere misurata utilizzando diversi approcci, noti come **linkage**:

- **Single linkage** (Collegamento Singolo): La distanza è misurata tra i punti più vicini nei due cluster.
- **Complete linkage** (Collegamento Completo): La distanza è misurata tra i punti più lontani nei due cluster.
- **Average linkage** (Collegamento Medio): La distanza media tra tutte le coppie di punti tra i cluster.
- **Ward's method** (Metodo di Ward): Mira a minimizzare l'aumento della varianza all'interno del cluster (within-cluster variance) quando i cluster vengono uniti.

Il Dendrogramma

Il **Dendrogramma** è la rappresentazione visiva generata (spesso utilizzando il Metodo di Ward) per il clustering gerarchico agglomerativo. Ogni **linea verticale** nel diagramma rappresenta **l'unione** di due **cluster**. L'**altezza** a cui due cluster vengono uniti riflette la **distanza** (o dissimilità) tra essi.

Un **vantaggio** significativo di questo strumento è la possibilità di "tagliare" il dendrogramma a

qualsiasi altezza per determinare il numero di cluster desiderato; ad esempio, un taglio a una certa altezza specifica potrebbe produrre tre cluster.

Applicazioni, Vantaggi e Svantaggi

I **vantaggi (Pros)** del **Clustering Gerarchico** includono:

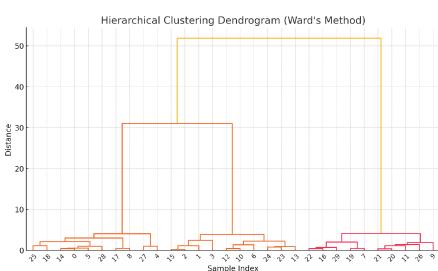
- la possibilità di una gerarchia visiva (il dendrogramma);
- non richiede la specifica preliminare del conteggio dei cluster;
- è versatile grazie alla possibilità di utilizzare diverse metriche di distanza;
- può scoprire cluster di varie forme;
- è molto utile per l'esplorazione della struttura dei dati.

I principali **svantaggi (Cons)** sono:

- è computazionalmente costoso, con una complessità temporale $O(n^3)$ (dove n è il numero di punti dati); es. 10 punti da clusterizzare, $10*10*10=1000$
- non è possibile annullare le decisioni di fusione o divisione (merge/split);
- è sensibile al *noise* (rumore) e agli *outliers*; e può risultare difficile da interpretare con dataset molto grandi.

Un **esempio** pratico di applicazione è l'**ottimizzazione della Catena di Fornitura** (*Supply Chain Optimization*) nel settore *retail* (vendita al dettaglio). Una catena nazionale con oltre 500 negozi affrontava sfide nella gestione dell'inventario, tra cui frequenti esaurimenti delle scorte (*stockouts*) in alcune sedi, inventario in eccesso in altre, aumento dei costi logistici e declino della soddisfazione del cliente.

Gli approcci tradizionali basati su zone geografiche o dimensioni dei negozi non riuscivano a catturare efficacemente i complessi pattern di domanda. Il team di analisi ha implementato il clustering gerarchico agglomerativo con Ward's linkage per identificare raggruppamenti naturali di negozi con pattern operativi simili. Hanno raccolto 18 mesi di



dati, includendo la velocità di vendita per categoria di prodotto, le fluttuazioni della domanda stagionale, i tassi di reso, i tempi di consegna e le caratteristiche del mercato locale. Analizzando il dendrogramma, sono stati identificati sette cluster distinti, che rappresentavano punti di rottura naturali nei pattern operativi, permettendo una gestione dell'inventario più mirata.

DBSCAN: Raggruppamento Basato sulla Densità e Rilevamento del Rumore

Il **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**, letteralmente *Clustering Spaziale Basato sulla Densità di Applicazioni con Rumore*, è un approccio di *clustering* basato sulla **densità**. Esso si distingue significativamente da K-means e dal *Clustering Gerarchico* per le sue capacità specifiche.

DBSCAN è progettato per trovare **cluster di forme arbitrarie** nei dati, identificando e isolando efficacemente i **punti di rumore** (*noise points*). A differenza di K-means, che tende a trovare *cluster* circolari o sferici, DBSCAN può identificare *cluster* di qualsiasi forma. Inoltre, come il *clustering* gerarchico, **non richiede di specificare in anticipo il numero di cluster**. DBSCAN classifica automaticamente gli **outliers** come punti di "rumore" che non appartengono a nessun *cluster*.

Parametri e Classificazione dei Punti in DBSCAN

Il funzionamento di DBSCAN si basa su 2 **parametri** fondamentali:

1. **Epsilon (\epsilon)**: La distanza massima tra due punti affinché siano considerati vicini (*neighbors*).
2. **MinPts**: Il numero minimo di punti richiesto per formare una regione densa.

L'algoritmo classifica i **punti** in **3 tipologie** specifiche:

- **Core points** (Punti Centrali): Punti che hanno almeno MinPts vicini entro la distanza \epsilon.
- **Border points** (Punti di Bordo): Punti che si trovano entro la distanza \epsilon da un punto centrale, ma che hanno meno di MinPts vicini.
- **Noise points** (Punti di Rumore): Punti che non sono né punti centrali né punti di bordo.

DBSCAN è particolarmente **utile** quando la quantità di *cluster* esistenti nei dati non è nota e quando i *cluster* possono avere forme complesse e non convesse, condizioni in cui algoritmi come K-means avrebbero difficoltà.

È possibile osservare la **differenza operativa** confrontando DBSCAN e K-means sulla stessa distribuzione spaziale di dati. Mentre la distribuzione spaziale dei *cluster* di K-means dipende strettamente dalle coordinate dei centroidi, DBSCAN è in grado di raggruppare forme complesse e isolare punti anomali, rappresentati visivamente da punti bianchi che DBSCAN raggruppa o identifica come rumore.

Dal Neurode al Perceptron Multistrato: Fondamenti Storici e Computazionali dell'Intelligenza Artificiale Ispirata al Cervello

L'Ispirazione Biologica e la Ricerca di Pattern nei Dati

L'evoluzione dei modelli di calcolo neurale trae origine dall'osservazione di processi cognitivi elementari presenti nel mondo animale, in particolare la capacità di discernere *pattern* (schemi o modelli) nell'ambiente circostante. Già negli anni '50, gli etologi avevano identificato schemi comportamentali significativi negli animali. Un esempio lampante è fornito dagli anatroccoli, i quali dimostrano la capacità di distinguere le proprietà degli oggetti che vedono muoversi subito dopo la schiusa, un fenomeno noto come *imprinting*. È stato notato

che gli anatroccoli di *Mullard* non solo possono eseguire l'imprinting sulla prima creatura vivente in movimento, ma anche su oggetti inanimati e, significativamente, su **concetti relazionali** che gli oggetti incorporano.

Se gli anatroccoli vedono inizialmente due oggetti rossi in movimento, in seguito seguiranno due oggetti che presentano la stessa relazione numerica (due oggetti), anche se il colore è differente (ad esempio, blu e non rosso). Questa osservazione rivela la capacità di individuare la *similarità*. Allo stesso modo, gli anatroccoli mostrano l'abilità di riconoscere la *dissimilarità*. Se i primi oggetti che vedono sono, ad esempio, un cubo e un prisma rettangolare, riconosceranno la differenza di forma; di conseguenza, seguiranno in seguito due oggetti diversi nella forma (come una piramide e un cono), ignorando due oggetti aventi la stessa forma. Tali capacità—rilevare somiglianze e dissomiglianze con una brevissima esposizione a stimoli sensoriali e agire in base a essi—rappresentano la base dell'intelligenza. Sebbene l'Intelligenza Artificiale contemporanea sia ancora lontana dall'implementare esattamente tali compiti, condivide con gli anatroccoli la capacità fondamentale di selezionare e apprendere *pattern* nei dati. Fu in questo contesto che Frank Rosenblatt inventò il **Perceptron** alla fine degli anni '50, un algoritmo innovativo e ispirato al cervello che poteva apprendere modelli nei dati semplicemente esaminandoli.

Le Radici Storiche del Neurode: McCulloch e Pitts

Le fondamenta teoriche del Perceptron risiedono in un articolo pubblicato nel 1943, frutto della collaborazione tra il neurofisiologo americano **Warren McCulloch** e l'adolescente prodigo **Walter Pitts**. McCulloch, con una formazione in filosofia, psicologia e medicina, si era precedentemente concentrato sulla neuroanatomia e sulla mappatura delle connessioni cerebrali delle scimmie, spostando successivamente la sua attenzione sulla "logica del

cervello". La ricerca di McCulloch era influenzata dalle suggestioni dei matematici e filosofi come Alan Turing, Alfred North Whitehead e Bertrand Russell, che avevano evidenziato i legami tra logica e computazione.

Il quesito centrale che guidava la ricerca di McCulloch era: se il cervello è un dispositivo computazionale, come implementa la logica, ad esempio una proposizione logica come "Se P è vera E Q è vera, allora S è vera"? Nel 1943, McCulloch e Pitts pubblicarono il saggio intitolato "*A Logical Calculus of the Ideas Immanent in Nervous Activity*", in cui proposero un modello semplice del **neurone biologico**.

Un neurone biologico è composto da un *Cell Body* (corpo cellulare) che riceve gli input tramite le sue proiezioni ramificate, i **Dendrites** (dendriti). Il corpo cellulare esegue un calcolo sugli input e, in base ai risultati, può inviare segnali *spiking* (a picco) lungo una proiezione più lunga denominata **Axon** (assone). Questo segnale viaggia attraverso i *Axon Terminals* (terminali assonici) per comunicare con i neuroni vicini.

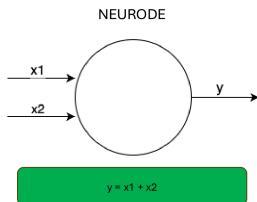
McCulloch e Pitts tradussero questo meccanismo biologico in un modello computazionale, l'**artificial neuron** (neurone artificiale), chiamato anche **Neurode** (un neologismo che unisce "neurone" e "nodo"). Il Neurode è in grado di implementare operazioni logiche **BOOLEAN** (booleane) di base come AND, OR, NOT, che costituiscono i mattoni fondamentali della computazione digitale. Il funzionamento di un **Neurode** (graficamente rappresentato come un cerchio che riceve input x_1 e x_2 e produce un output y) si basa su un calcolo sem-

- The NEURODE performs a simple computation:

- Given the Inputs: $x_1, x_2 \in \{0, 1\}$
- Sum the inputs.
- Compare the sum to a threshold (θ).
- Output: y equals 1 if the sum of x_1 and x_2 is greater than the threshold, 0 otherwise.

$$y = \begin{cases} 1 & \text{if } (x_1 + x_2) \geq \theta \\ 0 & \text{else} \end{cases}$$

plisce: dati gli input $x_1, x_2 \in \{0, 1\}$, il Neurode ne somma i valori e confronta la somma con una *threshold* (soglia), indicata con θ . L'output y è pari a 1 se la somma di x_1 e x_2 è maggiore o uguale alla soglia (θ), ed è 0 altrimenti. Formalmente:

$$y = 1 \text{ se } (x_1 + x_2) \geq \theta, \text{ altrimenti } y = 0.$$


L'Architettura del Perceptron di Rosenblatt e l'Introduzione dell'Apprendimento

Frank Rosenblatt partì dal contributo del Neurode e lo estese in un nuovo modello computazionale caratterizzato da **"neuroni artificiali che si riconfigurano mentre imparano, incorporando informazioni nella forza delle loro connessioni"**. Rosenblatt, uno psicologo, non disponeva di risorse hardware sufficienti per testare la sua teoria e pertanto utilizzò un **IBM 704**, un gigantesco computer di cinque tonnellate delle dimensioni di una stanza, presso il Cornell Aeronautical Laboratory. L'articolo del 1958 che presentava il Perceptron era intitolato "The Design of an Intelligent Automation: Introducing the Perceptron – a Machine that Senses, Recognises, Remembers, and Responds like the Human Mind". Sebbene Rosenblatt in seguito si rammaricò del nome Perceptron perché suonava troppo "macchina", intendeva con esso descrivere una classe di modelli del sistema nervoso.

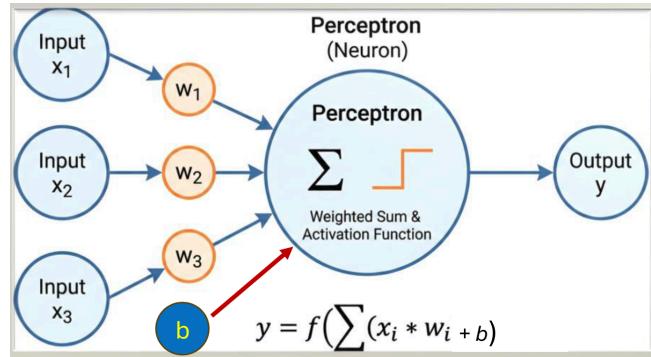
L'architettura del Perceptron introduce due elementi fondamentali che lo distinguono dal Neurode: i **weights** (w_i , pesi) e il **bias** (b , distorsione o termine di polarizzazione). Nel Perceptron, ogni input (x_i) è moltiplicato per il suo peso corrispondente (w_i).

Il calcolo eseguito dal Perceptron è una **somma pesata e una funzione di attivazione**: somma =

$w_1x_1 + w_2x_2 + \dots + w_nx_n + b$ Dove b è il termine di *Bias*.

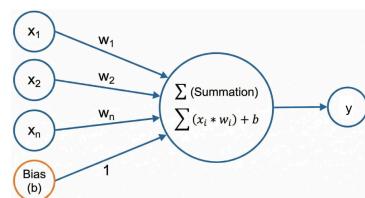
La funzione di attivazione (di soglia) determina l'output y :

- Se $\text{somma} > 0$, allora $y = 1$
- Altrimenti, $y = -1$



Si noti che l'output del Perceptron utilizza i valori 1 e -1, a differenza del Neurode che usava 0 e 1. La funzione di attivazione e la somma pesata sono rappresentate graficamente all'interno del nodo centrale del Perceptron.

Crucialmente, a differenza del Neurode, il Perceptron ha la capacità di **apprendere** il valore corretto per i pesi (w_i) e per il *Bias* (b) per risolvere un determinato problema.



Il Processo di Apprendimento del Perceptron (Learning Process)

Il processo di apprendimento del Perceptron si basa sull'aggiustamento iterativo dei pesi e del *Bias* in risposta agli errori di previsione, con l'obiettivo di trovare un confine decisionale ottimale.

Per illustrare il processo, si consideri un semplice Perceptron addestrato a prevedere se un individuo

gradirà una pizza, basandosi sul livello di piccantezza (x_1) e la quantità di formaggio (x_2).

Configurazione Iniziale:

- Pesi iniziali: $w_1 = 0.5$ (piccantezza), $w_2 = 0.5$ (formaggio).
- Bias iniziale: $b = -1$ (scelto arbitrariamente).
- Funzione di attivazione: Se (somma pesata + bias) > 0 , l'output è $+1$ (gradisce); altrimenti l'output è -1 (non gradisce).

Esempio di Addestramento (a): Classificazione Corretta

- Input: Poco piccante ($x_1 = 2$), molto formaggio ($x_2 = 3$). Target (etichetta di addestramento): $+1$ (gradisce la pizza).
- Calcolo della somma pesata: $y = (0.5 \times 2) + (0.5 \times 3) - 1 = 1 + 1.5 - 1 = 1.5$.
- Previsione: Poiché $1.5 > 0$, l'output è $+1$. La previsione è corretta, in quanto corrisponde al target.

Esempio di Addestramento (b): Misclassificazione e Aggiornamento dei Pesi

- Input: Molto piccante ($x_1 = 5$), poco formaggio ($x_2 = 1$). Target: -1 (non gradisce la pizza).
- Calcolo della somma pesata: $y = (0.5 \times 5) + (0.5 \times 1) - 1 = 2.5 + 0.5 - 1 = 2$.
- Previsione: Poiché $2 > 0$, l'output è $+1$ (gradisce la pizza).
- Errore: Vi è una **misclassification** (errore di classificazione), poiché il target è -1 ma la previsione è $+1$.

Aggiornamento dei Pesi (Weight Update) L'algoritmo di apprendimento aggiusta i pesi in base a questo errore di previsione, muovendo il confine decisionale. Supponendo un *Learning Rate* (tasso di apprendimento) $\alpha = 0.1$.

La regola di aggiornamento del peso è: $\Delta w_i = \alpha \times (\text{target} - \text{prediction}) \times x_i$.

1. Aggiornamento del peso w_1 : $\Delta w_1 = 0.1 \times (-1 - (+1)) \times 5 = 0.1 \times (-2) \times 5 = -1$.
2. Aggiornamento del peso w_2 : $\Delta w_2 = 0.1 \times (-1 - (+1)) \times 1 = 0.1 \times (-2) \times 1 = -0.2$.
3. Aggiornamento del Bias: $\Delta \text{bias} = 0.1 \times (-1 - (+1)) = -0.2$.

I **nuovi pesi** dopo l'aggiornamento sono:

- $w_1 = 0.5 - 1 = -0.5$
- $w_2 = 0.5 - 0.2 = 0.3$
- bias = $-1 - 0.2 = -1.2$.

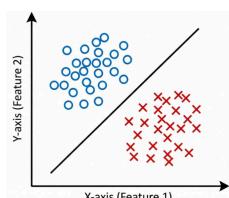
Il processo di addestramento continua finché non vengono soddisfatti specifici criteri di arresto. I criteri di convergenza (Convergence Criteria) includono la classificazione corretta di tutti gli esempi di addestramento o l'assenza di aggiornamenti dei pesi in un intero ciclo (epoch) sui dati di addestramento. I meccanismi pratici di arresto possono includere un limite massimo di iterazioni o l'assenza di misclassificazioni nell'ultimo ciclo completo.

Vincoli e Limitazioni del Perceptron: Il Concetto di Linearmente Separabile

Il Perceptron ha una limitazione fondamentale legata alla natura dei dati che può classificare: la **linear separability** (separabilità lineare).

I **Dati Linearmente Separabili** (Linearly Separable Data) sono un insieme di punti che possono essere perfettamente divisi in due classi distinte tracciando una singola linea retta (in due dimensioni) o un *hyperplane* (iperpiano) in dimensioni superiori. Questo concetto è cruciale perché determina l'efficacia dell'algoritmo Perceptron nel trovare un confine decisionale.

Quando i dati sono linearmente separabili, l'algoritmo



di apprendimento del Perceptron è **garantito di convergere**(converge), il che significa che troverà un insieme di pesi capace di classificare correttamente tutti gli esempi di addestramento dopo un numero finito di iterazioni.

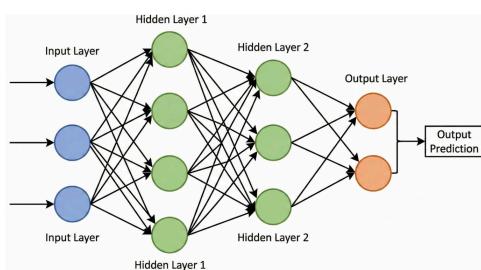
Al contrario, se i dati **non sono linearmente separabili**, il Perceptron non convergerà e continuerà indefinitamente a commettere errori di classificazione.

Un'immagine (un diagramma cartesiano) illustra questo concetto di separabilità lineare, mostrando punti dati di due classi diverse (cerchi blu e croci rosse) disposti nello spazio definito da due *features* (caratteristiche). L'asse X è la Feature 1 e l'asse Y è la Feature 2. Tali punti, che potrebbero rappresentare ad esempio i clienti in base alla localizzazione geografica e alla spesa, sono chiaramente separati da una singola linea retta, a conferma che sono linearmente separabili. L'applicazione pratica del Perceptron include la classificazione di spam/non-spam.

L'Evoluzione verso il Perceptron Multistrato (MLP)

Per superare le restrizioni del Perceptron, in particolare la sua incapacità di gestire dati non linearmente separabili, è stato sviluppato il **Multilayer Perceptron (MLP)**, che mantiene i concetti primari del Perceptron ma introduce maggiore complessità.

Le differenze principali nell'architettura MLP (visibile in un diagramma come una rete di nodi disposti in strati) includono:



1. Molteplici Strati: Un *Input Layer* (strato di ingresso), uno o più *Hidden Layers* (strati nascosti), e un *Output Layer* (strato di uscita).

2. Funzioni di Attivazione Non Lineari: L'introduzione di funzioni di attivazione non lineari (come la *sigmoid* o la *ReLU*, **Rectified Linear Unit**).

Il Perceptron Multistrato può modellare **confini decisionali complessi e non lineari** e apprendere **rappresentazioni gerarchiche delle caratteristiche**. Le relazioni intricate tra input e output sono modellate grazie agli strati nascosti, che imparano *higher level features* (caratteristiche di livello superiore) a partire dai dati grezzi.

Gli Strati Nascosti (Hidden Layers)

Gli strati nascosti rappresentano i livelli intermedi tra l'input e l'output. Matematicamente, la loro funzione è trasformare lo spazio degli input in uno spazio di *features* più separabile, rendendo possibili classificazioni complesse.

Funzioni e Meccanismi: Gli strati nascosti trasformano le caratteristiche di input attraverso combinazioni lineari pesate e applicano successivamente le **funzioni di attivazione non lineari**. Questo passaggio è essenziale per introdurre la complessità necessaria per gestire dati non linearmente separabili. Essi ricevono gli input pesati dallo strato precedente, applicano la funzione di attivazione (come ReLU o Sigmoid) e generano *transformed features* (caratteristiche trasformate) che vengono trasmesse agli strati successivi. Più si avanza negli strati, più le caratteristiche apprese sono di alto livello.

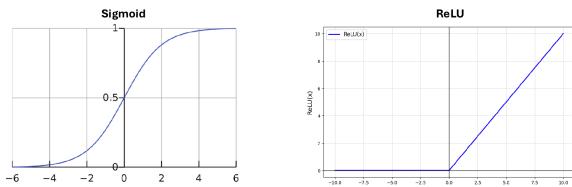
Caratteristiche Chiave: Gli strati nascosti sono spesso considerati una *black box* (scatola nera) poiché non sono direttamente osservabili dall'esterno della rete. Il numero di neuroni in questi strati determina la capacità rappresentativa del modello. I pesi all'interno dello strato nascosto

apprendono rappresentazioni di caratteristiche intermedie.

Funzioni di Attivazione Non Lineari: Le funzioni **Sigmoid** e **ReLU** sono introdotte per dotare l'architettura di non linearità, permettendo di trattare dati non linearmente separabili.

- **Funzione Sigmoid:** Graficamente, mostra una curva a S, con un range di output compreso tra 0 e 1, centrato a 0.5.
- **Funzione ReLU (Rectified Linear Unit):** Graficamente, la funzione è nulla per tutti i valori di input negativi ($x < 0$) e cresce linearmente per i valori positivi.

Il meccanismo di apprendimento in un MLP utilizza la **backpropagation** (retropropagazione), un processo per aggiornare i pesi attraverso tutti gli strati, ritornando indietro dall'output agli strati intermedi.



Rilevanza Contemporanea

Il Perceptron, pur nella sua semplicità, è fondamentale per l'apprendimento di concetti chiave nell'IA, inclusi il processo di addestramento, la somma pesata degli input, il *Bias*, il *Learning Rate*, l'*Prediction Error* (errore di previsione) e la comprensione dei dati linearmente separabili.

Il MultiLayer Perceptron (MLP) estende il Perceptron aggiungendo non linearità tramite le funzioni di attivazione e gli strati nascosti. L'MLP è ampiamente adottato oggi perché può inferire conoscenza da una vasta gamma di scenari reali, dove i dati sono ricchi di non linearità. Inoltre, l'MLP è spesso preferito per certe applicazioni grazie al suo costo computazionale relativamente contenuto rispetto ai metodi di *deep learning* (apprendimento profondo) più complessi.

In sostanza, se il Perceptron funge da semplice classificatore lineare—come un confine netto che separa i punti sul piano—il Perceptron Multistrato è come un sistema di canali complessi e interconnessi che possono curvarsi e intrecciarsi nello spazio dimensionale, consentendogli di distinguere forme e relazioni molto più complesse e sfumate.

Analisi dei Dati Testuali e Text Mining: Concetti Fondamentali, Applicazioni e Sfide nell'Era dei Modelli Linguistici di Grandi Dimensioni (LLM)

Contesto Accademico e Fondamenti Definitori

Il campo del **Text Mining** (Estrazione di Informazioni dal Testo) e della **Text Analytics** (Analisi del Testo) costituisce un pilastro fondamentale nei programmi avanzati di intelligenza artificiale, come dimostrato dalla sua inclusione nel programma post-laurea "AI for Business and Society" per l'Anno Accademico 2025-2026. Il Text Mining è una disciplina complessa e multidisciplinare che integra l'Elaborazione del Linguaggio Naturale (Natural Language Processing - NLP), la Classificazione di Pattern (Pattern Classification) e il Machine Learning (Apprendimento Automatico).

A causa della sua natura ibrida, il Text Mining non possiede una definizione canonica universalmente accettata. Per questo motivo, si ricorre spesso a definizioni mutuate dal più ampio campo del Data Mining (Estrazione di Dati). Una definizione essenziale per il Text Mining è "**l'estrazione non banale di fatti impliciti, precedentemente sconosciuti e interessanti da una raccolta di testi**". In parallelo, il Data Mining è definito come l'estrazione non banale di informazioni implicite, precedentemente sconosciute e potenzialmente utili dai dati. Dal punto di vista aziendale, il glossario Gartner descrive il Text Mining come "**il processo di estrazione di informazioni da raccolte di dati testuali e il loro utilizzo per obiettivi di business**". In tutti i casi, il Mining implica compiti quali la scoperta, la

ricerca, l'induzione e il perfezionamento (*refinement*), poiché le informazioni cercate sono spesso latenti e celate nel testo.

La Struttura del Contenuto Testuale e la Sua Rilevanza per il Text Mining

Il Text Mining si concentra sull'analisi e la modellazione del contenuto in linguaggio naturale. Una caratteristica cruciale dei dati testuali è che sono quasi sempre **non strutturati**, a differenza dei dati gestiti in ambienti come database o data warehouse. È essenziale distinguere tra le diverse tipologie di dati basate sul loro livello di organizzazione.

I **Dati Strutturati** (*Structured Data*) presentano un alto grado di organizzazione, tipicamente disposti in modo tabellare o simile a un foglio di calcolo. Esempi di questa tipologia includono file CSV (*Comma-separated value file*), tabelle di database relazionali e fogli di calcolo Excel. Si stima che i dati strutturati costituiscano circa il **20%** del volume totale dei dati a livello globale. A un livello intermedio si collocano i **Dati Semi-strutturati** (*Semi-structured Data*), che possiedono un certo grado di organizzazione, spesso caratterizzata da una struttura gerarchica. Ne sono esempi i file HTML, XML e Json. Infine, i **Dati Non Strutturati** (*Unstructured Data*) sono privi di un'organizzazione predefinita. Esempi di dati non strutturati includono documenti PDF, file Word e semplici file di testo. Attualmente, la maggior parte dei dati che vengono generati quotidianamente, come i post sui social media (ad esempio, Tweets e Facebook posts), ricade in questa categoria non strutturata.

Structured Data	Semi-structured Data	Unstructured Data
Excel spreadsheets; Comma-separated value file; Relational Database Tables. Around 20% of worldwide data is structured.	HTML files; Json files; XML files. Characterised by hierarchical structure.	PDF files; Word Files, Plain text files. Most of data that is created today is unstructured. (Tweets, Facebook posts, social media comments).

Distinzione Funzionale tra Text Mining e Natural Language Processing (NLP)

Nonostante la stretta integrazione tecnologica, Text Mining e NLP perseguono obiettivi diversi. Il **Natural Language Processing (NLP)**, le cui origini risalgono agli anni '50, ha l'obiettivo primario di comprendere il linguaggio umano, concentrandosi sull'analisi della sintassi grammaticale, del parlato o del testo. L'obiettivo storico dell'NLP è rendere i computer capaci di comprendere il linguaggio umano.

Al contrario, il **Text Mining** è impiegato per estrarre informazioni da contenuti, siano essi strutturati o non strutturati. Mentre l'NLP si concentra principalmente sul *significato* (la semantica) del contenuto, il Text Mining pone maggiore enfasi sull'estrazione della *struttura* (le relazioni, i fatti impliciti). Storicamente, le prime applicazioni di Text Mining (fine anni '90) applicavano algoritmi di Data Mining e Machine Learning direttamente sul testo, senza ricorrere alle tecniche di NLP. Solo successivamente il Text Mining ha incorporato l'NLP; tuttavia, negli ultimi dieci anni, i due campi hanno sviluppato intersezioni e punti comuni, pur mantenendo scopi finali distinti.

Nel contesto applicativo, le problematiche che il Text Mining è chiamato ad affrontare possono essere raggruppate in due scenari principali: utenti che hanno **domande specifiche ma non conoscono la risposta**, e utenti che hanno chiaro l'ambito o l'obiettivo generale ma **non hanno formulato domande specifiche**.

Domini di Applicazione e Dinamiche di Mercato della Text Analytics

Il Text Mining è una tecnologia pervasiva, spesso incorporata in diverse applicazioni aziendali e sociali. I suoi domini applicativi sono vasti e includono: Economia, Gestione Sociale, Servizi Informativi, Gestione del Rischio di Sicurezza, Servizio di Assi-

stenza Clienti (*Customer Care Service*), Rilevamento di Frodi (*Fraud Detection*), Business Intelligence, Analisi dei Social Media, Previsione dell'Abbandono del Cliente (*Customer Churn Prediction*), Sistemi di Domanda e Risposta (*Q&A Systems*) e Marketing.



Nel Marketing, l'analisi del sentimento (*Sentiment Analysis*), ad esempio, è uno strumento eccellente, utilizzato per categorizzare e valutare i risultati delle risposte ai sondaggi, comprendere l'esperienza complessiva del cliente (*customer experience*) e misurare l'interesse verso nuovi prodotti. Un esempio di integrazione profonda del Text Mining si trova nei sistemi Q&A, dove esso supporta compiti quali la ricerca nella base di conoscenza, l'inferenza e il filtraggio delle risposte potenziali, oltre all'analisi della domanda (*question parsing*).

Dal punto di vista economico, il mercato della Text Analytics è caratterizzato da una robusta crescita. I dati di mercato per il periodo 2018-2028 indicano un Tasso di Crescita Annuale Composto (*Compound Annual Growth Rate - CAGR*) del **17,35%**. Il grafico a barre fornito illustra chiaramente questa tendenza, mostrando che il volume di mercato previsto per il 2026 è sensibilmente superiore a quello registrato nel 2021. L'area geografica con la crescita più rapida è l'Asia Pacifico, mentre il mercato più grande in termini assoluti è il Nord America. Tra gli attori principali (*Major Players*) del settore figurano SAS, Microsoft, IBM, SAP e Clarabridge.

I Compiti Centrali del Text Mining Tradizionale

L'analisi testuale standard comprende sette compiti principali (*tasks*), che sono fondamentali per l'estrazione di conoscenza dai testi non strutturati.

Classificazione e Raggruppamento del Testo

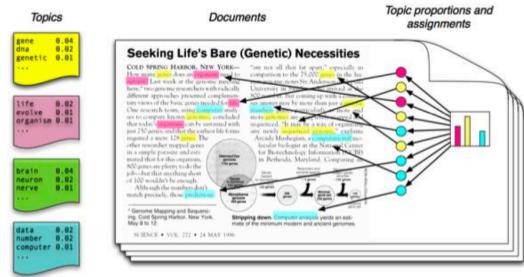
Il compito di **Classificazione del Testo** (*Text Classification*) mira a dividere un testo in categorie o tipi predefiniti. Essa è considerata una tecnologia di classificazione di pattern. Un esempio di questa operazione è la Classificazione Bibliotecaria Cinese, che raggruppa tutti i libri in cinque categorie e ventidue sottocategorie, basando la classificazione sul contenuto.

Diversamente, il **Clustering del Testo** (*Text Clustering*) è un processo che divide il testo in diverse categorie senza fare affidamento su categorie predefinite. Il numero e la natura delle categorie emerse dipendono da specifici indici e criteri di valutazione. Questo può portare alla clusterizzazione del testo in macro-aree tematiche (come sport, finanza o intrattenimento) o, basandosi su prospettive soggettive, in categorie come positive (atteggiamenti supportivi) o negative (atteggiamenti passivi). Il diagramma visivo del *Text Clustering* mostra la separazione di diversi punti colorati in tre gruppi distinti, circondati da ellissi, rappresentando il principio del raggruppamento dei dati per similitudine intrinseca.

L'Estrazione di Argomenti e Sentimenti

Il **Modello di Argomento** (*Topic Model*) è un approccio statistico che assegna un valore di probabilità di argomento a ciascuna parola. Un argomento è concettualmente definito da parole che condividono forti relazioni semantiche e concettuali, e ogni argomento è associato a un proprio dizionario specifico. La figura che illustra il *Topic Model* evidenzia come un singolo documento sia composto da una miscela di argomenti, ciascuno rappresen-

tato da una serie di parole chiave e probabilità (ad esempio, "gene", "dna", "genetic" con probabilità 0.04 e 0.02). L'immagine mostra come le parole nel testo vengano assegnate a specifici argomenti e come le proporzioni complessive dei topic all'interno del documento vengano visualizzate tramite un istogramma.



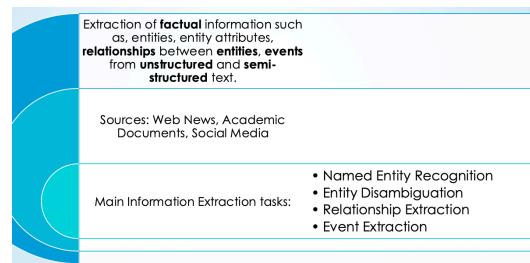
L'Analisi del Sentiment e Opinion Mining si concentra sull'estrazione dell'informazione soggettiva veicolata dagli autori, con l'obiettivo di rivelare il loro punto di vista e atteggiamento. I compiti principali includono la Classificazione del Sentiment e l'Estrazione degli Attributi. La Classificazione del Sentiment può essere considerata una forma specializzata di Classificazione del Testo, in cui la classificazione avviene su basi soggettive. Questa analisi è cruciale quando, per esempio, un'azienda cerca di monitorare tempestivamente le valutazioni dei clienti su un nuovo prodotto o quando la rete viene inondata da commenti in seguito a un evento.

Il Rilevamento e Tracciamento di Argomenti (Topic Detection and Tracking) si occupa di identificare e monitorare gli argomenti che polarizzano l'attenzione pubblica (*hot topics*) attraverso l'analisi di notiziari e commenti. Tali tecniche sono essenziali nell'analisi delle opinioni.

Estrazione di Informazioni e Riassunto Automatico

L'Estrazione di Informazioni (*Information Extraction - IE*) è il processo di estrazione di dati fattuali precisi – come entità, attributi di entità, relazioni tra entità ed eventi – da contenuti semi-strutturati

e non strutturati provenienti da fonti come documenti accademici, social media o notizie web.



L'IE include compiti specifici come il Riconoscimento di Entità Nominate (*Named Entity Recognition*), la Disambiguitazione di Entità (*Entity Disambiguation*), l'Estrazione di Relazioni (*Relationship Extraction*) e l'Estrazione di Eventi (*Event Extraction*). Le relazioni in IE si riferiscono a collegamenti semantici tra due o più concetti. L'Estrazione di Eventi è particolarmente tecnica: mentre nel linguaggio comune l'evento è una narrazione con riferimenti temporali e spaziali, nell'IE l'evento è un determinato stato o azione "attivato" da un verbo all'interno di una specifica struttura predicativa (*predicate framework*).

Infine, il **Riassunto Automatico del Testo** (*Automatic Text Summarisation*) è una tecnologia che utilizza metodi NLP per generare automaticamente riassunti. In un contesto di saturazione informativa, le aziende utilizzano questo strumento per estrarre rapidamente gli estratti più significativi dal testo. Tuttavia, un limite importante risiede nella difficoltà di creare strumenti "intelligenti" capaci di comprendere non solo la grammatica, ma anche la semantica profonda di un testo.

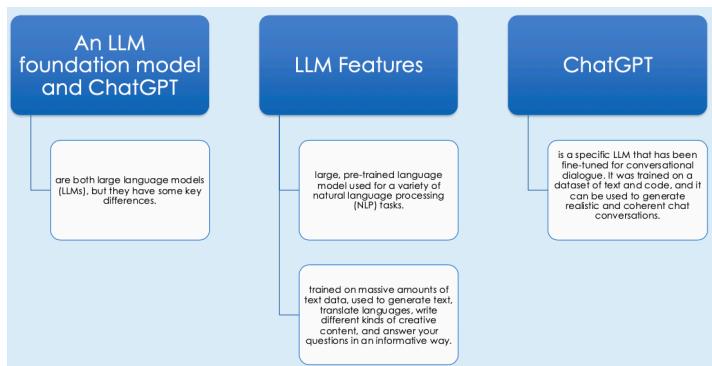
I Modelli Linguistici di Grandi Dimensioni (LLM)

I Modelli Linguistici di Grandi Dimensioni (LLM) hanno introdotto una nuova era per l'NLP e il Text Mining. Sebbene i metodi standard rimangano in uso, gli LLM sono oggetto di studio approfondito. Questi modelli, ancora in fase di sviluppo, hanno la capacità di eseguire una vasta gamma di compiti complessi. Tra le loro funzioni figurano la generazione di diversi formati di testo creativo (come co-

Feature	LLM foundation model	ChatGPT
Purpose	General-purpose NLP	Conversational dialogue
Training data	Massive amounts of text data	Dataset of text and code
Tasks	Generate text, translate languages, write different kinds of creative content, answer questions	Generate realistic and coherent chat conversations
Strengths	Versatility, accuracy	Fluency, realism
Weaknesses	Bias, computational cost	Limited scope

dice, poesie o email), la capacità di seguire istruzioni e completare richieste in modo ponderato e di rispondere a domande in modo completo e informativo, anche se aperte o inusuali.

Un concetto fondamentale è l'**LLM Foundation Model** (Modello Fondamentale LLM), definito come un modello linguistico pre-addestrato di grandi dimensioni utilizzato per una varietà di compiti di NLP. Questi modelli sono addestrati su enormi volumi di dati testuali e possono essere impiegati per la traduzione linguistica, la genera-



zione di testo e la risposta a domande. Le immagini fornite indicano la rapida diffusione di questi modelli ("It's raining LLMs"), citando esempi noti come **ChatGPT**, **Vicuna** (un Open-Source LLM), **Falcon LLM** (definito "King of Open-Source LLMs"), **Claude** (sviluppato da Anthropic) e **Gemini**, oltre a varianti di **Llama 3.1** con diversi parametri (8B, 70B, 405B).

È cruciale distinguere tra un Modello Fondamentale LLM generico e **ChatGPT**. La differenza risiede nella loro specializzazione. Un Modello Fondamentale LLM ha uno scopo di NLP generico ed è addestrato su quantità massicce di dati testuali, puntan-

do a versatilità e accuratezza. **ChatGPT è invece un LLM specifico che è stato ottimizzato (fine-tuned) per il dialogo conversazionale**. È stato addestrato su un dataset di testo e codice, e il suo compito principale è generare conversazioni di chat coerenti e realistiche, con punti di forza nella fluidità e nel realismo, pur avendo un ambito di applicazione più limitato rispetto al modello fondamentale non ottimizzato.

Principali Sfide e Limitazioni Tecniche

Il percorso del Text Mining e dell'NLP è costellato di sfide che limitano l'accuratezza e la completezza dei risultati.

Una difficoltà rilevante è rappresentata dal **rumore e dalle espressioni mal formate**. Mentre i testi formali (pubblicazioni accademiche, articoli politici) seguono regole semantiche standard, il testo online presenta una quantità significativa di espressioni non standardizzate o mal formate. Ciò riduce l'accuratezza dei compiti di NLP e Text Mining; per esempio, la segmentazione delle parole cinesi (*Chinese Word Segmentation*) vede i suoi tassi di accuratezza scendere al di sotto del 90% nel testo online, rispetto a oltre il 95% sui testi più strutturati.

Un'altra sfida complessa è l'**ambiguità e l'occultamento della semantica del testo**. Il linguaggio naturale è ambiguo: una parola come "Bank" può significare sia un istituto finanziario che la riva di un fiume, mentre "Apple" può riferirsi al frutto o al prodotto tecnologico. L'ambiguità si estende alla struttura sintattica delle frasi, come nel caso di "Ho visto un ragazzo con un telescopio", dove l'interpretazione (il ragazzo aveva il telescopio o io ho usato il telescopio per vedere il ragazzo?) rimane indefinita (contesto). Attualmente, i sistemi NLP trovano difficile risolvere queste ambiguità, e non esistono metodi completamente efficaci per superare tale ostacolo.

Inoltre, i metodi moderni di Text Mining, basati su Machine Learning e *Deep Learning*, richiedono **grandi quantità di dati annotati** per l'addestramento. La raccolta di tali dati da fonti online è resa difficile dalla presenza di contenuti protetti da copyright. Anche quando il copyright non è un problema, il contenuto online è spesso formattato male, rendendo necessari numerosi passaggi di pre-elaborazione. La difficoltà aumenta quando i contenuti riguardano aree specialistiche, richiedendo l'intervento di esperti per l'annotazione manuale, un processo dispendioso.

La Sfida della Rappresentazione Vettoriale

Una sfida specifica e fondamentale riguarda la **Rappresentazione Vettoriale delle Parole** (*Word-Vector Representation* o *Word Embedding*). Questa tecnica in NLP converte parole e frasi in vettori numerici. L'obiettivo è codificare le relazioni semantiche e sintattiche tra le parole. Un metodo di rilievo è **word2vec**, sviluppato da Google, che utilizza architetture come *continuous bag-of-words* (CBOW) e *continuous skip-gram*, entrambe basate sulla predizione delle parole circostanti data una parola centrale. I vettori numerici risultanti (Embeddings) rappresentano la parola; ad esempio, in una frase come "The King is born", ogni parola è associata a una colonna di valori numerici. La rappresentazione vettoriale riesce a catturare le dimensioni semantiche; ad esempio, associando concetti come *animal* (animale), *domesticated* (ad-domestico) o *fluffy* (soffice) a vettori numerici di parole specifiche (es. *dog*, *cat*).

Nonostante l'efficacia a livello lessicale, il limite risiede nel **collegamento della semantica delle singole parole alla semantica di insiemi più grandi di parole**, come frasi, proposizioni, paragrafi e la semantica del discorso (*discourse semantics*). Sebbene i metodi di Machine Learning siano efficaci nella rappresentazione semantica delle sin-

gole parole, l'elaborazione della semantica di intere frasi e paragrafi non risulta altrettanto immediata.

Estrazione e Analisi del Testo: Acquisizione, Pre-elaborazione e Strumenti di Linguistica Computazionale Avanzata

Il presente documento accademico esplora le metodologie e le tecniche fondamentali impiegate nell'ambito del *Data Mining* e della *Text Analytics* (Analisi del Testo), concentrandosi in particolare sulle fasi cruciali di acquisizione, annotazione e pre-elaborazione dei dati testuali, elementi essenziali per i programmi post-laurea in Intelligenza Artificiale (*AI for Business and Society*), come illustrato nella *Lecture 05: Text Mining 2* tenuta dal Prof. Alessandro Bruno. L'intero processo di estrazione della conoscenza dal testo può essere concettualizzato, come suggerito da un'immagine chiave, come un **imbuto di filtraggio** (funnel). Questo imbuto poligonale, composto da punti interconnessi, riceve una vasta quantità di dati dispersi e disordinati dall'alto (che rappresentano il "rumore" o i dati grezzi), elaborandoli e concentrando in un flusso ridotto e strutturato in uscita. Tale metafora visiva evidenzia l'obiettivo principale di queste discipline: trasformare i macro-dati non strutturati in informazioni concise e fruibili.

Acquisizione dei Dati: Fonti e Sfide di Dominio

L'acquisizione dei dati testuali rappresenta il primo passo in qualsiasi progetto di *Text Mining*. Le fonti di dati possono essere distinte in due categorie principali, ciascuna con le proprie implicazioni per il trattamento e la preparazione.

Dati di Dominio Aperto e Chiuso

I dati possono provenire da **Domini Chiusi** (*Closed Domain Data*) o **Domini Aperti** (*Open Domain*

Data). I Domini Chiusi sono ambienti specifici e altamente specializzati, come il campo finanziario o i sistemi sanitari (*Healthcare Systems*). In questi contesti, i documenti contengono termini molto specifici; ad esempio, nel campo finanziario si incontrano termini relativi a investimenti e finanziamenti, mentre nei sistemi sanitari le Cartelle Cliniche Elettroniche (*EHR - Electronic Health Records*) sono sature di terminologia medica.

Affidarsi esclusivamente ai dati di un dominio specifico potrebbe non essere sufficiente per l'analisi, poiché l'interpretazione di tali dati richiede una **conoscenza professionale di dominio**. Di conseguenza, le fonti di dati pubblicamente disponibili sono spesso utilizzate per compensare le informazioni mancanti o la ristrettezza del contesto nei Domini Chiusi.

I **Domini Aperti** includono fonti di dati accessibili al pubblico come Wikipedia, Baidu (un motore di ricerca e enciclopedia cinese), encyclopedie e libri di testo. Tuttavia, è fondamentale notare che i *Public Networks* (Reti Pubbliche) tendono a contenere espressioni molto più **rumorose** (*noisy*) e mal formate. Questa caratteristica richiede un notevole *overhead processing*, ovvero la necessità di eseguire complesse routine di pulizia e pre-elaborazione per rendere i dati utilizzabili.

Il Caso di Studio IMDB e l'Estrazione Web (Web Scraping)

Un esempio pratico di acquisizione di dati da Dominio Aperto è il caso di studio di IMDb (Internet Movie Database). IMDb è un sito molto popolare che fornisce agli utenti commenti e recensioni sui film, ed è ricco di collegamenti a contenuti cinematografici. Gli utenti possono visualizzare facilmente voti e commenti, con una grande quantità di recensioni consultabili cliccando sull'icona della stella gialla.

Il processo di **Estrazione Web** (*Web Scraping*) o **Web Crawling** (letteralmente, "scansione web") si pone come soluzione per scaricare automaticamente l'intero contenuto delle recensioni. Utilizzando il linguaggio di programmazione Python, strumenti come la libreria *urllib2* possono aiutare a scaricare l'intero contenuto della sezione desiderata. La possibilità di effettuare un *crawling* (scansione automatica) è essenziale, ad esempio, per scorrere tutte le recensioni degli utenti fino in fondo alla pagina.

Il Protocollo Robots.txt e le Considerazioni Etiche

Prima di procedere con l'estrazione automatica dei contenuti di una pagina web, è prassi comune e necessaria verificare il **protocollo robots** del sito. Questo protocollo è specificato in un file denominato **Robots.txt**. Il *Robots.txt* è un file che contiene tutte le limitazioni imposte dal proprietario del sito contro la scansione automatica (*crawling*). Ad esempio, l'URL specifico <https://www.imdb.com/robots.txt> contiene il protocollo con un elenco di limitazioni.

Nel contesto del caso IMDB, si è osservato che, non essendoci restrizioni in atto per scaricare il contenuto delle recensioni, il proprietario del sito consente agli utenti/sviluppatori di effettuare il *crawling* di tali contenuti. Sebbene il file *robots.txt*, introdotto come "*Robots Exclusion Protocol*", **non imponga vincoli legali** contro il *web scraping*, esso svolge un ruolo cruciale nello stabilire le aspettative e le autorizzazioni del proprietario del sito web riguardo all'accesso automatizzato. È inoltre raccomandato che il *crawling* dei contenuti avvenga durante periodi di bassa attività di rete, tipicamente durante la notte.

Per l'estrazione vera e propria dei contenuti della pagina web e per ottenere i collegamenti alla pagina successiva, un kit di strumenti (*toolkit*) basato

su Python chiamato **Beautiful Soup** è ampiamente utilizzato.

Processi di Pulizia e Annotazione dei Dati

Una volta acquisito il contenuto testuale, esso deve sottoporsi a processi di *data cleaning* (pulizia dei dati) e di pre-elaborazione per eliminare il **rumore** e prepararlo per l'analisi.

Rimozione del Rumore e dei Simboli Speciali

I contenuti delle pagine web sono spesso pieni di simboli speciali che non hanno significato semantico per l'analisi del testo. Il compito di analizzare e depurare il contenuto è chiamato **parsing**. Esempi di questi simboli speciali includono (che rappresenta uno spazio non divisibile, *non-breaking space*) e < (che rappresenta il simbolo "minore di", *less than*).

Una tabella chiarisce ulteriormente questi simboli e le relative entità HTML (*Entity Name*):

Result	Description	Entity Name
	non-breaking space	
<	less than	<
>	greater than	>
&	ampersand	&
¢	cent	¢
£	pound	£
¥	yen	¥
€	euro	€
©	copyright	©
®	registered trademark	®

La finalità della rimozione dei simboli speciali è la **riduzione del rumore** (*Noise Reduction*), che elimina caratteri irrilevanti che possono ostacolare l'analisi, e la **standardizzazione**, garantendo un formato di dati coerente. Ciò comporta un **miglioramento dell'accuratezza** (*Improved Accuracy*), focalizzando l'analisi sugli elementi testuali essenziali. L'impatto sul *Text Mining* è significativo, consentendo una rilevazione del sentimento più precisa (*Sentiment Analysis*), una più chiara identificazione dei temi sottostanti (*Topic Modeling*) e una maggiore accuratezza nella categorizzazione del testo (*Text Classification*).

Oltre ai simboli speciali, la fase di elaborazione del rumore (*Noise Processing*) rimuove elementi testuali come il simbolo "@" (spesso seguito da un nome utente) o collegamenti pubblicitari, in quanto non sono significativi per il *Text Mining*. L'eliminazione di questi elementi è tipicamente realizzata attraverso approcci basati su regole (*Rule-based*) o modelli (*template-based*).

Gestione dei Dati Brevi e Mappatura delle Etichette

All'interno dell'acquisizione dei dati, si effettua anche la rimozione dei commenti ritenuti troppo brevi e quindi potenzialmente privi di significato (*meaningless*). Per attuare questa rimozione, è necessaria la **segmentazione delle parole** (*word segmentation*) per ottenere un conteggio accurato. Mentre nel contenuto in lingua inglese il conteggio delle parole è relativamente semplice (si basa sul conteggio degli spazi), lingue come il cinese richiedono la combinazione di caratteri separati per formare parole. Un sistema basato su regole semplice può eliminare tutte le parole composte da un numero di caratteri inferiore a una certa soglia (ad esempio, meno di tre caratteri).

Un altro passaggio cruciale è la **Mappatura delle Etichette** (*Mappings of labels*), necessaria quando le etichette "nascoste" nel codice HTML del sito web differiscono per numero o nome da quelle previste da un classificatore. Questo passaggio è richiesto per risolvere qualsiasi ambiguità tra i due gruppi di categorie-etichette.

Un esempio illustrativo si verifica quando un punteggio di valutazione scaricato utilizza un sistema a 5 punti, ma il classificatore di sentimenti lavora solo con un sistema a 2 punti (ad esempio, Positivo/Negativo). Per evitare discrepanze, si procede alla mappatura: i voti 1 e 2 sono mappati come *Negative Feedback* (Feedback Negativo), mentre 4 e 5 sono mappati come *Positive Feedback* (Feedback Positivo). Il voto 3, che si trova a metà strada, potrebbe essere considerato un feedback neutro e viene rimosso per concentrare il classificatore sulle polarità chiare.

L'Annotazione dei Dati come Base per l'Apprendimento Supervisionato

L'**Annotazione dei Dati** (*Data Annotation*) costituisce il fondamento per i compiti di Apprendimento Automatico Supervisionato (*Supervised Machine Learning*). Essa trasforma i dati grezzi (come dichiarazioni estratte da Internet) in dati annotati. Ad esempio, le dichiarazioni possono essere etichettate come relative a "Marketing Topics" o "Topics different than Marketing" per addestrare un classificatore.

L'obiettivo è ottenere volumi maggiori di dati annotati e una copertura più ampia per migliorare la qualità dei risultati e le prestazioni dei modelli addestrati.

Tuttavia, il compito di annotazione non è sempre semplice e spesso richiede **conoscenza professionale di dominio** per applicare correttamente le etichette e identificare le parole pertinenti. Un esempio complesso tratto dal contesto sanitario mostra una descrizione post-operatoria di un paziente. Il testo grezzo, una volta annotato, identifica entità specifiche: il paziente (Sig. Shinabery), l'ospedale ([Surglute Leon Calcner Healthcare]Hosp), la data ([9/9/02]Time), i sintomi ([crescendo spontaneous angina]sym), la durata ([three-and-one-half months]dur), le malattie/diagnosi ([subacute left circumflex thrombosis]dis), i

trattamenti ([Dilation of the left circumflex]Treat, [placement of multiple stents]Treat), e il risultato del trattamento ([angiographic and clinical result]TR). Le categorie fondamentali identificate in questo contesto sono: Ospedale (*Hospital*), Tempo (*Time*), Sintomi (*symptoms*), Durata (*duration*), Malattia (*disease*), Trattamento (*Treatment*) e Risultato del Trattamento (*Treatment Result*).

L'annotazione dei dati non si limita al solo testo; nelle tecniche di *Data Mining* può essere estesa per includere l'**annotazione multimodale** di testo, video e immagini.

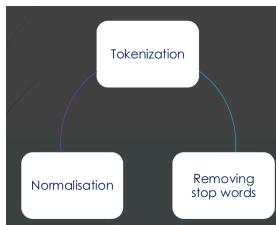
Per comprendere meglio il contesto dell'annotazione, è utile definire due concetti correlati:

1. **Lessico** (*Lexicon*): Generalmente ha una forma altamente strutturata, immagazzinando i significati e gli usi di ciascuna parola e codificando le relazioni tra parole e significati.
2. **Corpus**: Una collezione di testi o audio autentici (scritti o parlati da madrelingua o dialettali) organizzati in *dataset*. Un *corpus* può includere elementi come giornali, romanzi, ricette, trasmissioni radiofoniche, programmi televisivi, film e tweet. Nel contesto del *Natural Language Processing* (NLP), un *corpus* contiene i dati testuali e vocali utilizzati per addestrare sistemi di intelligenza artificiale e *machine learning*.

In senso lato, il **Text Mining** (Estrazione di Testo) è un campo interdisciplinare che coinvolge diverse aree come il *Data Mining*, l'Intelligenza Artificiale (AI), il *Machine Learning*, il *Natural Language Processing* (NLP) e la *Computational Linguistics* (Lingistica Computazionale). Le attività principali del *Text Mining* includono la Classificazione di Documenti (*Document Classification*), il Clustering di Documenti (*Document Clustering*), l'Estrazione di Informazioni (*Information Extraction*), l'Estrazione di Concetti (*Concept Extraction*) e il *Web Mining*.

Fasi Cruciali di Pre-elaborazione del Testo (NLP)

Immediatamente dopo l'acquisizione dei dati, entrano in gioco i passaggi di pre-elaborazione (*Preprocessing Steps*) fondamentali, molti dei quali derivano dal *Natural Language Processing* (NLP).



Tokenizzazione: La Segmentazione Lessicale

La **Tokenizzazione** (*Tokenization*) è un processo che segmenta un dato testo in **unità lessicali** chiamate "token". Ad esempio, la frase "That's" viene tokenizzata in due token: "that" e "'s"; mentre "rule-based" viene separato in tre token: "rule", "-", e "based".

Nelle lingue latine e flessive (come l'inglese), gli spazi e i segni di punteggiatura sono sufficienti per realizzare la lessicalizzazione. Al contrario, le lingue prive di marcatori di separazione (come il cinese) e alcune lingue agglutinanti (come il giapponese, il coreano e il vietnamita) richiedono prima un passaggio di **segmentazione delle parole**. La toolkit NLTK in Python fornisce un pacchetto dedicato alla tokenizzazione.

Rimozione delle Stop Words

La fase di rimozione delle **stop words** (parole funzionali o ausiliarie) mira a minimizzare lo spazio di archiviazione necessario per il *Text Mining*. Le stop words sono parole ad alta frequenza che appaiono comunemente nei documenti ma che veicolano poca informazione testuale. Esempi di queste parole funzionali includono articoli, preposizioni, congiunzioni e parole modali (come "The", "is", "at", "which", "on").

Durante la fase di rappresentazione del testo, le parole funzionali vengono scartate. Nell'implemen-

tazione di un modulo di *Text Mining*, viene stabilita una lista di *stop words*, e tutte queste parole vengono rimosse prima di procedere all'estrazione delle feature (caratteristiche).

Normalizzazione della Forma Verbale: Lemmatizzazione e Stemming

La **Normalizzazione della Forma della Parola** (*Word Form Normalisation*) è un processo cruciale per garantire che parole diverse, ma grammaticalmente correlate o derivanti dalla stessa radice, siano trattate come equivalenti. Questo concetto si articola in due tecniche principali: la **Lemmatizzazione** (*Lemmatisation*) e lo **Stemming**.

1. **Lemmatizzazione:** Consiste nel ripristino delle parole arbitrariamente deformate alle loro forme originali (o lemmi). Ad esempio, la forma plurale "cats" viene ripristinata a "cat", e la forma passata "did" a "do".
2. **Stemming:** È il processo di rimozione degli affissi (suffissi e prefissi) per ottenere la radice (root) della parola. Ad esempio, "fisher" viene ridotto a "fish", e "effective" a "effect".

La normalizzazione è solitamente realizzata tramite regole basate su espressioni regolari (*regular expressions*). Uno degli algoritmi di stemming più diffusi è l'algoritmo di **Porter stemming**, che consiste in quattro fasi principali:

1. Divisione delle lettere in vocali e consonanti.
2. Utilizzo di regole per processare le parole che terminano con i suffissi -s, -ing e -ed.
3. Progettazione di regole speciali per gestire suffissi più complessi (ad esempio, -ational).
4. Affinamento dei risultati del processo tramite regole aggiuntive.

Esistono diversi algoritmi di stemming che possono produrre risultati differenti anche per la stessa lingua. A livello applicativo, il toolkit NLTK in

Python fornisce funzioni per richiamare l'algoritmo di Porter stemming. Quando si impiegano metodi statistici, è fondamentale che le parole che condividono la stessa radice (*stem*) vengano considerate come la stessa parola; si pensi a forme diverse come "Take", "takes", "taken", "took", "taking".

Strumenti di Analisi Linguistica Avanzata

Per ottimizzare i compiti di *Text Mining*, entrano in gioco diversi processi avanzati di NLP, inclusi il *Part-of-Speech(POS) Tagging*, il Riconoscimento delle Entità Nominate (NER) e i *Syntactic Parsers*. Il POS Tagging, per esempio, elabora tutte le funzioni grammaticali svolte da ogni parola in una frase.

Riconoscimento delle Entità Nominate (NER)

Il **Riconoscimento delle Entità Nominate (NER - Named Entity Recognition)** è un componente di pre-elaborazione cruciale nei flussi di lavoro di analisi del testo e NLP, poiché l'integrazione di questo strumento migliora la comprensione e la preparazione del testo per le attività a valle.

Il NER offre molteplici vantaggi operativi:

- **Filtro Intelligente del Contenuto (Smart Content Filtering)**: Isola automaticamente le entità significative dal testo, permettendo di focalizzare l'analisi sulle informazioni chiave (ad esempio, estraendo figure di spicco e luoghi dagli articoli di notizie).
- **Pulizia del Testo Migliorata (Enhanced Text Cleaning)**: Rimuove in modo strategico il contenuto non essenziale, concentrando sul mantenimento delle entità identificate.
- **Creazione di Caratteristiche Avanzate (Advanced Feature Creation)**: Trasforma le entità riconosciute in featurepronte per il modello, aggiungendo contesto (ad

esempio, tipo di entità: Persona, Luogo, Organizzazione) per migliorare le prestazioni del modello.

- **Tecniche di Standardizzazione (Standardization Techniques)**: Crea rappresentazioni coerenti delle entità e può abilitare l'anomizzazione tramite *placeholder* (ad esempio, convertendo "Sarah Johnson" in <PERSON>).
- **Connessione alla Conoscenza (Knowledge Connection)**: Identifica le entità da collegare a database esterni o basi di conoscenza (ad esempio, collegando nomi di aziende alle loro voci nel database).
- **Segmentazione Intelligente del Testo (Intelligent Text Segmentation)**: Preserva le entità multi-parola come singole unità (ad esempio, mantenendo "San Francisco" come un'unica entità), migliorando l'accuratezza della tokenizzazione.
- **Elaborazione Specializzata (Specialized Processing)**: Si adatta alle esigenze specifiche del dominio (come termini medici o indicatori finanziari).

Per esempio, data la frase: "Tim Cook met with Microsoft executives in Seattle last Friday to discuss AI developments worth \$50 million," (Tim Cook ha incontrato i dirigenti di Microsoft a Seattle venerdì scorso per discutere sviluppi di intelligenza artificiale del valore di 50 milioni di dollari), il NER produce le seguenti mappature di entità:

- "Tim Cook" - **PERSONA** (People, including fictional)
- "Microsoft" - **ORG** (Organizzazione: Companies, agencies, institutions)
- "Seattle" - **GPE** (Geo-political Entity: Paesi, città, stati)
- "Friday" - **DATE** (Absolute or relative dates or periods)
- "AI" - **ORG** (Organizzazione)
- "\$50 million" - **MONEY** (Monetary values, including unit)

Analisi Sintattica e Alberi di Dipendenza

L'analisi sintattica (*Syntactic parsing*) è un altro strumento chiave che mira a estrarre la struttura delle frasi (*phrase structure*) e le relazioni di dipendenza. L'output di questo processo è un **albero della struttura sintattica** (*Syntactic structure tree*) della frase analizzata.

Per ottimizzare il *Text Mining*, si ricorre sia all'**Albero di Dipendenza** (*Dependency Tree*) che all'**Albero Sintattico** (*Syntax Tree*), che fornisce informazioni a livello di frase (*phrase-level information*).

Considerando una frase, ad esempio: "I drive a car to my college," (Io guido un'auto verso il mio college), è possibile visualizzare le differenze tra le due rappresentazioni strutturali. L'**Albero Sintattico** mostra una struttura gerarchica della frase. La frase viene scomposta in costituenti come Sintagma Nominale (*Noun Phrase*) e Sintagma Verbale (*Verb Phrase*), e utilizza **nodi intermedi** per raggruppare le parole in frasi, illustrando come le frasi sono costruite. Si concentra sulla composizione strutturale e sull'analisi a livello di frase.

L'**Albero di Dipendenza**, invece, mostra le **relazioni dirette tra le parole**. In questa rappresentazione, ogni parola si collega direttamente alla sua parola principale (*head word*). Le relazioni sono etichettate da funzioni grammaticali (ad esempio, *nsubj* per soggetto nominale, *dobj* per oggetto diretto, *prep* per preposizione, *pobj* per oggetto di preposizione), e non sono presenti nodi intermedi. L'Albero di Dipendenza è più efficace nel mostrare le **relazioni grammaticali** ed è particolarmente utile per l'estrazione di relazioni. Offre una rappresentazione più compatta e di più facile elaborazione computazionale, focalizzandosi sull'analisi a livello di parola e sulle connessioni grammaticali funzionali.

In sintesi, mentre l'Albero Sintattico aiuta a comprendere la composizione della frase e la struttura

gerarchica, l'Albero di Dipendenza è migliore per identificare le relazioni funzionali dirette tra le parole, essenziale per l'estrazione precisa di informazioni.

Data Mining, Analisi Testuale e Classificazione: Paradigmi e Architetture di Apprendimento Automatico

I. Introduzione e Contesto: Data Mining, Analisi Testuale e Classificazione del Testo

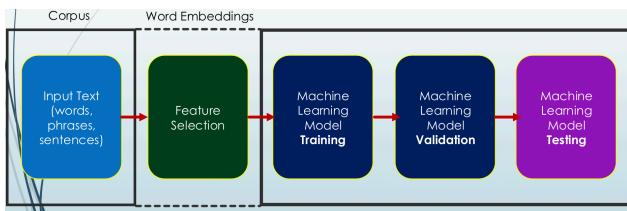
Un'applicazione cruciale in questo campo è la **Classificazione del Testo**, conosciuta anche come *text tagging* (etichettatura del testo) o *text categorization* (categorizzazione del testo). La classificazione del testo è definita come il processo di assegnazione di testo a gruppi organizzati. I classificatori di testo sono strumenti capaci di analizzare automaticamente il testo in input e assegnare un insieme di etichette o categorie predefinite basate sul suo contenuto. Ad esempio, un sistema di classificazione può prendere un input testuale come l'affermazione "You're the best" (Sei il migliore) e classificarla nell'Output (Uscita) come "Positive" (Positiva), tipico nei compiti in cui si desidera distinguere affermazioni "buone" da quelle "cattive".

II. Paradigmi di Apprendimento Automatico: Tradizionale vs. Moderno (Deep Learning)

Le numerose metodologie di *Machine Learning* (Apprendimento Automatico) applicate in svariati scenari possono essere raggruppate in due famiglie principali: gli approcci Tradizionali e gli approcci di *Newer Machine Learning* (Apprendimento Automatico più Recent). La distinzione cruciale tra queste famiglie risiede nel passaggio di *feature extraction* (estrazione delle caratteristiche).

A. Apprendimento Automatico Tradizionale

Negli approcci tradizionali di *Machine Learning*, l'estrazione delle *feature* (caratteristiche) è un passaggio separato e distinto da tutte le fasi coinvolte nell'addestramento (*training*), nella validazione (*validation*) e nel test (*testing*) del modello. Gli sviluppatori devono ricorrere a metodi di estrazione delle *feature* per configurare array e vettori che si adattino meglio all'esperimento o all'applicazione specifica. In certi casi, si possono usare descrittori statistici come *feature* rappresentative di un dato di input. Queste *feature* vengono poi fornite all'architettura di *machine learning* tradizionale per l'addestramento. Il concetto fondamentale in questo paradigma è che la bontà della classificazione dipende in gran parte dalla capacità dell'operatore di estrarre *feature* che aiutino efficacemente il modello a discriminare tra le categorie di interesse.



B. Apprendimento Profondo (*Deep Learning*)

Gli approcci più nuovi, in gran parte riconducibili al *Deep Learning* (Apprendimento Profondo), si basano sulle Reti Neurali Profonde (*Deep Neural Networks*, DNNs). A differenza dei metodi tradizionali, gli approcci di *Deep Learning* apprendono automaticamente le rappresentazioni gerarchiche, eliminando la necessità di un'ingegneria delle *feature* (ovvero, l'estrazione manuale delle caratteristiche). Questi modelli hanno la capacità di migliorare le loro prestazioni con l'aumento dei dati disponibili e della potenza computazionale (caratteristica nota come "scalabilità"). Sono particolarmente adatti a gestire enormi volumi di dati non strutturati, come immagini, testo e voce, dai quali sono in grado di estrarre schemi complessi.

III. Il Framework Tradizionale per la Classificazione del Testo

La struttura (*framework*) tradizionale per la classificazione del testo si articola in tre componenti essenziali e distinti: la Rappresentazione del Testo (*Text Representation*), la Selezione delle Caratteristiche (*Feature Selection*) e, infine, la Classificazione (*Classification*).

A. Rappresentazione del Testo e Vettorializzazione

La Rappresentazione del Testo svolge un ruolo cruciale, poiché deve riflettere fedelmente il contenuto testuale e disporre di una capacità sufficiente per distinguere i diversi tipi di testo. La selezione del metodo di rappresentazione è strettamente correlata all'algoritmo di classificazione scelto. Ad esempio, l'algoritmo *Support Vector Machine* (SVM) utilizza spesso il *vector space model* (VSM, Modello dello Spazio Vettoriale) come metodo di rappresentazione.

Per rendere il testo leggibile dal modello, l'input (che consiste in parole, frasi o periodi) deve essere trasformato in un formato numerico. Un concetto chiave in questo processo è l'*word embedding* (rappresentazione vettoriale di parola), che è definito come un array numerico utilizzato per rappresentare una parola. L'output di questa fase, insieme alla selezione delle *feature*, viene inviato alle fasi successive di *Training*, *Validation* e *Testing* del modello.

B. Selezione delle Caratteristiche e Algoritmi Tradizionali

La Selezione delle Caratteristiche è il processo mediante il quale viene scelto un sottoinsieme ottimale di *feature* per la rappresentazione e la classificazione del testo. Esistono due grandi categorie di approcci per la selezione delle *feature*: supervisionati e non supervisionati.

Gli **approcci non supervisionati** vengono applicati a un corpus (insieme di testi) che non dispone di annotazioni di categoria. Questi metodi si basano

comunemente su misure statistiche come la Frequenza del Termine (*Term Frequency*, TF) e la Frequenza del Documento (*Document Frequency*, DF). Al contrario, gli **approcci supervisionati** sfruttano l'annotazione di categoria, il che permette loro di selezionare un sottoinsieme di *feature* più efficace per il compito di classificazione.

Immediatamente dopo la rappresentazione e la selezione delle *feature*, viene applicato un algoritmo di classificazione. I più utilizzati nel *Machine Learning* tradizionale per la classificazione del testo sono *Naïve Bayes*, *Maximum Entropy*(Massima Entropia) e *Support Vector Machines* (SVM).

Il **Naïve Bayes** è un modello probabilistico versatile, impiegato sia per la classificazione binaria che per quella multi-classe. L'algoritmo **Maximum Entropy** è utilizzato nei compiti di classificazione in NLP (*Natural Language Processing*, Elaborazione del Linguaggio Naturale) e assegna la probabilità congiunta $P(x,y)$, che rappresenta la probabilità del dato osservato (x) e della sua etichetta corrispondente (y). La **SVM** (*Support Vector Machine*) è un algoritmo di apprendimento discriminativo supervisionato principalmente utilizzato per la classificazione binaria, il cui obiettivo primario è identificare un *iperpiano* all'interno dello spazio dei dati capace di separare i campioni in modo ottimale.

L'implementazione pratica di questi metodi, come SVM e Naïve Bayes, richiede una serie di passaggi tecnici preparatori, tra cui l'installazione delle librerie necessarie, l'impostazione del *random seed* (seme casuale per la riproducibilità), l'aggiunta del *corpus*, la pre-elaborazione dei dati (*Data Pre-processing*), la preparazione dei dataset di *Train* e *Test*, l'*Encoding* (Codifica) e la *Word Vectorisation* (Vettorizzazione delle Parole), prima di poter utilizzare i metodi ML per le predizioni finali.

IV. Architetture di Apprendimento Profondo (*Deep Learning*) e Meccanismi di Addestramento

Il Deep Learning offre architetture avanzate che possono automaticamente imparare e classificare i dati, in particolare quelli non strutturati.

A. Reti Neurali Feed-Forward Multistrato (MFFNN)

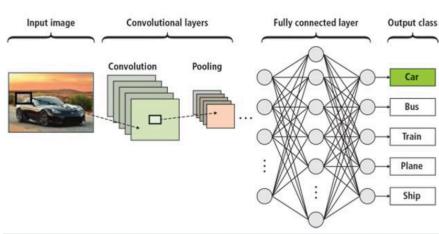
La Rete Neurale Feed-Forward Multistrato (*Multi-layer Feed-Forward Neural Network*) è una rete neurale con una struttura in avanti (*forward-structure*) che stabilisce una mappatura tra un insieme di vettori di input e un insieme di vettori di output in modo completamente connesso tra i suoi strati.

Un diagramma tipico di questa architettura mostra quattro strati distinti. Il primo strato è lo **Strato di Input** (Layer 1), dove vengono alimentate le variabili di input (ad esempio, quattro variabili di input), ma dove non avvengono calcoli. Seguono due **Strati Nascosti** (*Hidden Layers*, Layer 2 e 3). In ciascun neurone di questi strati, tutti i valori in ingresso vengono sommati (una somma pesata dei segnali di input) e successivamente processati attraverso una funzione di attivazione (*activation function*). La funzione di attivazione è un elemento cruciale all'interno del neurone artificiale. Infine, lo **Strato di Output** (Layer 4) riceve i valori sommati da tutti i nodi precedenti, li elabora con una funzione, spesso la funzione *softmax* in caso di classificazione, per produrre le probabilità.

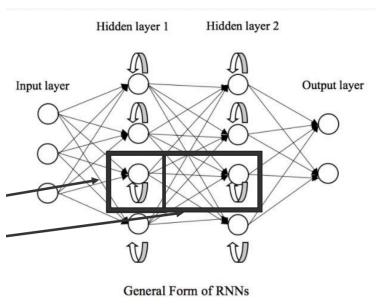
B. Architetture Specializzate: CNN e RNN

Le **Reti Neurali Convoluzionali (CNNs)** (*Convolutional Neural Networks*) costituiscono un tipo specifico di rete neurale feed-forward. I loro strati nascosti sono composti da una sequenza di filtri convoluzionali e di *pooling* (*sottocampionando il dato*). Gli strati convoluzionali hanno il compito di selezionare le *feature* dall'input di testo, mentre gli strati di *pooling* (*pooling layers*) riducono la dimensione (*downsample*) dei vettori di *feature* per ottenere una rappresentazione dei dati più compatta. L'adozione delle CNNs in compiti di Classifi-

cazione di Argomento e Classificazione di Sentimento (*Sentiment Classification*) dal 2014 ha portato a miglioramenti sostanziali delle prestazioni rispetto agli algoritmi ML tradizionali, confermando la loro elevata accuratezza sia in *Computer Vision* (Visione Artificiale) che in NLP. Il diagramma di una CNN, sebbene mostrato per la classificazione di immagini, illustra chiaramente la sequenza: *Input image* (Immagine di input), *Convolutional layers* (Strati convoluzionali, con convoluzione e *pooling*), *Fully connected layer* (Strato completamente connesso) e *Output class* (Classe di output).



Le **Reti Neurali Ricorrenti (RNNs)** (*Recurrent Neural Networks*) o Reti Neurali Ricorsive, rappresentano un'altra classe di reti neurali profonde che applicano il medesimo insieme di pesi in modo ricorsivo su un dato input, che può essere testuale o visivo. Queste architetture sono ampiamente impiegate nell'NLP per l'apprendimento di sequenze. La struttura ricorsiva opera sui nodi della rete e viene espansa in una sequenza. La feccia indietro, tiene conto non solamente del layer precedente ma di una serie storica. RNN: Ripetere in modo ricorsivo la funzione.



C. Il Processo di Addestramento e Retropropagazione

L'addestramento (*learning* o *training*) di una rete neurale è concepito come un processo di ottimiz-

zazione della *loss function* (funzione di perdita). L'obiettivo è determinare i parametri ottimali del modello che garantiscano il miglior adattamento ai dati di addestramento in relazione alla *loss function*. La *loss function* serve a misurare la distanza tra le predizioni generate dal modello e gli output desiderati.

Il meccanismo di **Backpropagation** (Retropropagazione) è ciò che consente di abbassare il valore della *loss function* aggiornando i parametri (o pesi) della rete. Il processo si articola in tre fasi:

1. **Forward Pass** (Passaggio in avanti): La rete esegue una predizione basandosi sui pesi correnti, confronta questa predizione con la risposta effettiva e calcola l'errore o perdita (*loss*).
2. **Backward Pass** (Passaggio all'indietro): Questa fase è il nucleo della *backpropagation*. Inizia dallo strato di output, calcola il contributo di ciascun peso all'errore totale, e si muove all'indietro strato per strato. Durante questo passaggio, vengono calcolati i *gradients* (gradienti), che indicano la direzione e la pendenza del cambiamento dell'errore (inclinazione della curva).
3. **Weight Update** (Aggiornamento dei pesi): Infine, ogni peso viene aggiustato in base al suo contributo all'errore, utilizzando un *learning rate* (tasso di apprendimento) che ne controlla l'entità. L'obiettivo è minimizzare l'errore nel *forward pass* successivo.

D. Iperparametri del Deep Learning

Gli iperparametri giocano un ruolo cruciale nella configurazione e nell'addestramento dei modelli di Deep Learning. Essi sono suddivisi per categoria:

- **Dataset:** Include il rapporto di divisione *Train-Test* (Addestramento-Test), che tipicamente segue la regola pratica 80/20.
- **Functions (Funzioni):** Riguarda la scelta dell'algoritmo di ottimizzazione, della fun-

- zione di attivazione utilizzata nello strato della rete neurale e della *loss function* (funzione di perdita) che il modello impiegherà.
- **Layers (Strati):** Comprende il numero di strati nascosti nella rete neurale, il numero di unità di attivazione in ogni strato, la dimensione degli strati convoluzionali e la dimensione del *pooling*.
 - **Iterations (Iterazioni):** Fa riferimento al numero di iterazioni (*epochs*) utilizzate nell'addestramento della rete neurale e alla dimensione del *batch* (*batch size*), ovvero il numero di campioni processati prima che il modello venga aggiornato.

V. Valutazione delle Prestazioni dei Sistemi di Classificazione

La valutazione delle prestazioni è indispensabile per misurare l'efficacia di un sistema di classificazione. A tal fine, vengono impiegati concetti chiave come *True Positive (TP)*, *True Negative (TN)*, *False Positive (FP)* e *False Negative (FN)*.

A. Matrice di Confusione e Definizioni di Errore

La **Matrice di Confusione (Confusion Matrix)** è uno strumento visivo e concettuale che permette di cogliere le prestazioni complessive di un sistema. In un contesto di classificazione, un *dataset* viene fornito con campioni e le loro etichette reali. Il sistema di classificazione esegue delle predizioni che vengono poi confrontate con queste etichette reali.

Se, ad esempio, consideriamo un *dataset* con affermazioni "bad" (cattivo) e "good" (buono), e definiamo "good" come la classe positiva e "bad" come la classe negativa:

		Prediction	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

- **True Positive (TP):** Il modello identifica correttamente un'istanza come appartenente alla classe positiva (es. un'affermazione "good" viene classificata come "good").
- **True Negative (TN):** Il modello identifica correttamente un'istanza come appartenente alla classe negativa (es. un'affermazione "bad" viene classificata come "bad").
- **False Positive (FP):** Errore in cui il modello identifica erroneamente un'istanza negativa come positiva (es. un'affermazione "bad" viene classificata erroneamente come "good").
- **False Negative (FN):** Errore in cui il modello identifica erroneamente un'istanza positiva come negativa (es. un'affermazione "good" viene classificata erroneamente come "bad").

Il diagramma della Matrice di Confusione organizza queste quattro possibilità incrociando l'Etichetta Reale (*Actual*) con la Predizione (*Prediction*), fornendo una visione strutturata degli esiti della classificazione.

B. Precisione, Richiamo e F1-Score

Per quantificare le prestazioni, vengono utilizzate metriche derivate direttamente dai valori TP, TN, FP e FN.

- **Precisione (Precision):** Misura la proporzione di predizioni positive che erano effettivamente corrette.
- **Richiamo (Recall):** Misura la proporzione di tutti i campioni positivi reali che sono stati identificati correttamente dal modello.
- **F1-Score (o F-Measure):** Questa metrica funge da media che tiene conto in qualche modo sia della Precisione che del Richiamo.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1 Score} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

La rappresentazione grafica o tabellare di queste tre metriche è utile per descrivere al meglio le prestazioni di un sistema di classificazione.

C. Esempio Pratico di Calcolo delle Metriche

Considerando un problema di classificazione su 100 affermazioni etichettate, un modello che produceesse i seguenti risultati: TP = 65, FP = 20 e FN = 15, le metriche di prestazione si calcolerebbero come segue:

- **Precisione:** $65 / (65 + 20) = 65 / 85 = 0.76$.
- **Richiamo:** $65 / (65 + 15) = 65 / 80 = 0.81$.
- **F-Measure:** 0.78 (ottenuto dal calcolo $1.231 / 1.57$).

Questi risultati (0.76 per la Precisione, 0.81 per il Richiamo e 0.78 per l'F-Measure) offrono una valutazione quantitativa delle prestazioni del sistema.

Inoltre, il confronto (*benchmarking*) tra due sistemi di classificazione testuale, come S1 e S2, richiede il calcolo e l'analisi di queste metriche per determinare quale sistema sia più performante sulla base dei risultati osservati. Ad esempio, confrontando S1 (TP = 65, FN = 5, FP = 30) con S2 (TP = 62, FN = 12, FP = 26), l'analisi delle metriche derivate sarebbe essenziale per la valutazione finale.