

Machine Learning

1 - Intro & Overview

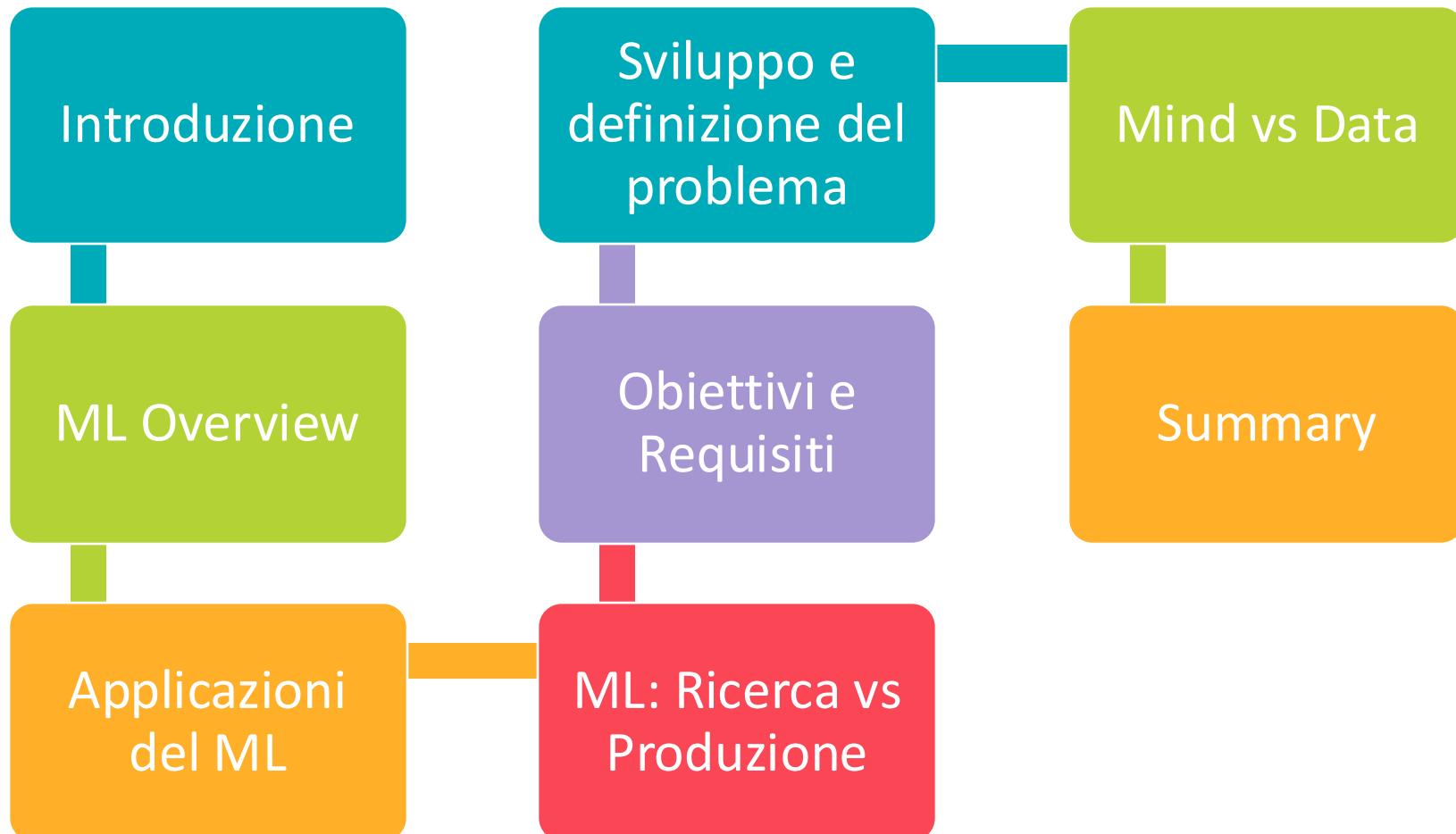


Prof. Giuseppe Polese

Prof.ssa Loredana Caruccio



Outline



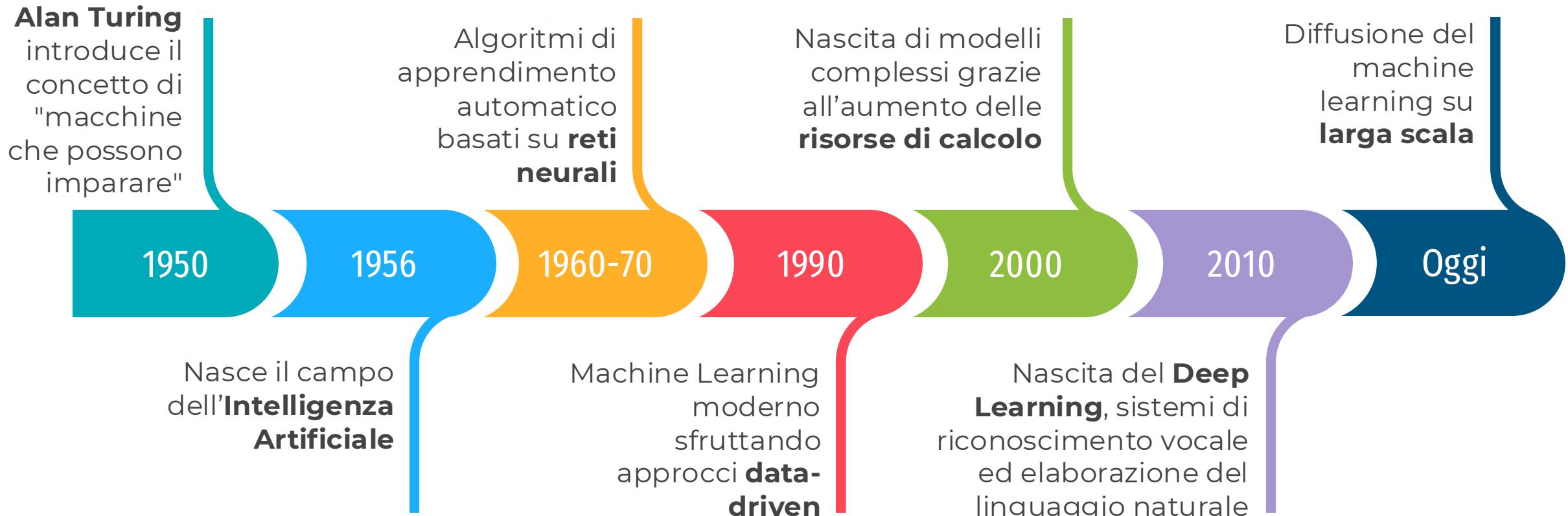
Introduzione

Introduzione

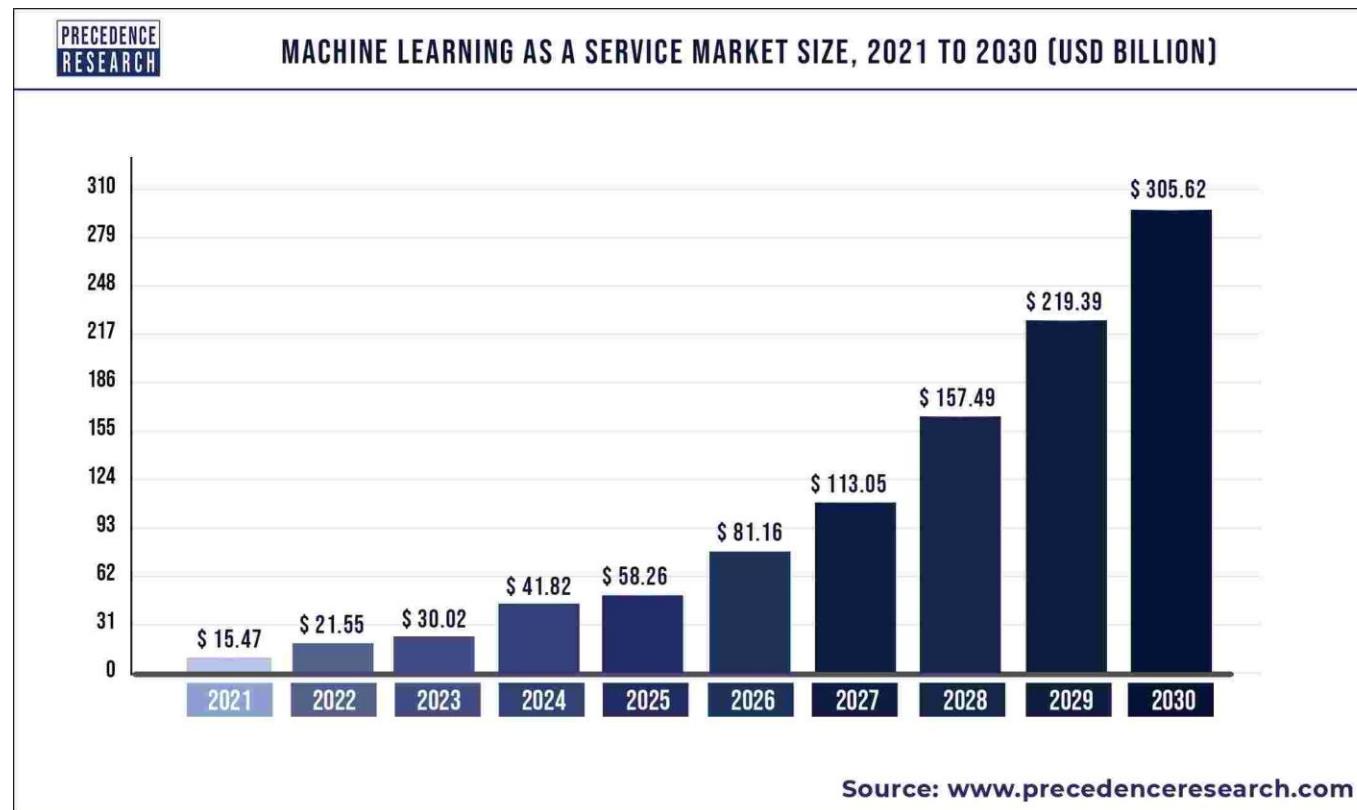
- Il termine **Machine Learning (ML)** si riferisce al campo di studio che consente ai computer di **apprendere dai dati senza essere esplicitamente programmati**
- Introdotto inizialmente per il riconoscimento ottico dei caratteri (OCR), ha avuto la sua prima applicazione diffusa con i **filtri antispam**
- Ad oggi è impiegato in molteplici applicazioni che usiamo tutti i giorni



ML Timeline



Applicazioni del ML



- Dal 2010 ad oggi, il ML si è diffuso in maniera capillare e le stime sono in continua crescita

Domini del ML oggi



Industria



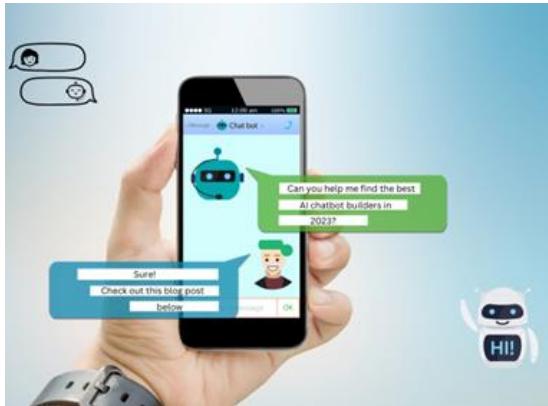
Meteo



Medicina



Marketing



Natural Language Processing



Internet of Things



Computer Vision



Gaming

Lo studio del ML oggi

- Quando si parla di ML di solito si pensa solo ai differenti **algoritmi** utilizzati
 - Regressione logistica, reti neurali, alberi decisionali...
- In realtà, gli algoritmi **sono solo una piccola parte** di un sistema di machine learning che viene messo in funzione
- Ad esempio, il sistema comprende i **requisiti aziendali** che hanno portato alla nascita del progetto, lo **stack dei dati** e la **logica di sviluppo**, il **monitoraggio** e l'**aggiornamento dei modelli**, l'**interfaccia** con cui utenti e sviluppatori interagiscono col sistema e così via...

Obiettivi Formativi

- Il corso fornisce gli strumenti metodologici e tecnologici fondamentali per **progettare ed implementare** sistemi di machine learning partendo da dati di vario genere.
- Gli studenti svilupperanno la capacità di **analizzare problemi reali** e di **scegliere le più opportune tecniche** di machine learning per la loro risoluzione.

Prerequisiti:

- Un corso di Fondamenti di Basi di Dati
- Un linguaggio di programmazione

Contenuti del corso

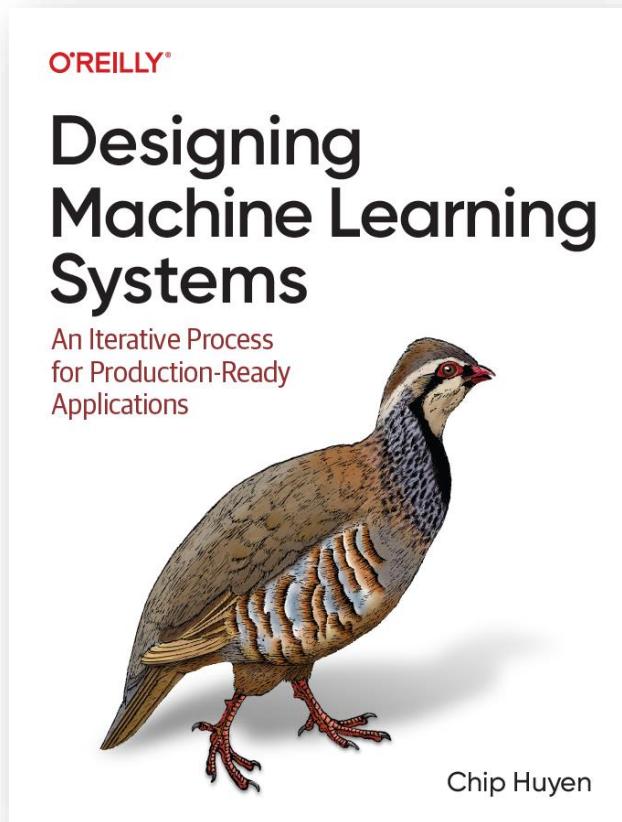
Teoria (32 ore)

- Introduzione al ML
- Progettazione di sistemi di ML
- Data Engineering
- Dati di addestramento
- Feature engineering
- Sviluppo e valutazione dei modelli di Machine Learning
- Cenni su modelli di ML
- Aspetti etici del ML

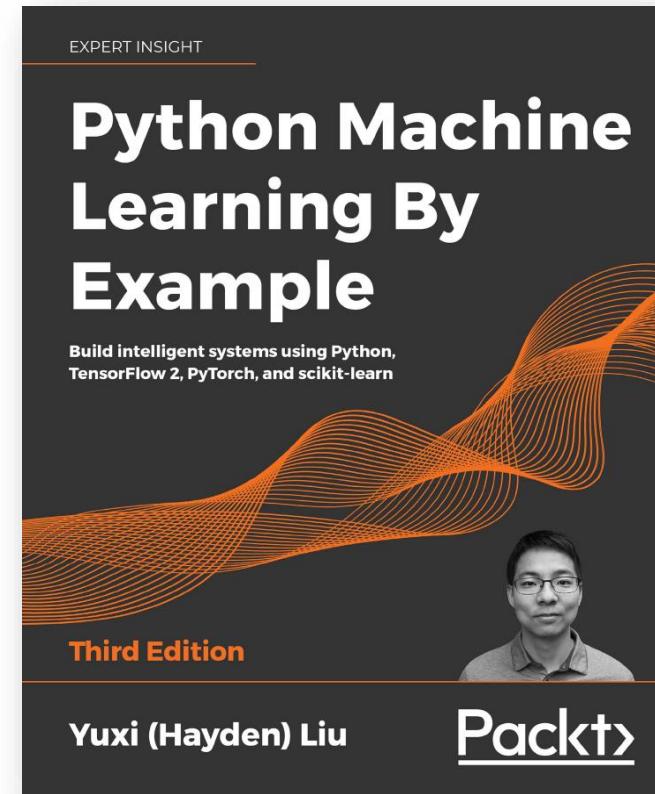
Laboratorio (16 ore)

- Introduzione al Linguaggio Python e alle librerie per il ML
- Definizione di pipeline di ML: Applicazioni pratiche

Testi



Chip Huyen Designing Machine
Learning Systems



Yuxi Liu Python Machine
Learning by Example

Modaltà di esame



**Prova scritta /
Prova in Itinere**



Progetto



Prova orale

Interazione con i Docenti



Giuseppe Polese
gpolese@unisa.it



Loredana Caruccio
lcaruccio@unisa.it

Utilizzo della piattaforma e-learning per comunicazioni e condivisione del materiale didattico

- elearning.informatica.unisa.it/el-platform/

ML Overview

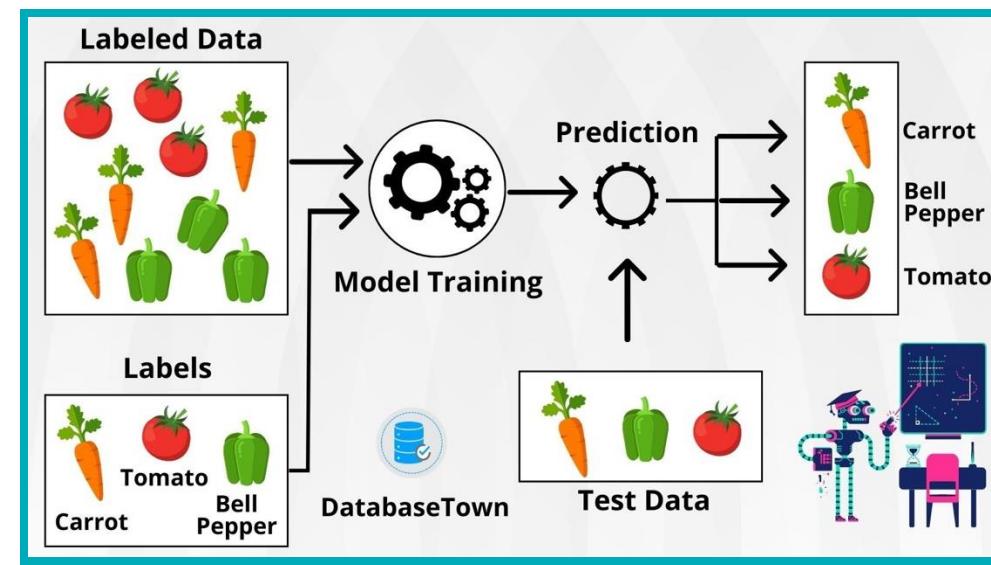
2

Categorie di approcci di ML

- Le tecniche di machine learning possono essere categorizzate in base a diversi criteri:
 - Livello di supervisione umana (apprendimento **supervisionato, non supervisionato e per rinforzo**)
 - Capacità di apprendere incrementalmente (**batch** e **online** learning)
 - In base a come effettuano le predizioni (**instance-based** e **model-based** learning).
- Queste categorie **non sono mutualmente esclusive.**

Livello di supervisione

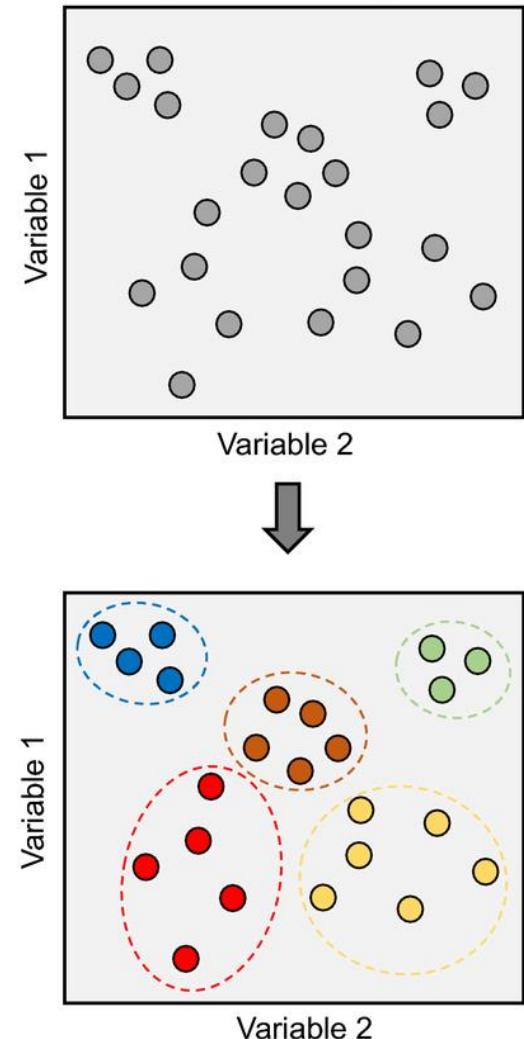
- Nell'apprendimento **supervisionato** i dati di training sono etichettati con la soluzione corretta.



- La **classificazione** è un tipico task di apprendimento supervisionato che consiste nell'assegnamento di una label. Un altro task supervisionato è la **regressione**, che invece restituisce un valore numerico.

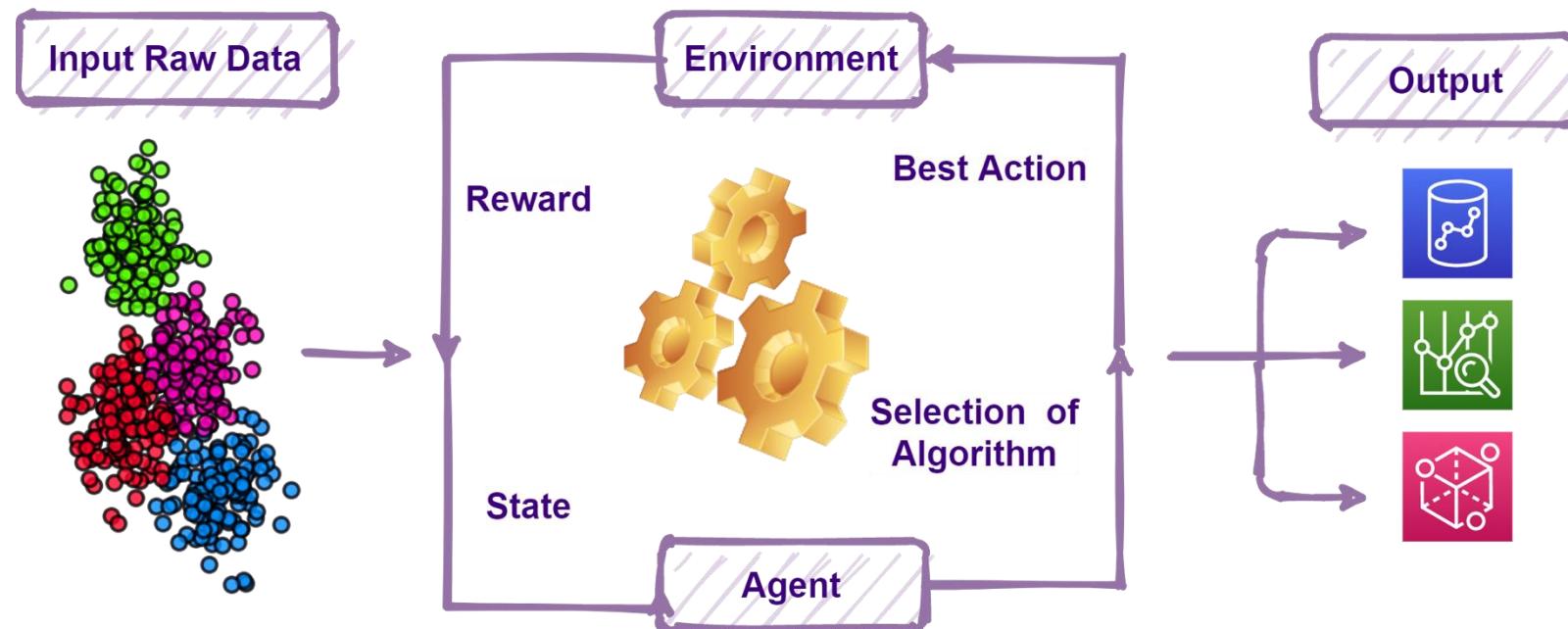
Livello di supervisione

- Nell'apprendimento **non supervisionato** i dati di training non sono etichettati e il sistema deve apprendere da solo.
- Tipici task non supervisionati sono il **clustering**, la **riduzione della dimensionalità** e l'estrazione di **association rule**.
- Le tecniche **semisupervisionate** usano invece dati parzialmente etichettati (tipicamente pochi con label e molti senza).



Categorie di approcci di ML

- Nell'**apprendimento per rinforzo** (reinforcement learning) un agente osserva l'ambiente ed esegue delle azioni, ricevendo **ricompense e penalità**.
- L'agente apprende la miglior strategia per **massimizzare le ricompense**.



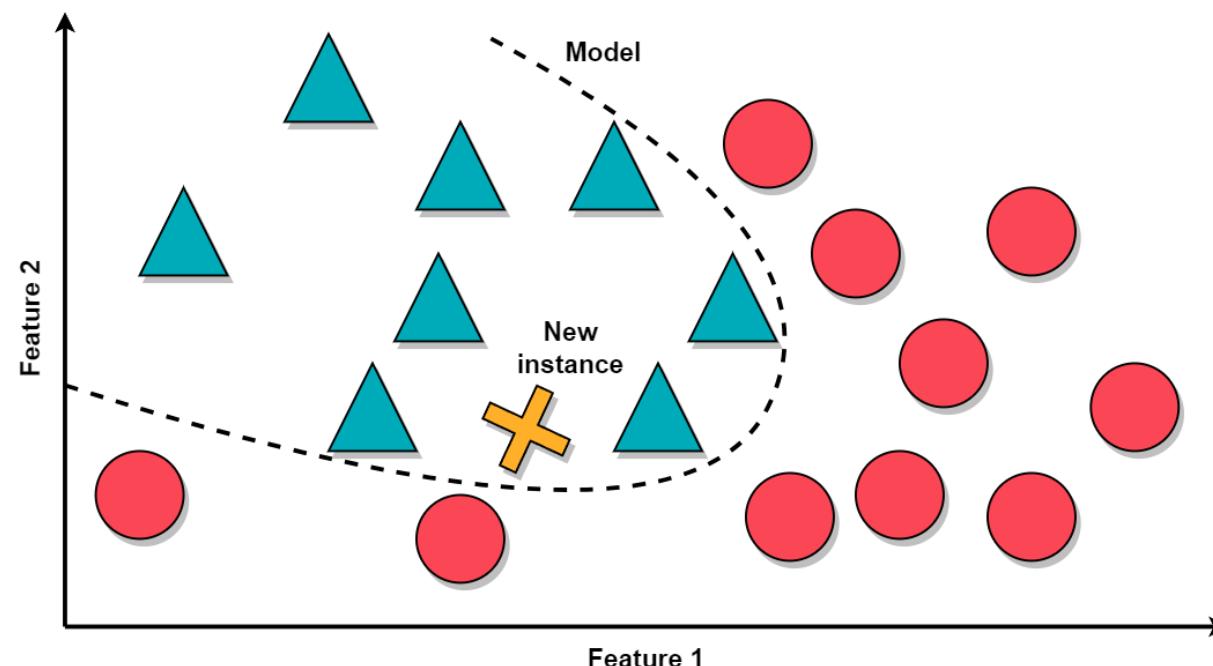
Tipologia di apprendimento

- Nell'apprendimento **batch/offline** il sistema viene addestrato usando tutti i dati a disposizione. Il modello ottenuto viene messo in funzione senza essere più addestrato.
- Nell'**online learning** il sistema apprende incrementalmente, ricevendo nuove istanze singolarmente o in gruppo.

Online Learning	Offline Learning
Learning is done incrementally on the dataset	Learning is done once on the dataset
Model is adaptable to different data	Model is not adaptable
Complex to develop	Less complex to develop
Requires more computations	Fewer computations required
Less storage space is required	Requires storage to store the entire dataset
It can be expensive as it is resource intensive	Less expensive

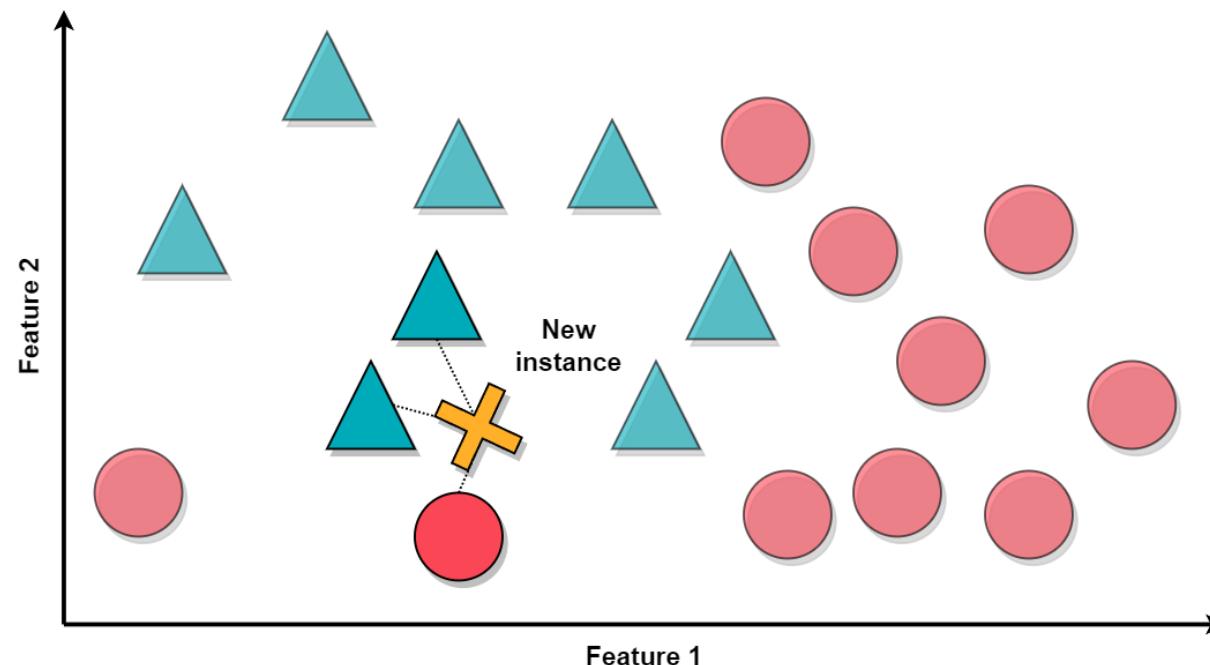
Tipologia di generalizzazione

- Nell'apprendimento **model-based** il sistema costruisce un modello sulla base dei dati di training e lo usa per fare predizioni. Il sistema va quindi ad apprendere le caratteristiche del problema, individuando una funzione adatta per la predizione.



Tipologia di generalizzazione

- Nell'apprendimento **instance-based** il sistema effettua predizioni valutando la similarità delle nuove istanze con quelle di training.
 - Ad esempio, se una nuova e-mail è molto simile ad una conosciuta e classificata come spam, allora il sistema può segnalare la nuova istanza.



Esempio

- **Esempio:** Supponiamo che si voglia scoprire se i soldi rendono le persone felici. Scarichiamo i dati del **Better Life Index** dal sito dell'OCSE (Organizzazione per la Cooperazione e lo Sviluppo Economico) e le statistiche sul **Prodotto Interno Lordo** (PIL) pro capite dal sito web del Fondo Monetario Internazionale (FMI). Uniamo le due tabelle e le ordiniamo in base al PIL

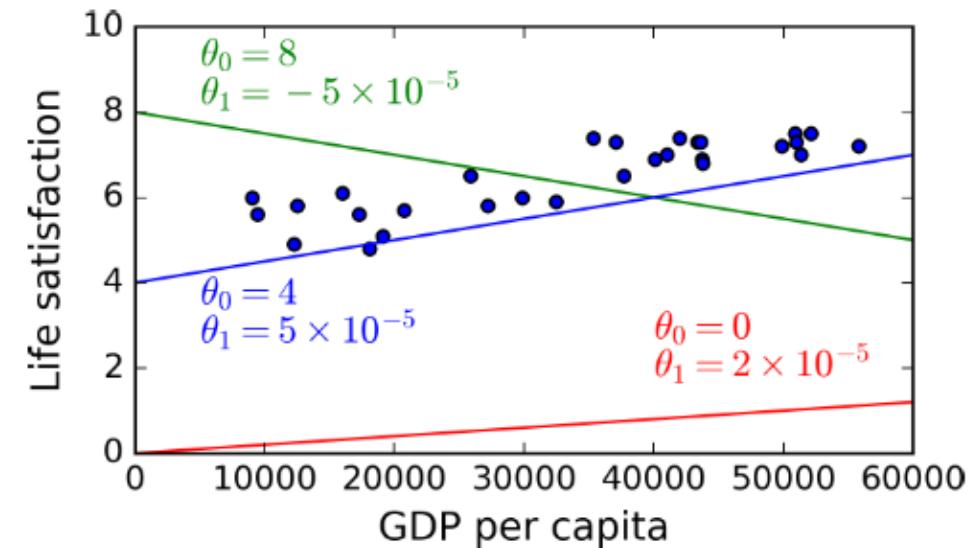
Paese	PIL (USD)	Livello di soddisfazione
Ungheria	12,240	4.9
Corea	27,195	5.8
Francia	37,675	6.5
Australia	50,962	7.3
Stati Uniti	55,805	7.2

Esempio (2)

- Decidiamo di usare un approccio **model-based**. Sembra che il livello di soddisfazione nella vita aumenti più o meno linearmente all'aumentare del PIL pro capite del paese.
- Quindi, un modello lineare basato su un solo attributo (PIL pro capite) potrebbe essere adatto per modellare la soddisfazione nella vita. Questo passaggio è chiamato **selezione del modello**:

$$\text{life_satisfaction} = \theta_0 + \theta_1 \times \text{GDP_per_capita}$$

- Modificando i parametri θ_0 and θ_1 , il modello può rappresentare qualsiasi funzione lineare



Esempio (3)

- Per utilizzare il modello è necessario definire i parametri θ_0 e θ_1 in modo da ottenere le prestazioni ottimali del modello.
- Una **funzione di utilità** (o di **fitness**) può essere utilizzata per misurare quanto il modello sia buono, o equivalentemente, una **funzione di costo** può misurare quanto sia cattivo.
- Per problemi di **regressione lineare**, viene comunemente utilizzata una funzione di costo che misura la distanza tra le previsioni del modello e gli esempi di addestramento, con l'obiettivo di minimizzare questa distanza.
- L'algoritmo di regressione lineare può essere utilizzato per trovare i parametri che consentono al modello lineare di adattarsi meglio agli esempi di addestramento. Questo processo è chiamato **addestramento del modello**.

Esempio (4)

- Nel nostro esempio l'algoritmo individua come ottimali i seguenti parametri:
 $\theta_0 = 4.85$ and $\theta_1 = 4.91 \times 10^{-5}$.
- Supponiamo di voler utilizzare il modello per effettuare delle previsioni, ad esempio per i cittadini ciprioti, il cui PIL pro capite è di **\$22,587**, ma per i quali i dati dell'OCSE non forniscono la risposta.
- Possiamo applicare il modello appreso e stabilire che la soddisfazione nella vita è circa:

$$4.85 + 22,587 \times 4.91 \times 10^{-5} = 5.96$$

Esempio (5)

- Utilizzando invece un algoritmo di apprendimento **instance-based**, avremmo scoperto che la Slovenia ha un PIL pro capite più vicino a quello di Cipro (\$20,732), e poiché i dati dell'OCSE ci dicono che la soddisfazione nella vita dei sloveni è 5.7, avremmo stimato una soddisfazione nella vita di **5.7** per Cipro.
- Se consideriamo **i successivi due paesi più vicini**, Portogallo e Spagna, con soddisfazioni nella vita di 5.1 e 6.5 rispettivamente, facendo la media di questi tre valori otteniamo **5.77**, che è più vicino alla previsione **model-based**.
- Questo semplice algoritmo è chiamato **k-Nearest Neighbors regression** (in questo esempio, $k = 3$).

Esempio (6)

- Se il modello selezionato **non garantisce buone prestazioni** di predizione, possiamo fare una delle seguenti cose:
 - utilizzare **più attributi** (tasso di occupazione, salute, inquinamento dell'aria, ecc.)
 - ottenere **più dati** di training o dati di **migliore qualità**
 - selezionare un **modello più potente** (ad esempio, un modello di **Regressione Polinomiale**).

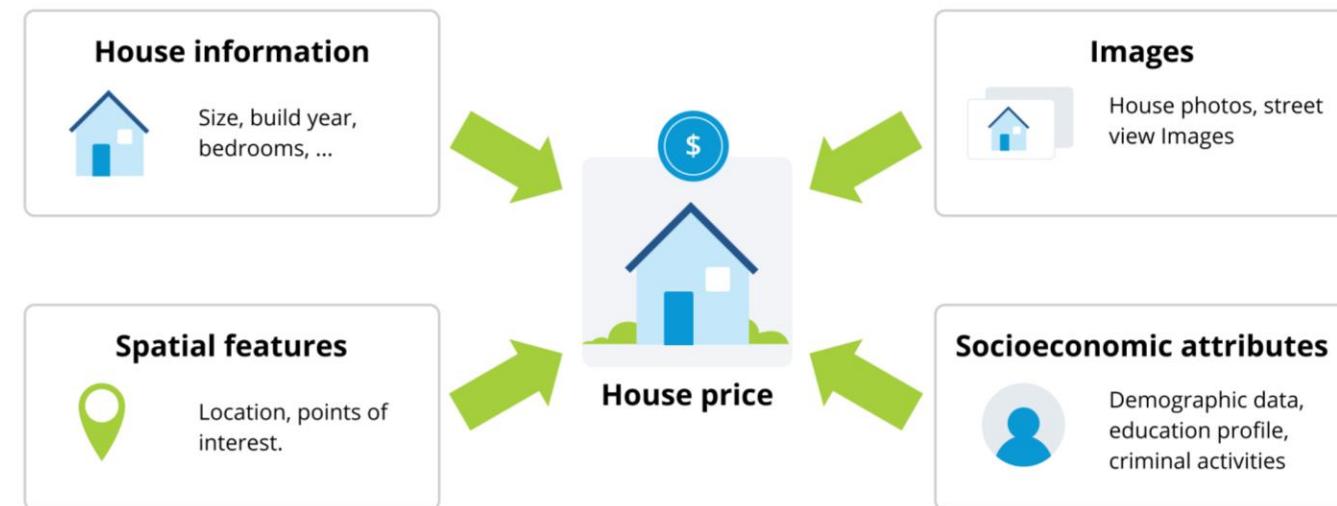
Quando usare il ML?

- Il machine learning ha dimostrato di essere uno strumento potente, adatto ad una vasta gamma di problemi. Tuttavia, **non è uno strumento magico che risolve tutto.**
- Inoltre, anche se è in grado di affrontare un problema, **non è detto che sia la soluzione ottimale.**
- Per capire cosa può fare il ML, esaminiamo cosa fanno le soluzioni che lo sfruttano.

Il machine learning è un approccio per **apprendere pattern complessi** da **dati esistenti** e usare questi modelli per fare **previsioni** su **dati non visti**

Quando usare il ML?

- **Apprendere:** il sistema è in grado di **apprendere dai dati**.
- Deve esserci qualcosa da cui apprendere, ed in particolare i sistemi di ML hanno bisogno di **dati**
- Esempio: per predire il prezzo di affitto di una casa servono dati con **caratteristiche rilevanti** (metratura, numero di stanze, servizi, rating) e il **prezzo**.



Quando usare il ML?

- **Pattern Complessi:** esistono pattern complessi da apprendere.
- Il ML ha senso solo se ci sono **pattern utili**
 - Se questi esistono o no non è ovvio. Inoltre, anche se esistono può darsi che il dataset o il modello utilizzato non sono sufficienti per catturarli.
- Il problema deve inoltre essere **complesso**, non serve investire risorse in sistemi complessi come quelli di ML.
- Ciò che è complesso per l'uomo **può non esserlo per la macchina** e viceversa
 - Elevare un numero alla potenza di 10 è facile per una macchina
 - Decidere se c'è un gatto in una foto è facile per l'uomo.

Quando usare il ML?

- **Dati esistenti:** c'è un dataset o è possibile reperirlo
- Il successo dei modelli di machine learning è fortemente dipendente dalla **qualità e quantità** dei dati disponibili.
- La disponibilità di molti dati permette di ottenere un modello più preciso ed in grado di **generalizzare** meglio
 - Inoltre, si ha una visione più ampia del problema ed aumentano le possibilità di scoprire **pattern nascosti**
- Lo **zero-shot learning** permette di fare buone predizioni senza essere stato addestrato su dati relativi a quel compito, ma richiede comunque un addestramento su dati legati ad un task correlato

Quando usare il ML?

- È anche possibile lanciare un sistema di ML senza dati. Nel **continual learning**, i modelli di ML possono essere impiegati senza essere stati sufficientemente addestrati ed impareranno dai dati in arrivo.
 - Tuttavia, fornire agli utenti modelli non sufficientemente addestrati **comporta rischi** come una cattiva esperienza del cliente.
- Senza dati e senza continual learning, le aziende seguono un approccio "**Fake-it-till-you-make-it**": lanciano un prodotto che fornisce predizioni fatte da umani e i dati raccolti vengono poi usati per addestrare un modello

Quando usare il ML?

- **Predizioni:** è un problema predittivo
- I modelli di ML producono predizioni, quindi sono adatti solo a problemi che richiedono **risposte predittive**. In particolare, essi sono molto utili quando si può beneficiare dall'avere tante stime rapidamente, **anche se approssimate**
- Con il miglioramento dei modelli di ML, molti task vengono riformulati per renderli "predittivi". I problemi ad **alta intensità computazionale** sono un esempio comune
 - Ci si chiede "Come sarebbe il risultato di questo processo?"
 - Si ottengono approssimazioni spesso sufficientemente accurate dell'output esatto

Quando usare il ML?

- **Dati non visti:** i dati su cui fare le predizioni condividono pattern con quelli di training
- Un modello per stabilire se un'applicazione verrà scaricata a Natale nel **2020** non sarà accurato se è addestrato su dati del **2008**, quando l'app più popolare era Koi Pond
- In pratica, I dati di training e quelli non visti devono provenire da **distribuzioni simili**



Quando usare il ML?

- Come stabilire se i dati non visti seguono lo stesso pattern di quelli di training?
 - Non possiamo, ma si possono fare **assunzioni**.
 - Ad esempio, potremmo assumere che il comportamento dei consumatori in futuro possa essere influenzato da schemi simili a quelli del passato.
- Se l'assunzione risulta corretta, il modello avrà una base solida su cui fare previsioni.
- Se l'assunzione non regge, il modello si comporterà male, evidenziando la necessità di **nuovi dati di training** che riflettano i nuovi pattern emergenti.

Quando usare il ML?

- Altre caratteristiche dei problemi per cui il ML è utile:
 - **Ripetitività:** quando un task è ripetitivo, è più facile individuare pattern.
 - **Basso costo per predizioni errate:** è improbabile che, per un task significativo, un modello effettui sempre predizioni corrette. Usare questo approccio è accettabile se le conseguenze di un errore non sono catastrofiche.
 - **È su larga scala:** il ML richiede investimenti non banali, quindi è ragionevole impiegarlo se ci si aspetta di dover fare molte predizioni.
 - **I pattern cambiano costantemente nel tempo:** le culture, le tecnologie e i gusti cambiano. Soluzioni **hardcoded** come regole scritte a mano diventano presto obsolete.

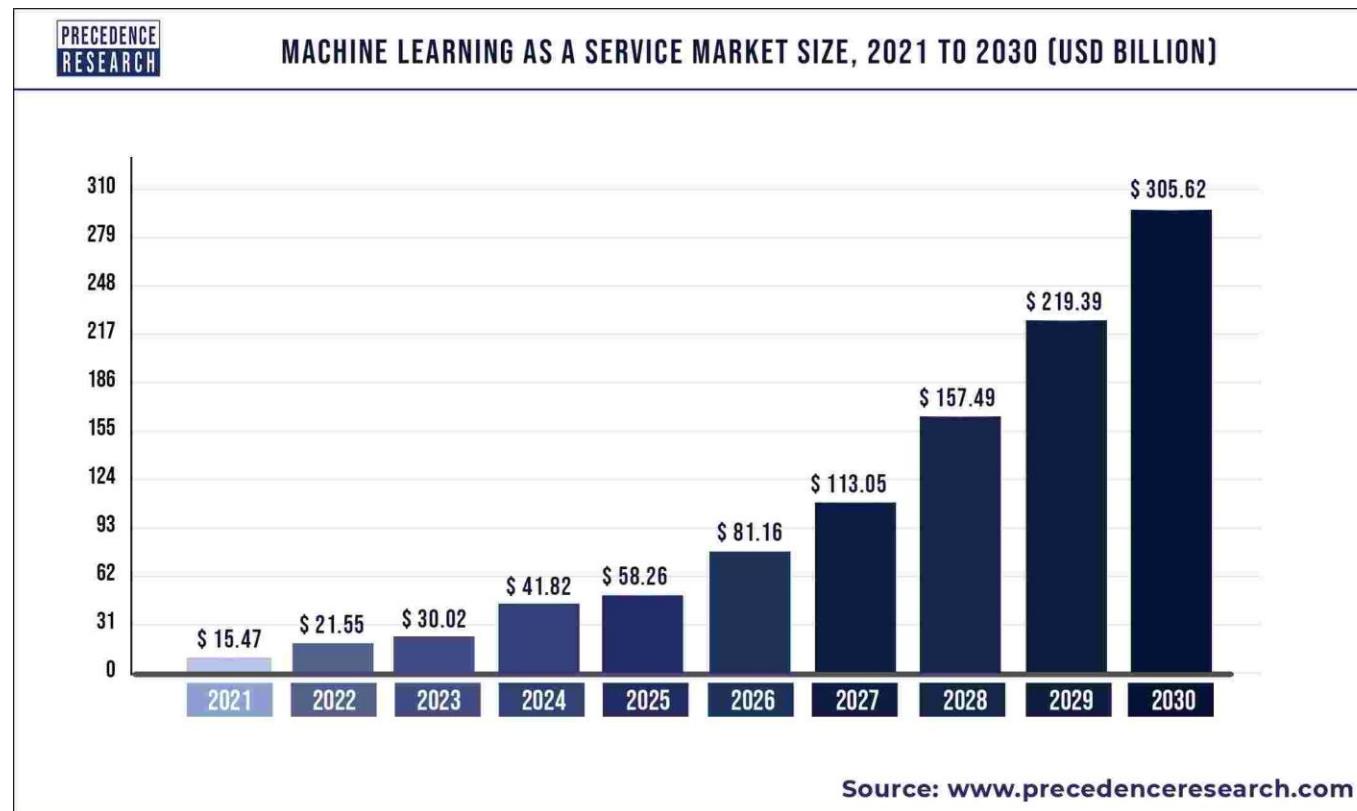
Quando **non** usare il ML?

- La maggior parte degli algoritmi di ML moderni non dovrebbe essere usata in presenza di una di queste condizioni:
 - **Non è etico**
 - Esistono **delle soluzioni più semplici** che ottengono già buoni risultati
 - Non è **economicamente conveniente**
- Tuttavia, anche se il ML non può risolvere il problema, è possibile scomporre quest'ultimo in sottoproblemi e risolverne qualcuno con esso.
- Inoltre, è possibile che una tecnica non conveniente con le attuali tecnologie lo diventi nel futuro, quindi non si dovrebbe abbandonarla del tutto.

Applicazioni del ML

3

Applicazioni del ML



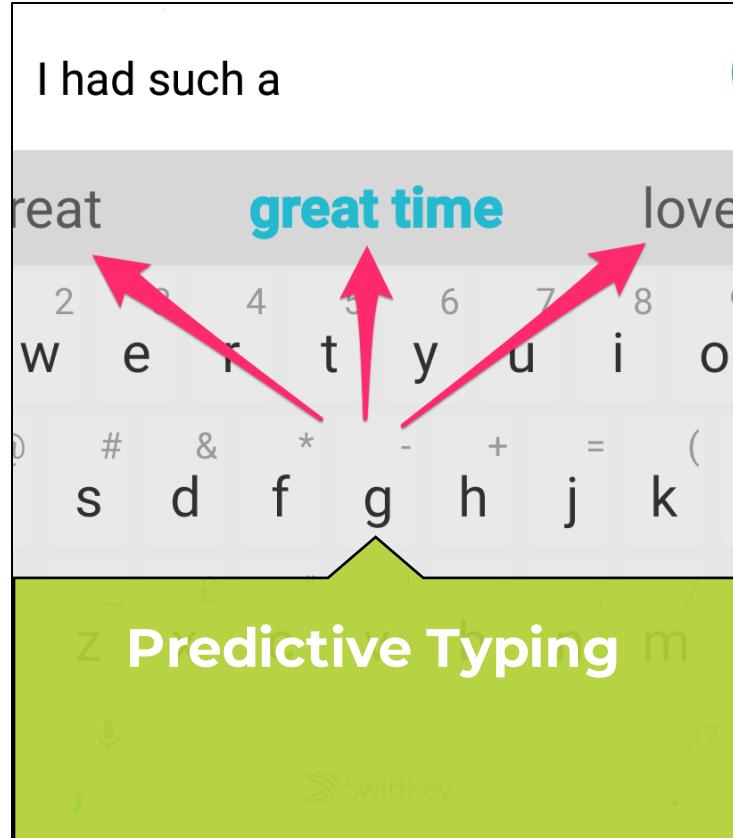
- Dal 2010 ad oggi, il ML si è diffuso in maniera capillare e le stime sono in continua crescita
- Di seguito vedremo alcune applicazioni in ambito **consumer**

Applicazioni del ML



- I **motori di ricerca** utilizzano algoritmi di ML per analizzare una vasta quantità di dati web e restituire **risultati pertinenti**. Questi algoritmi migliorano costantemente grazie ai **feedback** e **all'analisi delle interazioni** con i risultati di ricerca.
- I sistemi di raccomandazione usano il ML per predire le **preferenze individuali** e **suggerire contenuti** rilevanti (es. Netflix). Questo approccio personalizzato aumenta **l'engagement** e influenza le decisioni di consumo.

Applicazioni del ML



- Il **predictive typing** è uno strumento che migliora notevolmente la velocità e l'accuratezza della digitazione su tastiere virtuali.
- Utilizzo di modelli di linguaggio basati su dati storici per **anticipare e suggerire** le parole che un utente sta digitando.
- Man mano che l'utente digita, il sistema genera possibili continuazioni **basate sul contesto e sulle probabilità**.

Applicazioni del ML



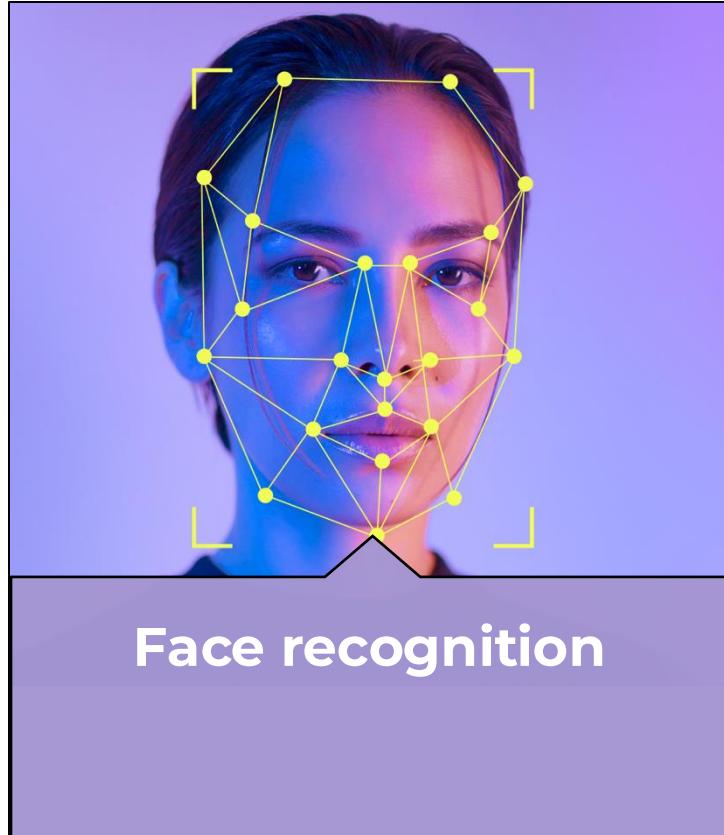
- La **traduzione automatica** è una delle applicazioni più emblematiche dell'apprendimento automatico nel mondo reale.
- Gli algoritmi di ML **analizzano pattern e strutture linguistiche** nei dati di input e producono traduzioni coerenti e precise.
- Piattaforme come Google Translate sfruttano questa tecnologia per **abbattere le barriere linguistiche e facilitare la comunicazione** globale.

Applicazioni del ML



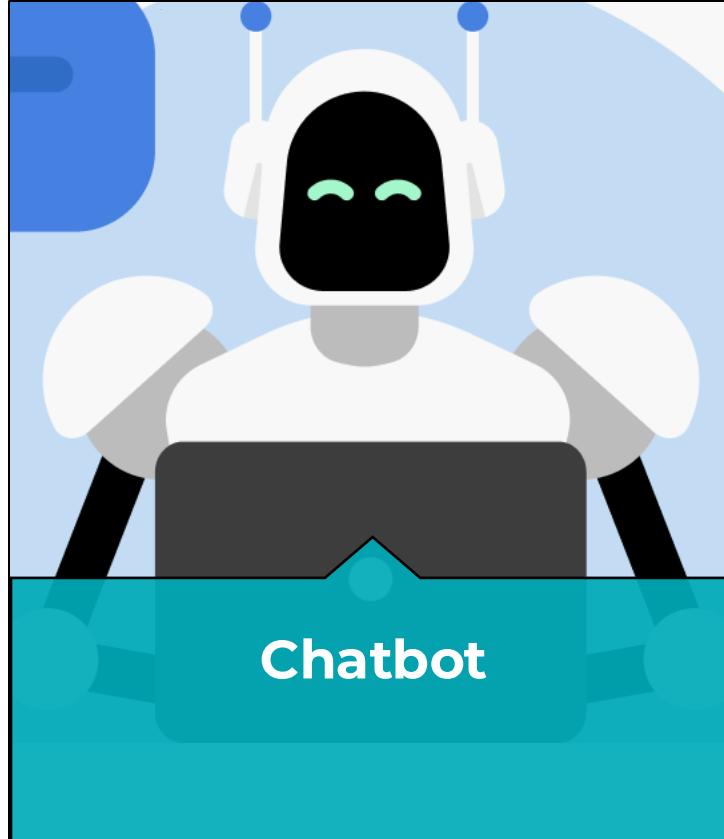
- I dispositivi nelle **smart home** raccolgono dati sulle abitudini e le preferenze degli utenti che vengono analizzati da algoritmi di ML per **adattare l'ambiente domestico**.
- Ad esempio, un termostato intelligente può imparare le preferenze di temperatura e regolare autonomamente il riscaldamento.
- Questa tecnologia migliora l'efficienza energetica, la comodità e la sicurezza delle case.

Applicazioni del ML



- Gli algoritmi di riconoscimento facciale utilizzano modelli di apprendimento automatico per **identificare e verificare** le persone in base alle loro caratteristiche facciali.
- Gli algoritmi di ML identificano estraendo e codificando queste caratteristiche in modo da creare una "**firma** **unica**" per ogni volto.
- Questa tecnologia è ampiamente utilizzata per il riconoscimento degli utenti su **dispositivi mobili** e per il controllo di accesso a edifici e **luoghi sensibili**.

Applicazioni del ML



- I chatbot **sono assistenti virtuali** che possono interagire con gli utenti attraverso chat testuali o vocali.
- Questi agenti sono in grado di **comprendere il linguaggio naturale** e rispondere alle domande degli utenti.
- Essi svolgono compiti come la fornitura di informazioni, la risoluzione di problemi o l'assistenza all'acquisto.

Applicazioni del ML

- Anche se il numero delle applicazioni consumer sta crescendo, la maggior parte dei casi d'uso del ML riguarda l'ambito **aziendale**
- In questi scenari, le prestazioni dei modelli rappresentano un requisito **nettamente più importante**
 - Aumentare l'accuratezza del riconoscimento vocale dal 95% al 95.5% non porta ad effetti rilevanti per gli utenti
 - Migliorare l'efficienza dell'allocazione delle risorse dello 0.1% in un'azienda come Google può far risparmiare **milioni di dollari**

Applicazioni del ML

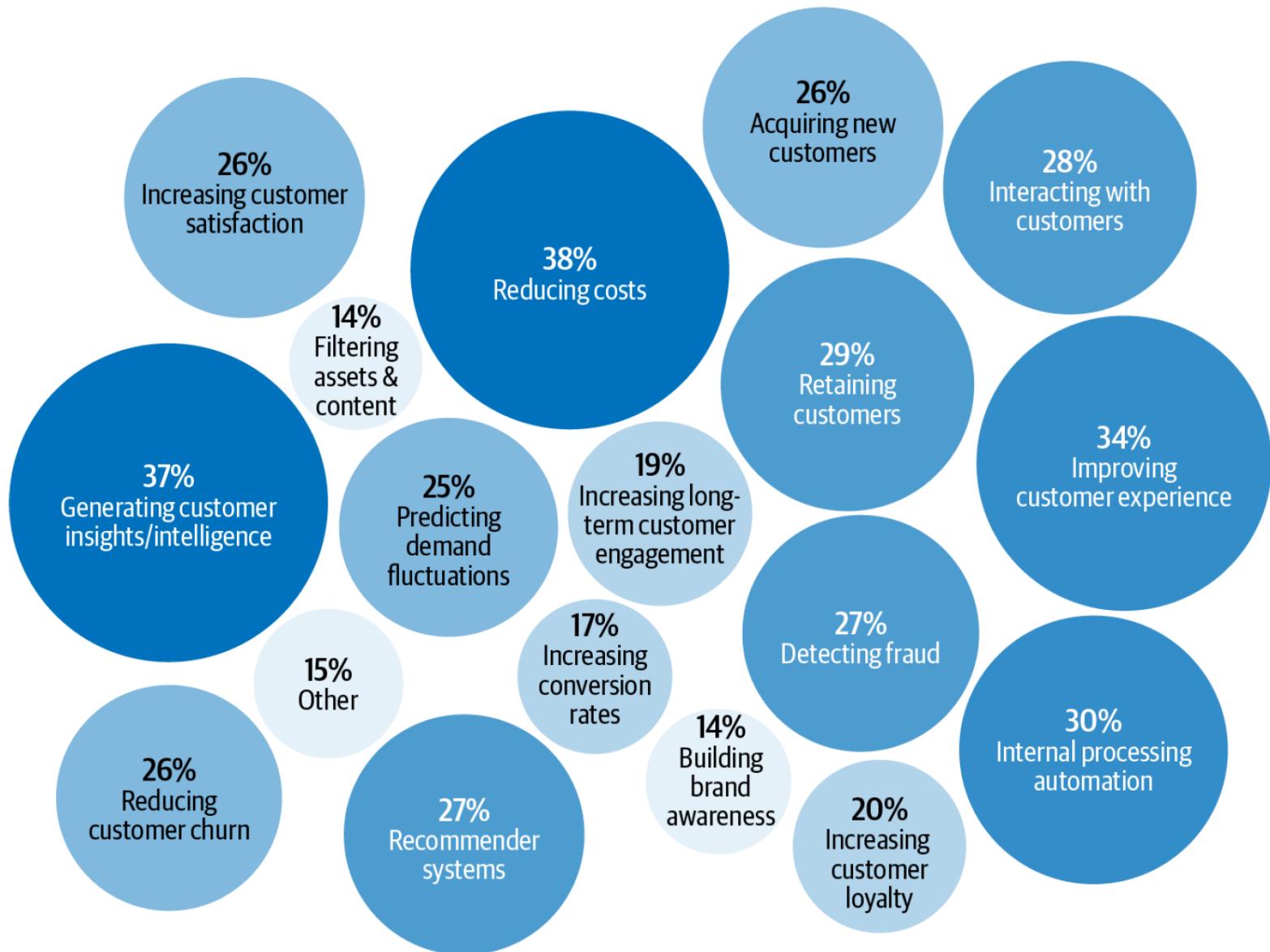
- Il **rilevamento di frodi** è una delle applicazioni più antiche del ML. I modelli vengono impiegati per identificare e prevenire attività fraudolente, proteggendo le aziende e i clienti dalle perdite finanziarie e dalle violazioni della sicurezza.
- Decidere il prezzo di un prodotto è una delle decisioni più difficili. Il ML può essere impiegato per individuare strategie di **price optimization**. Analizzando le dinamiche del mercato e il comportamento dei clienti, i modelli possono stabilire il prezzo ottimale per massimizzare una funzione obiettivo (es. profitto)

Applicazioni del ML

- Le aziende usano spesso modelli per la predizione della **domanda di un prodotto**, in modo da preparare budget, strategia di vendita e allocare le giuste risorse
- Inoltre, il ML può essere usato per ridurre il costo di acquisizione di **nuovi clienti**, suggerendo pubblicità mirate, sconti ed identificando potenziali nuovi utenti
- Altro task di interesse è la **churn prediction**, ovvero stimare quando un cliente attuale sta per abbandonare i prodotti dell'azienda, che può così provare a "riconquistare" il cliente.

Applicazioni del ML

- Per mantenere soddisfatti i consumatori è importante rispondere tempestivamente ai loro problemi. La **classificazione automatica dei ticket di supporto** è un modo per velocizzare il processo di assistenza.
- Un'altra applicazione popolare è il **brand monitoring**. È importante per le aziende monitorare la percezione dei loro brand attraverso tecniche di sentiment analysis.
- Infine, il ML è impiegato intensivamente in ambito **medico** per rilevare malattie, supportare le diagnosi e migliorare i piani di trattamento.



Uso del Machine learning nelle imprese (2020)

ML: Ricerca vs Produzione

ML: Ricerca vs Produzione

- L'impiego del machine learning nell'ambito produttivo è molto diverso da quello di ricerca

	Ricerca	Produzione
Requisiti	Performance del modello rispetto allo stato dell'arte	Molteplici stakeholder con requisiti differenti
Priorità	Training rapido ed alto throughput	Inferenza rapida, Bassa Latenza
Dati	Statici	Costantemente in evoluzione
Fairness	Spesso non considerata	Deve essere considerata
Interpretabilità	Spesso non considerata	Deve essere considerata

ML: Ricerca vs Produzione

- Spesso nel contesto di ricerca si ha un solo obiettivo, tipicamente le **performance** del modello
 - A volte, per migliorare leggermente l'accuratezza si propongono modelli **tropo complessi** da poter usare concretamente.
- In un contesto aziendale, ci sono **molteplici stakeholder** che stabiliscono requisiti differenti che a volte sono in **conflitto tra loro**.
- Addestrare un modello avente tutte le caratteristiche desiderate diventa molto complesso.

Esempio

- **Esempio:** Si consideri una applicazione che consiglia ristoranti agli utenti. L'applicazione genera introiti addebitando ai ristoranti una commissione del 10% su ogni ordine. Ciò significa che gli ordini costosi forniscono all'app più denaro rispetto agli ordini economici. Il progetto coinvolge diverse persone.
- **Il team responsabile dello sviluppo** vuole un modello che consigli ristoranti da cui gli utenti molto probabilmente ordineranno e credono di poterlo fare utilizzando un modello **più complesso addestrato con molti dati**
- **Il team responsabile delle vendite** desidera un modello che consigli ristoranti più costosi per avere **più commissioni**.

Esempio

- **I responsabili del prodotto** sono consapevoli che ogni aumento della latenza porta a una diminuzione degli ordini e desiderano un modello che possa restituire i ristoranti consigliati **in meno di 100 millisecondi**.
- **Il team responsabile della piattaforma** è concentrato sulla scalabilità del sistema, e quindi desidera **ritardare gli aggiornamenti** del modello per dare priorità al miglioramento della piattaforma.
- Il **manager** vuole massimizzare i margini di profitto, magari ridimensionando il team di sviluppo dopo aver ottenuto il modello, **sacrificando la manutenzione e gli aggiornamenti futuri**.

ML: Ricerca vs Produzione

- Quando si sviluppa un progetto di ML, un errore comune è concentrarsi prevalentemente/esclusivamente sullo sviluppo del modello **a discapito della fase di deployment e manutenzione**
- Durante lo sviluppo, il principale **collo di bottiglia** è il training, ma dopo la messa in funzione il problema diventa **l'inferenza**
- In ambito aziendale, è quest'ultima ad essere importante. Diversi studi hanno evidenziato i **danni economici** dovuti ad una latenza eccessiva

ML: Ricerca vs Produzione

- Nel 2017, uno studio di Akamai^[1] ha stabilito che un delay di 100ms può danneggiare il **tasso di conversione** del 7%
 - Tasso di conversione = percentuale di utenti o visitatori di un sito web che compiono un'azione desiderata, come effettuare un acquisto o iscriversi ad una newsletter
- Nel 2016 Google ha scoperto che più della metà degli utenti mobile **abbandona una pagina** se questa impiega più di 3 secondi per caricare^[2]
- Oggi gli utenti sono ancora **meno pazienti!**

[1] <https://www.akamai.com/newsroom/press-release/akamai-releases-spring-2017-state-of-online-retail-performance-report>

[2] <https://www.marketingdive.com/news/google-53-of-mobile-users-abandon-sites-that-take-over-3-seconds-to-load/426070/>

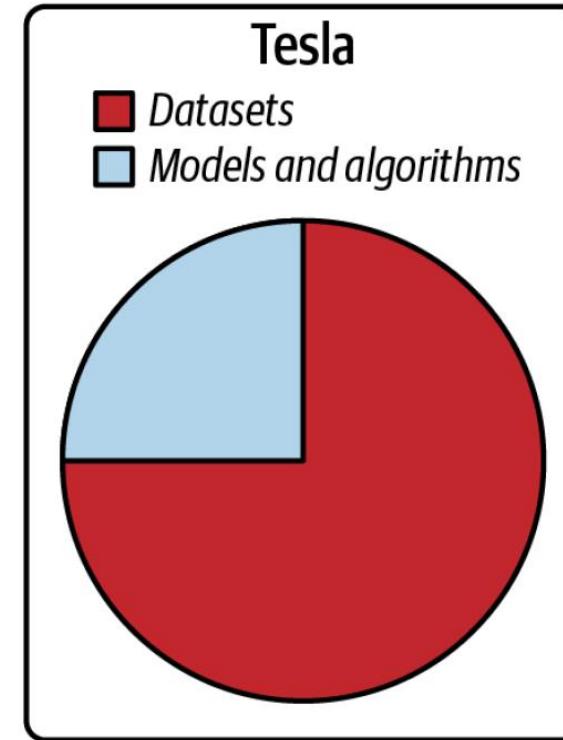
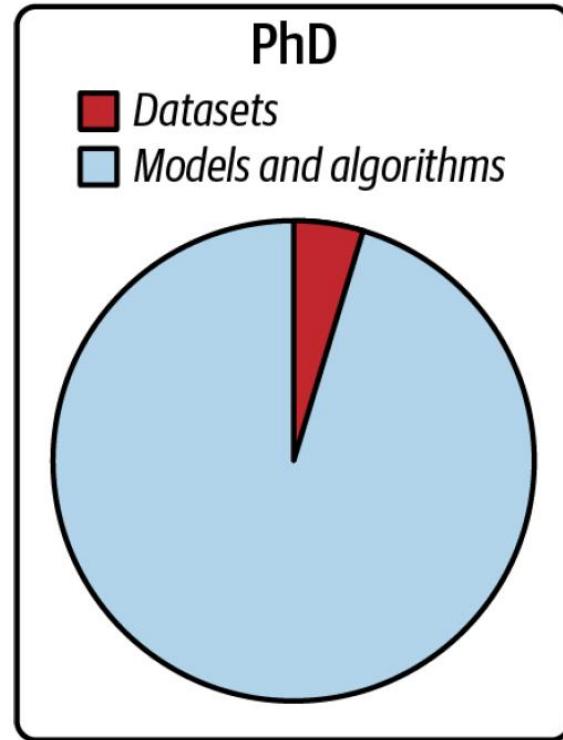
ML: Ricerca vs Produzione

- Un'altra differenza importante riguarda i dati. In ambito di ricerca si ha spesso a che fare con dataset **conosciuti, puliti e statici**. Questi dati sono già stati largamente utilizzati e le loro **caratteristiche sono note**.
- In ambito aziendale i dati, se disponibili, sono di qualità inferiore. Contengono **rumore, valori mancanti e cambiano costantemente nel tempo**. Inoltre, contengono probabilmente **bias ed errori**



ML: Ricerca vs Produzione

Amount of sleep lost over...



ML: Ricerca vs Produzione

- Quando si impiega un modello nel mondo reale è importante considerare la sua imparzialità (**fairness**).
- Molte persone sono state già "vittime" di modelli che presentano **bias**. Gli algoritmi di ML non predicono il futuro, ma **codificano il passato**, apprendendo anche i pregiudizi presenti nei dati.
- Se un operatore umano potrebbe esprimere giudizi su pochi individui alla volta, un algoritmo di ML può formularne su **milioni di persone** in una frazione di secondo. Questo può danneggiare in particolare i membri di gruppi di minoranza.

ML: Ricerca vs Produzione

- Uno studio condotto da ricercatori dell'Università di Berkeley¹ nel 2019 ha rilevato che i prestatori di denaro avevano respinto complessivamente **1,3 milioni di richieste** di prestiti da parte di candidati afroamericani e latini tra il 2008 e il 2015.
- Successivamente, quando i ricercatori hanno analizzato nuovamente le richieste di prestito utilizzando solo **i dati relativi al reddito** e ai **punteggi di credito**, senza considerare l'appartenenza razziale, molte di queste richieste di mutuo sono state accettate.
- È importante che i modelli evitino di apprendere questi pregiudizi.

[1] <https://www.cbsnews.com/news/mortgage-discrimination-black-and-latino-paying-millions-more-in-interest-study-shows/>

ML: Ricerca vs Produzione

- Mentre la maggior parte di noi si sente a proprio agio nell'usare un microonde senza capirne il funzionamento, molti non la pensano ancora allo stesso modo sull'IA, soprattutto se questa prende **decisioni importanti sulla loro vita.**
- Poiché la maggior parte della ricerca sul ML è ancora valutata sulle prestazioni del modello, i ricercatori non sono incentivati a considerare questo aspetto.
- In ambito industriale, invece, **l'interpretabilità** è spesso un requisito. Essa è importante sia per i manager aziendali che per gli utenti finali, che hanno bisogno di capire il motivo per cui viene presa una decisione, in modo da potersi fidare di un modello e identificare potenziali **bias**.

Obiettivi e Requisiti

Obiettivi

- Lo sviluppo di un sistema di ML deve essere **motivato da obiettivi precisi**. In genere le aziende non si preoccupano delle metriche di apprendimento come l'accuratezza a meno che non possano influenzare le **metriche aziendali**.
- L'obiettivo finale di qualsiasi progetto all'interno di un'azienda è quello di aumentare i profitti in due modi:
 - **Direttamente** (es. aumentando le vendite, riducendo i costi)
 - **Indirettamente** (es. maggiore soddisfazione del cliente, aumento del tempo trascorso su un sito web)
- Affinché un progetto abbia successo è fondamentale **collegare le prestazioni del sistema di ML alle prestazioni aziendali complessive**.

Obiettivi

- Molte aziende creano le **proprie metriche** per collegare le metriche aziendali alle metriche di ML
- Ad esempio, Netflix misura le prestazioni del proprio sistema di raccomandazione utilizzando il "**take-rate**"¹:
 - il numero di riproduzioni di qualità (consigliate dal sistema) diviso per il numero di raccomandazioni visualizzate da un utente.
 - Più alto è il take-rate, migliore è il sistema
- Comprendere l'effetto del ML sugli obiettivi aziendali può essere difficile e richiede spesso molti esperimenti

[1] <https://netflixtechblog.com/artwork-personalization-c589f074ad76>

Requisiti

- Non si può dire di aver sviluppato con successo un sistema di ML senza sapere quali sono i **requisiti da soddisfare**
- Questi variano in base allo scenario, ma di base la maggior parte dei sistemi dovrebbe avere le caratteristiche di **affidabilità, scalabilità, manutenibilità e adattabilità.**



Affidabilità

- Il sistema deve continuare a svolgere la funzione corretta **al livello di prestazioni desiderato** anche in presenza di avversità (guasti hardware o software e errori umani).
- Con i sistemi software tradizionali, spesso si riceve un avviso, come un crash di sistema o un errore di runtime.
- Tuttavia, i sistemi ML possono **fallire silenziosamente**. L'utente non sa che il sistema si è guastato e potrebbe continuare a usarlo come se fosse funzionante.
 - Ad esempio, se si utilizza Google Translate per tradurre in una lingua che non si conosce, è difficile capire se la traduzione è sbagliata o meno.

Scalabilità

- Un sistema di ML può crescere in vari modi:
 - **Complessità:** può essere necessario rimpiazzare un modello più semplice con uno più complesso per adattarsi all'evoluzione dei dati
 - **Volume del traffico:** con il crescere del numero di utenti, il numero di predizioni da effettuare può raggiungere livelli elevati
 - **Numero di modelli:** all'inizio si potrebbe iniziare con un solo modello ideato per uno specifico use case. Successivamente, con l'aggiunta di nuove funzionalità potrebbe essere necessario l'impiego di modelli aggiuntivi

Scalabilità

- Indipendentemente dalla tipologia di crescita, è necessario **gestirla in maniera ragionevole**
- Quando si parla di scalabilità il pensiero comune è quello inerente l'allocazione delle risorse (**up-scaling e down-scaling**).
 - Ad esempio, se il sistema durante un picco di carico richiede 100 GPU ma normalmente richiede 10 GPU, è costoso lasciare sempre 100 GPU attive. Il sistema deve adattarsi al carico corrente.
 - Un errore in questa procedura può causare costosi periodi di **downtime**¹

[1] <https://www.businessinsider.com/amazon-prime-day-website-issues-cost-it-millions-in-lost-sales-2018-7>

Scalabilità

- La scalabilità riguarda anche la **gestione** dei modelli. Avere 100 modelli da supervisionare è molto diverso da averne uno solo.
- Con un solo modello si può attuare un controllo **manuale** del comportamento, e si può intervenire senza problemi per aggiornarlo con nuovi dati.
- Con molti modelli, è necessario **automatizzare** sia il processo di monitoraggio sia quello di riaddestramento per garantire che i modelli rimangano aggiornati e performanti nel tempo.

Manutenibilità e Adattabilità

- Nello sviluppo di un sistema ML sono coinvolti molteplici soggetti con competenze e background molto diversi tra loro.
- È importante strutturare il sistema in maniera tale che **ognuno possa contribuire**. Il codice, i dati e i modelli devono essere organizzati per poter lavorarci anche in assenza degli autori originali.
- Inoltre, per adattarsi all'evoluzione dei dati e dei requisiti, il sistema deve essere capace di:
 - Scoprire nuovi aspetti per migliorare le performance
 - Permettere aggiornamenti senza interrompere il servizio

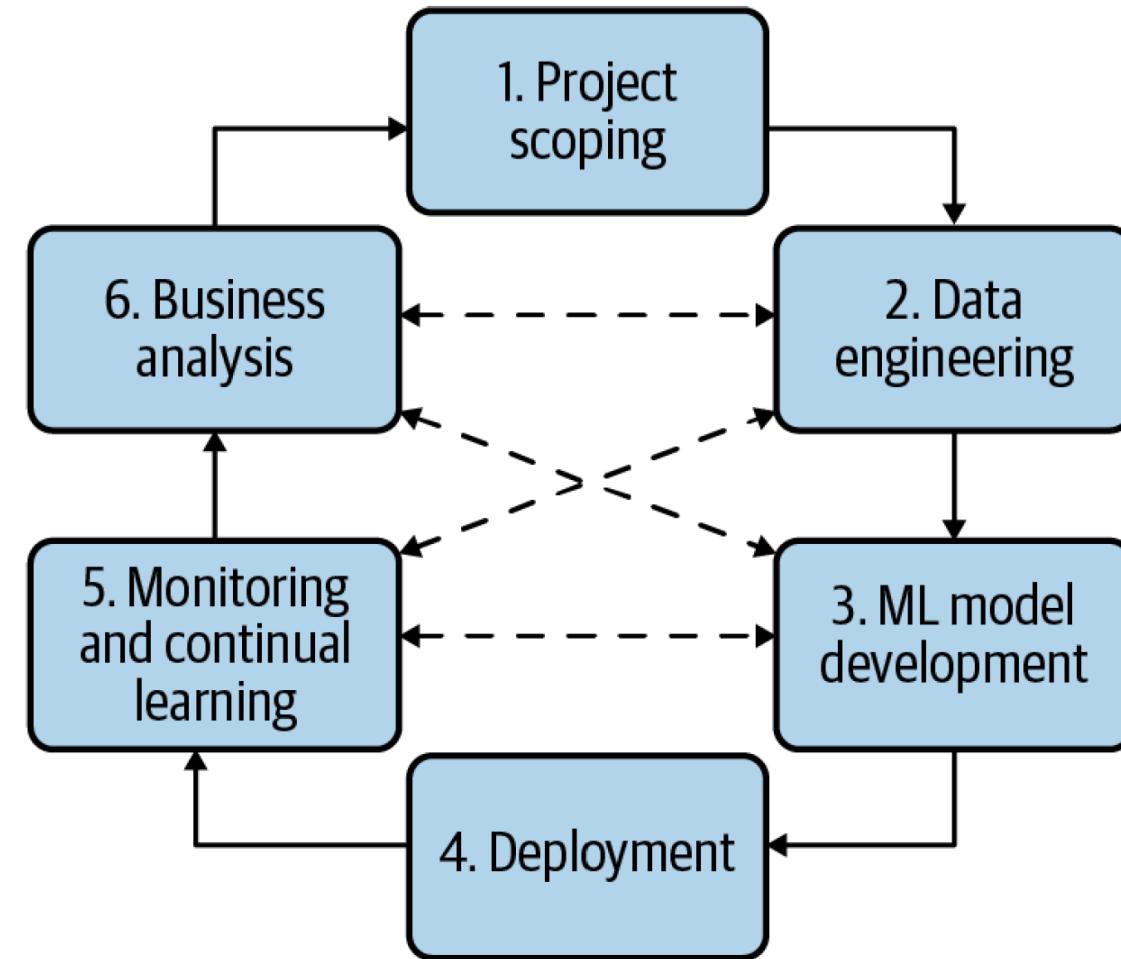
Sviluppo e definizione del problema

Sviluppo di un sistema di ML

- Sviluppare un sistema di ML è un processo **iterativo** e, nella maggior parte dei casi, **senza fine**
 - Una volta messo in produzione, il modello deve essere **costantemente monitorato e aggiornato**
- Contrariamente a quanto si potrebbe pensare, il processo di sviluppo non è lineare e **contiene spesso dei cicli**, con molti passaggi di revisione tra le varie fasi



Sviluppo di un sistema di ML

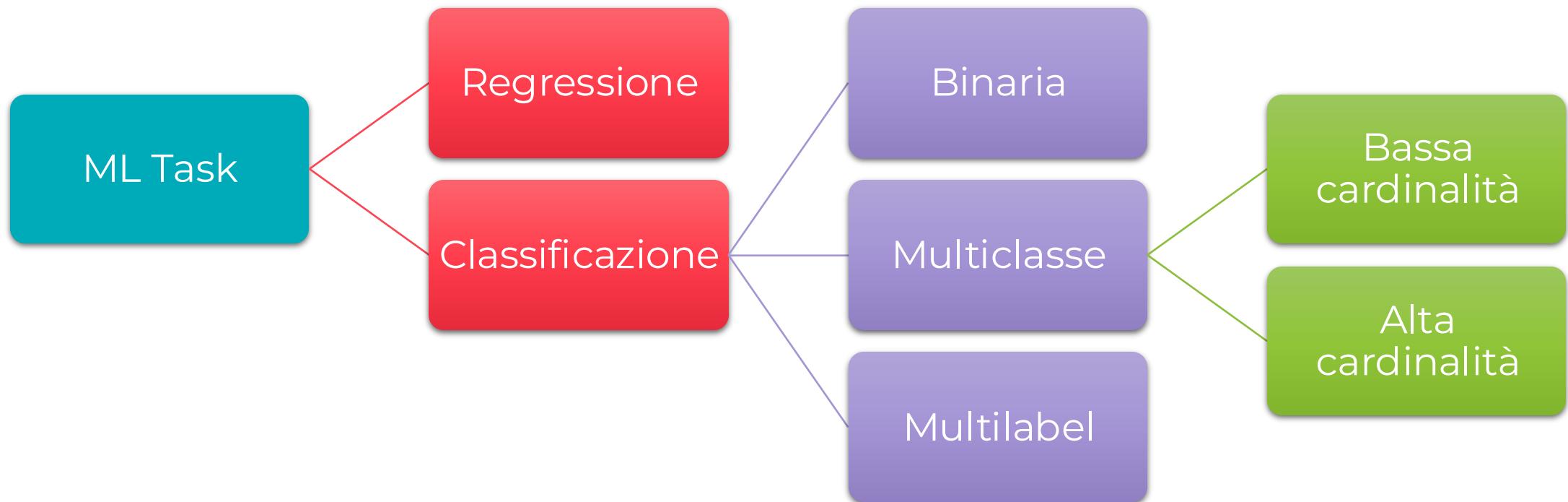


Sviluppo di un sistema di ML

- **Project scoping:** Identificazione di obiettivi e vincoli, coinvolgendo gli stakeholder e stimando le risorse necessarie.
- **Data Engineering:** Elaborazione dei dati, gestione di diverse fonti e formati, e creazione dell'insieme di dati di addestramento.
- **Model development:** Estrazione delle feature significative dai dati di addestramento e sviluppo del modello.
- **Deployment:** Messa in funzione del sistema.
- **Monitoring:** Monitoraggio delle performance del modello e fasi di aggiornamento per adattarlo ai cambiamenti.
- **Business analysis:** Valutazione delle performance rispetto agli obiettivi aziendali.

Definizione del problema

- Formulare il tipo di problema è una fase molto importante. Esistono diverse tipologie di task di ML, identificati dal **tipo di output del modello**. La distinzione più generale è tra problemi di **classificazione e regressione**



Definizione del problema

- Per i problemi di classificazione, meno classi ci sono e più semplice è il problema.
- Con solo due classi si parla di **classificazione binaria**. Alcuni esempi sono: classificare se un commento è fake, se una scansione polmonare mostra segni di cancro, se una transazione è fraudolenta.
- Con più di due classi il problema diventa **multiclasse**. Questi sono più complessi non solo per risolvere il problema, ma anche per valutare le prestazioni del modello.

Definizione del problema

- Quando il numero di classi è elevato (diagnosi di malattie, classificazione di prodotti) il task è ad **alta cardinalità**. I problemi ad alta cardinalità possono essere molto sfidanti.
- La prima difficoltà è nella **raccolta dei dati**. In genere, i modelli di ML hanno bisogno di almeno 100 esempi per ogni classe per imparare a classificare quella classe.
- Quindi, con 1.000 classi, servono già almeno 100.000 esempi. La raccolta dei dati può essere particolarmente difficile **per le classi rare**, che sono molto probabili in questo tipo di task.

Definizione del problema

- Con i task ad alta cardinalità di potrebbe impiegare la **classificazione gerarchica**
- Inizialmente, si impiega un classificatore che distingue le istanze in diversi **macrogruppi**. Successivamente, ulteriori classificatori vengono utilizzati per **classificare i sottogruppi**
- Esempio: classificazione di prodotti
 - Un primo classificatore distingue i prodotti in macrogruppi generali (elettronica, abbigliamento, alimentari). Per ogni macrogruppo, vengono usati modelli per sottocategorie specifiche (es. smartphone, laptop, televisori per l'elettronica).

Definizione del problema

- Sia nella classificazione binaria che in quella multclasse, ogni esempio appartiene **esattamente ad una classe**.
- Quando invece un esempio può appartenere a più classi il task è **multilabel**
 - Ad esempio, un articolo può trattare più di un argomento
- Ci sono due modi per trattare questi problemi. Il primo consiste nel trattarli come problemi multclasse:
 - Nella classificazione multclasse la label in output è del tipo **[0, 0, 1, 0]**
 - Nel caso multilabel l'output può sarà del tipo **[1, 0, 1, 0]**.

Definizione del problema

- Alternativamente, si può trasformare il task in un **problema di classificazione binaria**
 - Si addestra un **modello binario per ogni singola classe**, e ognuno di questi viene usato per fare una predizione su un nuovo esempio
- In generale i problemi multilabel sono i più complessi da trattare:
 - Ad esempio, gli annotatori potrebbero avere opinioni diverse su quali classi appartiene un esempio, creando difficoltà nel risolvere le loro divergenze.
 - Inoltre, possono sorgere delle problematiche sul decidere le classi da restituire in output

Definizione del problema

- Infatti, il numero di classi rende complesso estrarre previsioni dalle probabilità dei modelli. Nella classificazione multiclasse, dove un esempio può appartenere solo a una categoria, è possibile scegliere semplicemente la categoria **con la probabilità più alta**.
 - Con output **[0, 0.22, 0.78, 0]** scegliamo la terza classe
- Nella classificazione multilabel, poiché **non si sa a quante categorie può appartenere un esempio**, può essere necessario scegliere più di una categoria in base alle probabilità più alte.
 - Con output **[0, 0.35, 0.4, 0.25]** bisogna stabilire se l'esempio appartiene a una, due o tre classi

Definizione del problema

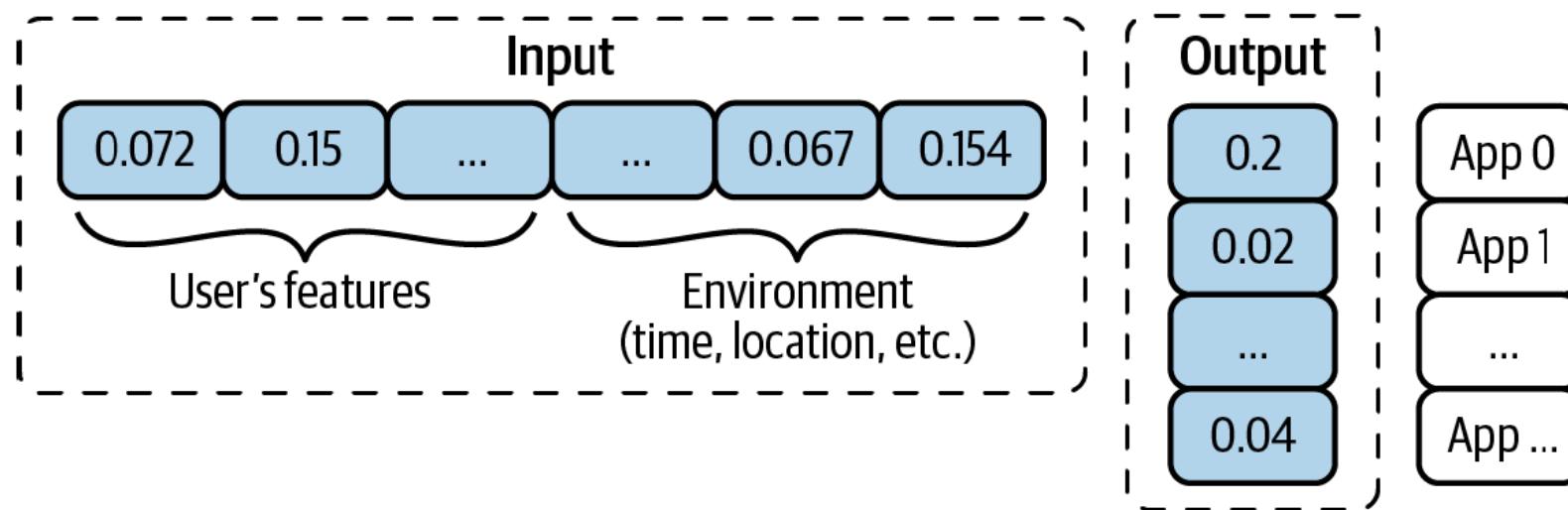
- Cambiare il modo in cui è formulato un problema può rendere il task **più semplice o più complesso**
- Ad esempio, si consideri il task di predire qual è la prossima app che l'utente ha intenzione di aprire tra le N disponibili
- Una soluzione naïve consiste nel formulare il task come un problema di classificazione multclasse
 - Si considerano feature relative all'utente e all'ambiente come input, e l'output consiste in una distribuzione di probabilità per ogni app installata

Definizione del problema

- Questa formulazione richiede, ogni volta che è necessario, una sola predizione che restituisce in output un vettore di N elementi

Problem: predict the app a user will most likely open next

Classification



Definizione del problema

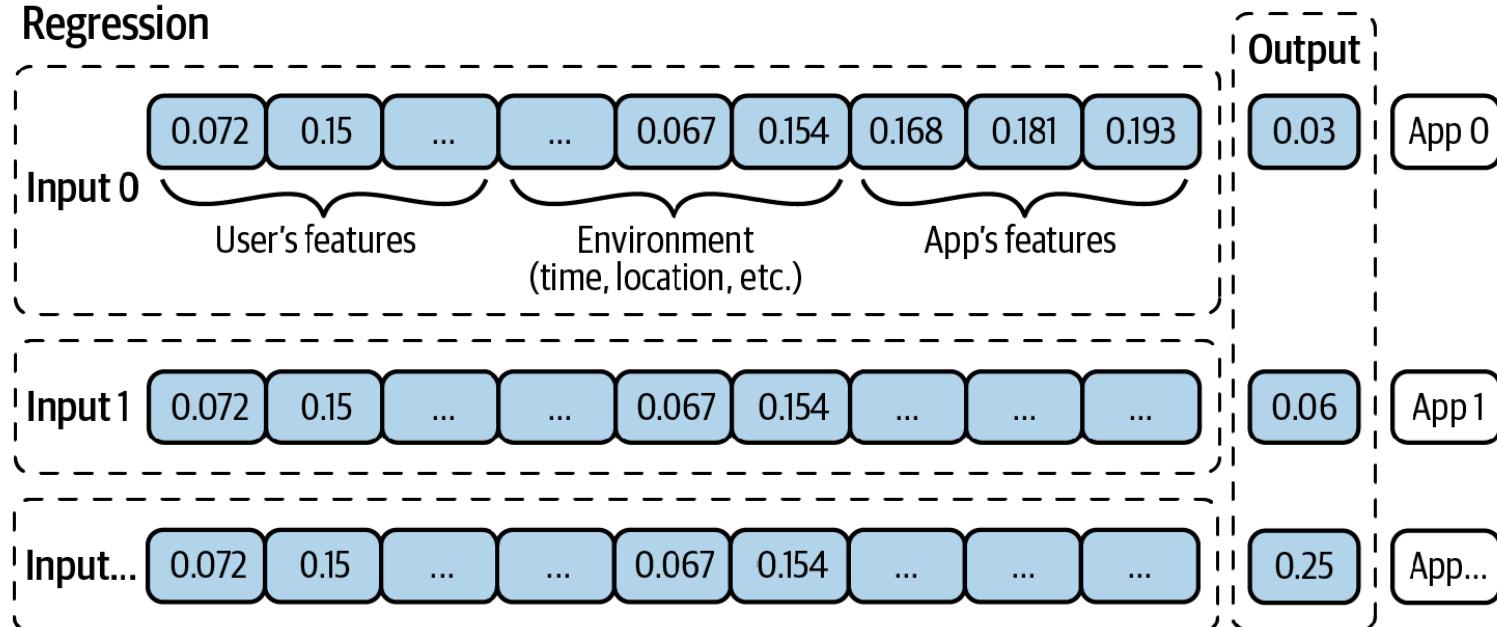
- Tuttavia, quest'approccio non è adeguato
 - Ogni volta che una nuova app viene installata, è necessario riaddestrare il modello
- Un approccio migliore consiste nel formulare il task come un problema di **regressione**
 - Il modello prende in input le feature relative all'utente, all'ambiente e all'applicazione
 - L'output è un singolo valore compreso tra 0 e 1. Più il valore è elevato, più è probabile che l'utente abbia intenzione di aprirla

Definizione del problema

- In questo caso, sono necessarie ogni volta N predizioni. Quando si aggiunge una nuova app, però, basta usare il nuovo input con le feature dell'app senza necessità di aggiornare il modello.

Problem: predict the app a user will most likely open next

Regression



Funzioni obiettivo

- Un altro aspetto fondamentale da stabilire è la **funzione obiettivo (loss function)**
 - Questa è necessaria per **guidare l'addestramento** del modello
- Di solito prende il nome di loss function perché l'obiettivo dell'addestramento è **minimizzare la loss** causata da predizioni sbagliate
- Nel ML supervisionato, la loss viene calcolata confrontando l'output del modello con le label effettive usando misure come **il Root Mean Square Error (RMSE)** o la **cross-entropy**

Funzioni obiettivo

- **Esempio:** si consideri un task di classificazione per stabilire il topic di un articolo tra tecnologia, intrattenimento, finanza e politica. Supponiamo che per un articolo di politica (quindi con label $[0, 0, 0, 1]$) il modello restituisca $[0.45, 0.2, 0.02, 0.33]$.
- **Calcoliamo la cross entropy:**

$$L(\hat{y}, y) = - \sum_{i=1}^n (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$$

Funzioni obiettivo

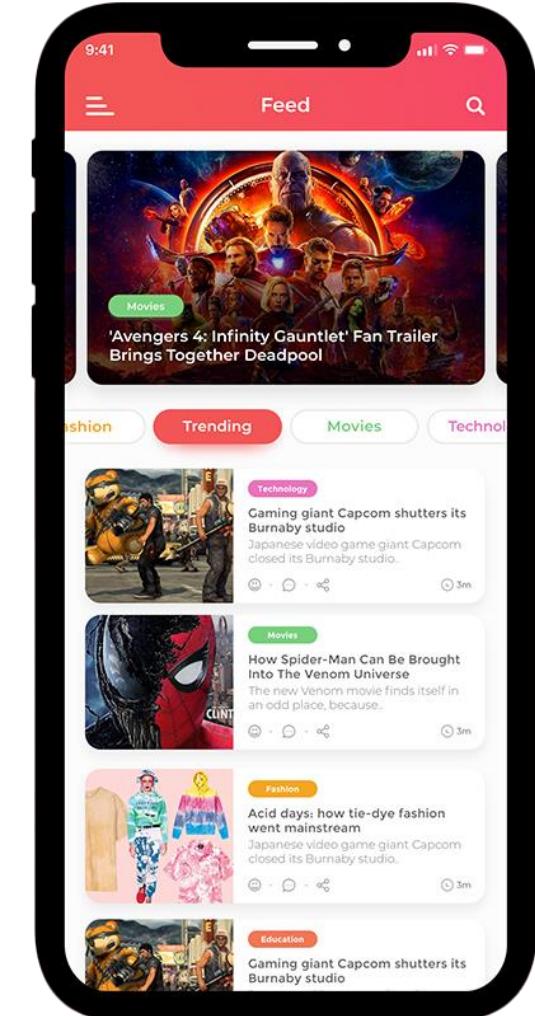
- Per le prime tre classi (dove $y = 0$):
 - Per la prima classe: $0 * \log(0,45) + (1 - 0) * \log(1 - 0,45) = 0 + 0,79851 = \mathbf{0,79851}$
 - Per la seconda classe: $0 * \log(0,2) + (1 - 0) * \log(1 - 0,2) = 0 + 0,22314 = \mathbf{0,22314}$
 - Per la terza classe: $0 * \log(0,02) + (1 - 0) * \log(1 - 0,02) = 0 + 0,01980 = \mathbf{0,01980}$
- Per la quarta classe (dove $y = 1$):
 - Per la quarta classe: $1 * \log(0,33) + (1 - 1) * \log(1 - 0,33) = -0,41504 + 0 = \mathbf{-0,41504}$
- Si ottiene la Cross Entropy sommando i valori ottenuti:
 - $0,79851 + 0,22314 + 0,01980 - 0,41504 \approx \mathbf{0,6264}$

Funzioni obiettivo

- La cross entropy è comunemente utilizzata per la classificazione multclasse
- Per la classificazione binaria si usa spesso la **logistic loss**
- Per i task di regressione le funzioni più usate sono **Root Mean Square Error (RMSE)** e **Mean Absolute Error (MAE)**
- A volte può essere necessario dover minimizzare **diverse funzioni di loss** per risolvere problemi più complessi
- In questi casi la definizione del problema diventa complicata

Funzioni obiettivo

- **Esempio:** Si consideri lo sviluppo di un sistema per classificare gli elementi nei feed di notizie degli utenti per massimizzare il loro coinvolgimento. Vengono identificati tre obiettivi:
 - Filtrare lo spam
 - Filtrare contenuti NSFW (Not Safe For Work)
 - Fare un ranking dei post in base all'interazione



Funzioni obiettivo

- Tuttavia, si scopre che il modello ha imparato a dare priorità ai contenuti estremi. Questo poiché i post estremi tendono a ottenere più interazioni.
- Si deve quindi massimizzare l'interazione degli utenti minimizzando al contempo la diffusione di opinioni estreme e disinformazione. I nuovi obiettivi sono:
 - Filtrare lo spam
 - Filtrare contenuti NSFW (Not Safe For Work)
 - Fare un ranking dei post in base all'interazione
 - **Filtrare la disinformazione**
 - **Classificare i post per qualità**

Funzioni obiettivo

- Due obiettivi sono **in conflitto tra loro**. Se un post è coinvolgente ma di dubbia qualità, dovrebbe essere classificato come alto o basso?
- Ognuno degli obiettivi è rappresentato da una funzione obiettivo. Per **classificare i post per qualità**, vogliamo che la qualità predetta sia il più vicino possibile a quella effettiva.
- Per **classificare i post in base all'interazione**, vogliamo minimizzare la differenza tra il numero di clic previsti per ciascun post e il suo numero effettivo di clic.

Funzioni obiettivo

- Un primo approccio consiste nel **combinare le due funzioni** per ottenerne una sola da usare per l'addestramento:

$$\text{loss} = \alpha \text{ quality_loss} + \beta \text{ engagement_loss}$$

- Si possono testare diverse combinazioni per α e β per trovare quelli che funzionano meglio. Un problema di questo approccio è che se la qualità dei feed di notizie dei tuoi utenti migliora ma il coinvolgimento diminuisce, si potrebbe dover ridurre α e aumentare β
 - È **necessario un riaddestramento** del modello

Funzioni obiettivo

- Un secondo approccio consiste nell'addestrare due modelli:
 - un **quality model** che minimizza la **quality loss**
 - un **engagement model** che minimizza la **engagement loss**
- Gli output dei modelli possono essere poi **combinati** per ottenere il ranking finale
- In generale, in presenza di più obiettivi, è consigliato separarli. Ciò rende **più facile sviluppo e manutenzione**
 - Ad esempio, uno dei modelli potrebbe richiedere aggiornamenti più frequenti degli altri

Mind vs Data

Mind vs Data

- Molte aziende si concentrano sulla gestione e il miglioramento dei dati anziché sull'ottimizzazione degli algoritmi di apprendimento automatico.
- Nonostante il successo dei modelli che utilizzano grandi quantità di dati, c'è scetticismo sull'accento posto sui dati come via da seguire.
- C'è molto dibattito su cosa sia più importante tra l'avere un'elevata quantità e qualità dei dati (**Data**) e il progettare modelli innovativi (**Mind**).
- In teoria si potrebbe cercare di avere entrambe le cose, ma di solito il tempo speso per un aspetto limitano quello disponibile per l'altro.

Mind vs Data

- Alcuni esperti sottolineano che l'intuizione umana, basata su conoscenze e comprensione approfondite, può guidare la progettazione di **algoritmi intelligenti**
 - Questo può facilitare l'apprendimento efficiente da parte dei modelli, migliorando le prestazioni **con meno dati**.
- Altri sostengono anche che usare una massiccia quantità di dati con un algoritmo di apprendimento semplice porta a modelli **incredibilmente scarsi**.

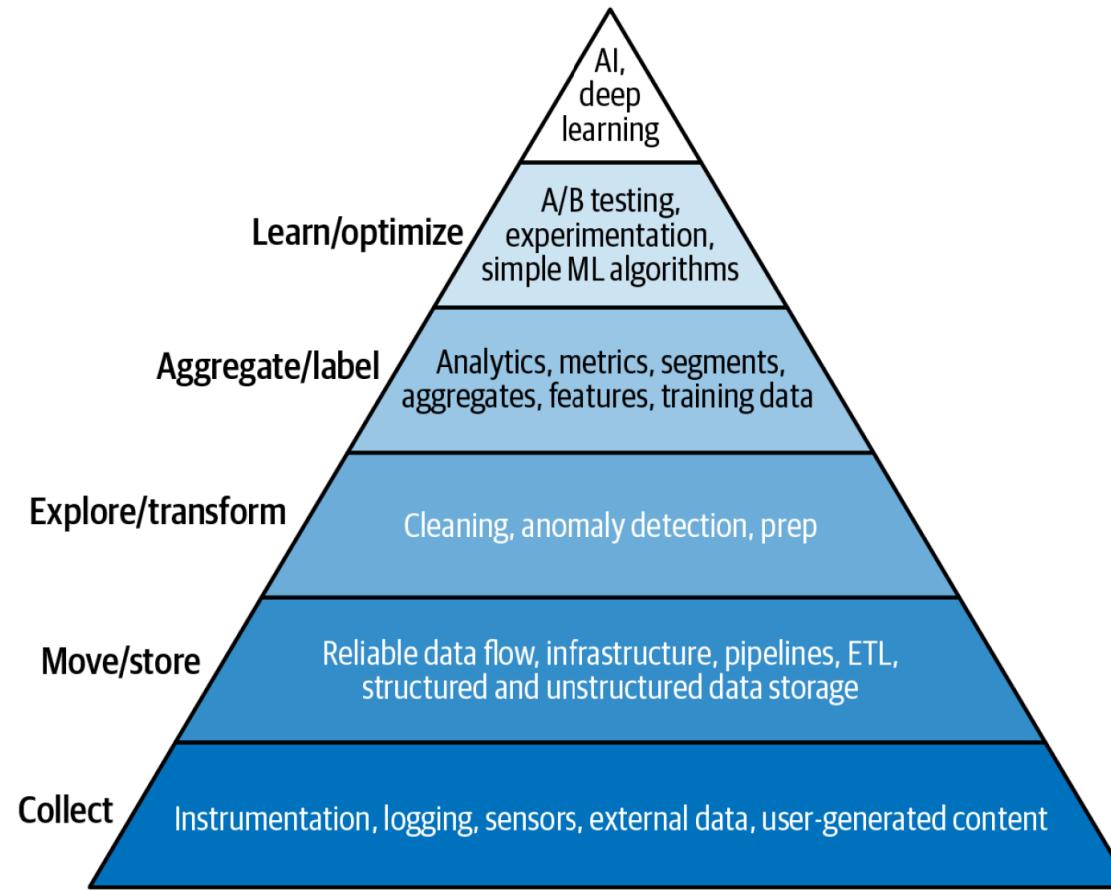
Mind vs Data

- D'altro canto, molti esperti sostengono invece **l'importanza dei dati**, sostenendo che grazie ad essi si ottengono sistemi più efficaci. Un ampio dataset può:
 - aiutare i modelli di ML a cogliere **pattern complessi**
 - aiutare a **mitigare l'impatto degli errori** nei dati, che hanno meno probabilità di influenzare l'apprendimento
- Inoltre, dati di elevata qualità permettono di ottenere modelli più **accurati e robusti**.

Mind vs Data

- Indipendentemente da quale campo dimostrerà di avere ragione alla fine, nessuno può negare che **i dati siano essenziali**, almeno per ora.
- I dati sono alla base della **Data Science**, una disciplina di cui l'apprendimento automatico è parte integrante.
- Pertanto, per migliorare i propri prodotti o processi, è sempre necessario partire dai dati, assicurandosi che siano soddisfacenti sia in termini di qualità che di quantità.

Mind vs Data



Gerarchia delle necessità nel campo della Data Science, da cui è evidente l'importanza dei dati

Mind vs Data

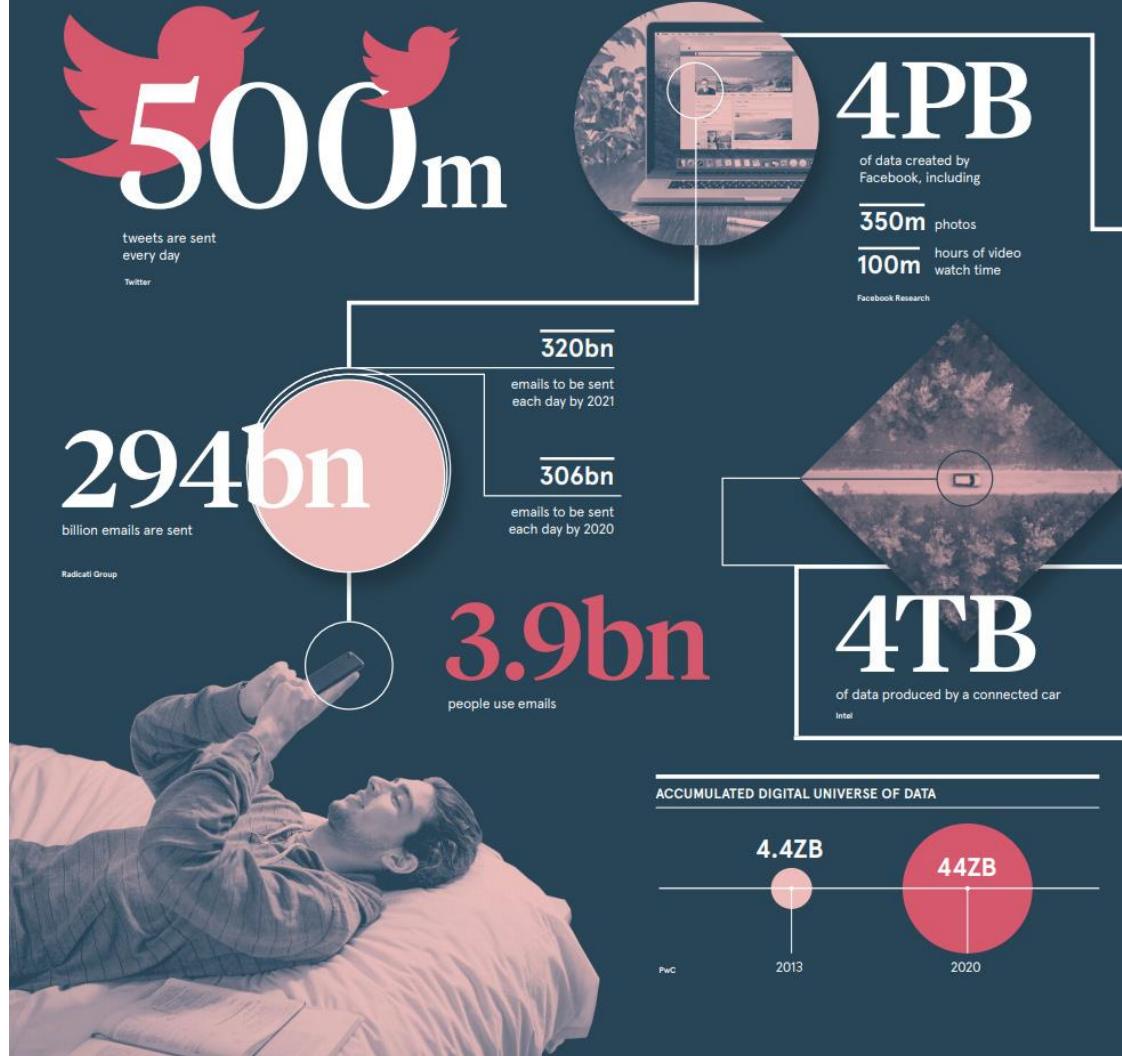
- Inoltre, il successo di AlexNet, BERT e GPT ha dimostrato che il progresso del ML nell'ultimo decennio **si basa sull'accesso a una grande quantità di dati**
- Ciò è stato reso possibile **dall'enorme mole di dati disponibile** negli ultimi anni, dovuto all'espansione delle piattaforme online, dei dispositivi connessi e delle interazioni digitali

Dimensione dello storage mondiale (Exabytes) ¹					
1986	1993	2000	2007	2014	2020
2.6	15.8	54.5	295	5000	6800

[1] <https://rivery.io/blog/big-data-statistics-how-much-data-is-there-in-the-world/>

A DAY IN DATA

The exponential growth of data is undisputed, but the numbers behind this explosion – fuelled by internet of things and the use of connected devices – are hard to comprehend, particularly when looked at in the context of one day

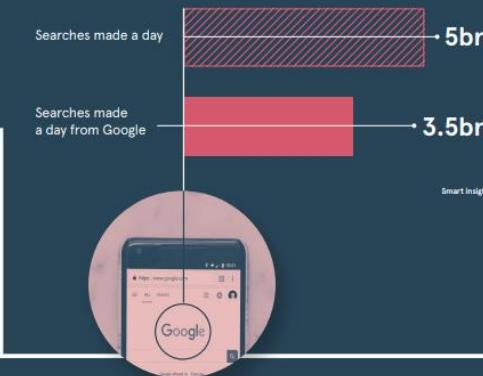


DEMYSTIFYING DATA UNITS

From the more familiar 'bit' or 'megabyte', larger units of measurement are more frequently being used to explain the masses of data

Unit	Value	Size
b	bit	0 or 1
B	byte	8 bits
KB	kilobyte	1,000 bytes
MB	megabyte	1,000 ² bytes
GB	gigabyte	1,000 ³ bytes
TB	terabyte	1,000 ⁴ bytes
PB	petabyte	1,000 ⁵ bytes
EB	exabyte	1,000 ⁶ bytes
ZB	zettabyte	1,000 ⁷ bytes
YB	yottabyte	1,000 ⁸ bytes

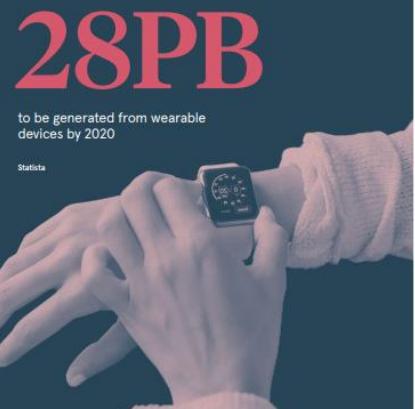
*A lowercase "b" is used as an abbreviation for bits, while an uppercase "B" represents bytes.



463EB

of data will be created every day by 2025

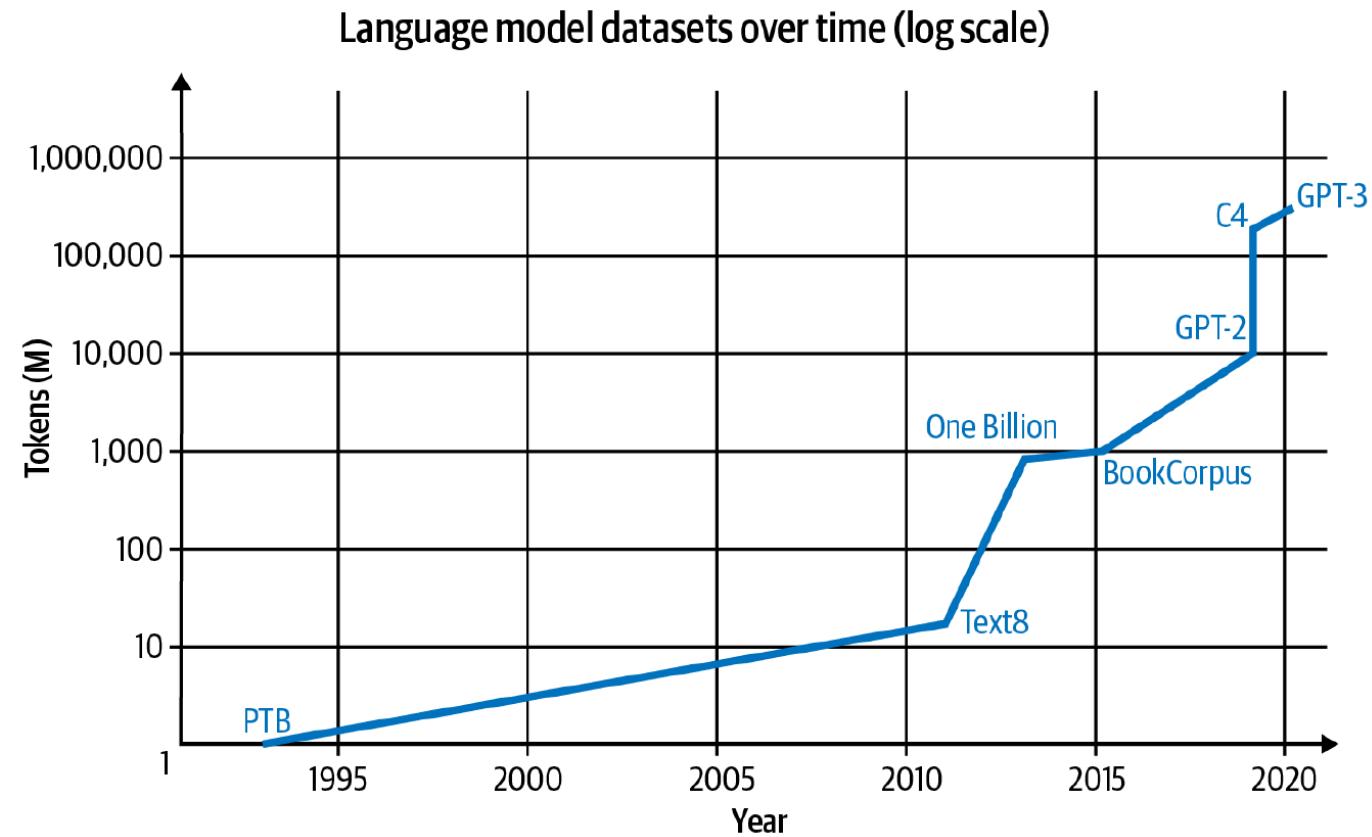
IDC



RACONTEUR

Mind vs Data

- Prendendo come esempio i language model, nel 2013 ci si è entusiasmati quando è stato rilasciato il "One Billion Word Benchmark", che **contiene 0,8 miliardi** di token
- Nel 2020, GPT-3 ha usato un dataset di **500 miliardi di token**



Summary

Summary

- Sebbene la maggior parte delle persone sia familiare con l'apprendimento automatico nelle applicazioni rivolte ai consumatori, la maggior parte dei casi d'uso dell'apprendimento automatico riguarda le aziende.
- Anche se l'apprendimento automatico può risolvere molti task con successo, non può risolvere tutte le sfide e certamente non è adatto per tutti i problemi.
- Ci sono importanti differenze tra l'apprendimento automatico nella ricerca e l'apprendimento automatico nell'ambito aziendale

Summary

- Ogni progetto deve essere motivato da obiettivi, e i progetti di apprendimento automatico non fanno eccezione
 - La maggior parte delle aziende non si preoccupa delle metriche di performance a meno che non possano influenzare le metriche aziendali
- Prima di costruire un sistema di apprendimento automatico, è necessario comprendere i requisiti che il sistema deve soddisfare
- Costruire un sistema di apprendimento è un processo iterativo. Un problema può essere formulato in diversi modi e questo può influenzarne la complessità
- I dati sono di fondamentale importanza nell'apprendimento automatico