Refresher

Parte I

Indice

Capitolo 1

Basics

1.1 Introduzione al decision making

Il *decision making* è una parte integrante del ruolo di gestione; i dirigenti prendono centinaia di decisioni ogni giorno.
La *decision science* è una disciplina che usa tecniche quantitative per il processo decisionale, ovvero suggerisce quale decisione prendere in base a tecniche quantitative; è usata da ricerca operativa, statistica, inforamtica ...
Per poter applicare queste tecniche, spesso è richiesto una elaborazione dei dati grezzi in dati elaborati per ottenere un risultato.
Il decision making ha 3 proprietà:

- **Accuratezza**
- **Spiegabilità:** è la capacità di spiegare un risultato
- **Accettabilità:** la procedura per passare dal dato grezzo al risultato è accettata da tutte le parti coinvolte

1.1.1 Data exploration

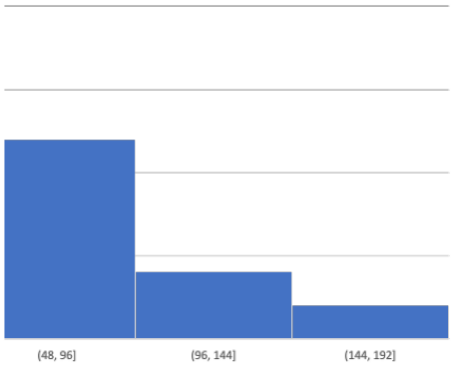
Supponiamo di avere un dataset di dipendenti, e di voler trarre una decisione sul licenziare o meno un dipendente in base all'assenteismo; vogliamo trarre delle informazioni dai dati usando delle semplici *statistiche descrittive*.
Nel nostro dataset troveremo sia dati **categorici** (etichette) che **numerici**.

Bucketizzazione

È un'operazione che permette di trasformare un dato numerico in uno categorico: l'insieme dei valori viene suddiviso in intervalli disgiunti; successivamente, ciascun valore viene sostituito con l'etichetta del suo intervallo.

Istrogammi e distribuzione

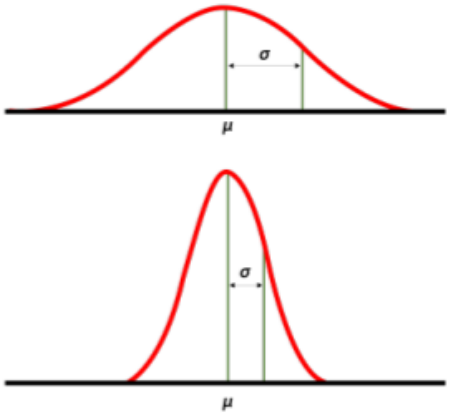
Una volta fatta la bucketizzazione, è più facile la visualizzazione grafica della distribuzione dei dati tramite istogrammi.



Deviazione standard campionaria

La deviazione standard misura quanto i valori numerici di una caratteristica sono dispersi rispetto alla media.
Se il grafico ha una forma a campana simmetrica:

- σ piccola indica valori vicini alla media
- σ grande indica valori lontani dalla media, valori dispersi



La formula prevede di calcolare la varianza di ciascun valore (distanza dalla media al quadrato) divisa per il numero di valori, il tutto sotto radice.

Mediana

La mediana è quel valore che divide l'insieme a metà; è quel valore per cui il 50% dei valori è più piccolo e l'altro 50% è più grande.

Quartili

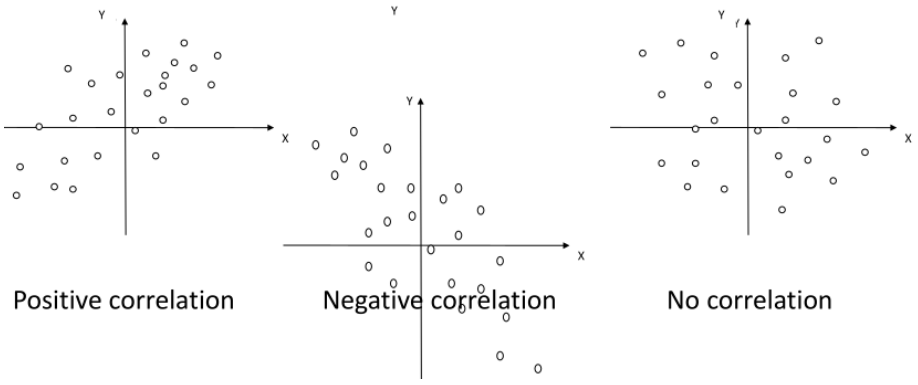
- **Primo quartile:** valore x tale che 25% dei valori è minore di x
- **Secondo quartile:** valore x tale che 50% dei valori è minore di x
- **Terzo quartile:** valore x tale che 75% dei valori è minore di x

$\Rightarrow$ il secondo quartile coincide con la mediana

La **differenza interquartile** (differenza tra terzo e primo quartile) è utile per fare analisi su dati la cui distribuzione non è a campana simmetrica, dato che la deviazione standard ha senso di essere usata solo su dati che seguono quella distribuzione (altrimeniti la media non sarebbe un valore tipico, non avrebbe senso misurare la distanza da essa).

1.1.2 Relazione tra due feature

Ci si può chiedere se due colonne di un dataset siano correlate, ovvero se sono legate a tale punto che mi basta saperne una per conoscere anche l'altra.
Si possono mettere le due caratteristiche su un piano e rappresentarle graficamente per vedere se esiste una correlazione:



È possibile usare due metriche:

- **Varianza:** dà informazioni sulla variabilità interna di una variabile
- **Covarianza:** dà informazioni su come due variabili variano insieme; mi dice quanto il variare di una influenzi il variare dell'altra
  - *Positiva:* le variabile variano nella stessa direzione

- *Negativa:* le varaibili variano nella direzione opposta
- *Zero:* nessuna correlazione

La covarianza viene usata per ottenere un numero tra -1 e 1 (tramite il *coefficiente di correlazione di Pearson*) che mi dice quanto due variabili sono correlate tra loro.

- due variabili si dicono indipendenti se non condividono informazioni
- se sono dipendenti, la *quantità di informazione* condivisa si stima con la correlazione

Bisogna fare attenzione a fare una distinzione tra correlazione e causa. Potrebbe sembrare che due correlazioni siano correlate solo perchè sono correlate ad una terza grandezza; è il caso ad esempio di suicidi e mangiare aringhe.

Distribuzioni asimmetriche

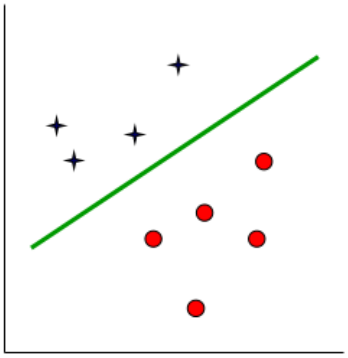
Quando un istogramma è costruito su valori che seguono una *distribuzione normale*, allora la forma delle colonne sarà *a campana*; tuttavia, in casi reali gli istogrammi sono spesso asimmetrici.

1.1.3 Decision Making via algorithms

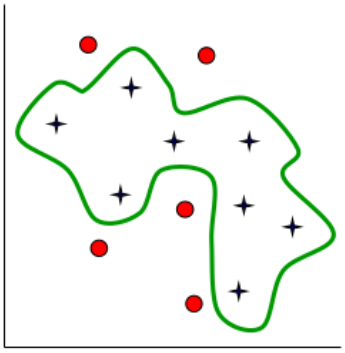
Per quale motivo si cerca di rispondere ad un problema tramite un modello supervisionato? Non basterebbe avere un algoritmo?
Gli algoritmi applicano una **teoria** alle grandezze (ad esempio la deteriorità di un motore elettrico dovuta alla temperatura); se ho una teoria, la posso **esplicitare con un algoritmo:** ho una funzione che lega input e output; non ho più bisogno di avere esempi, ma ho la teoria che mi dice che la deteriorità del motore è dovuta al numero di ore, ad una determinata temperatura...
Molti problemi non possono essere risolti in questo modo, dato che non c'è un'equazione esplicita; la devo *tirare fuori* dall'addestramento del modello.

1.1.4 Decision Making via Supervised Machine Learning

Non viene usata una formula e algoritmo che descrive in modo esplicito un fenomeno per derivare l'informazione per prendere una decisione. Viene usata una **formula generica con dei parametri, che vengono regolati per minimizzare l'errore**. Si tratta dunque di trovare i valori giusti dei parametri, viene fatto nella fase di addestramento.
Viene associato a ciascun valore un peso; alla somma pesata viene poi aggiunta una costante a cui *viene applicato un sogliatore*; in questo modo l'output è una funzione non lineare, invece che una semplice somma pesata.
La procedura di addestramento consiste nel **considerare l'errore come funzione dei pesi**, andando poi a modificarli per minimizzarli.
Un singolo neurone può approssimare solo una funzione lineare:



Una *rete neurale multi-livello* può teoricamente approssimare qualsiasi funzione dopo una certa soglia (*diventa Touring-complete*):

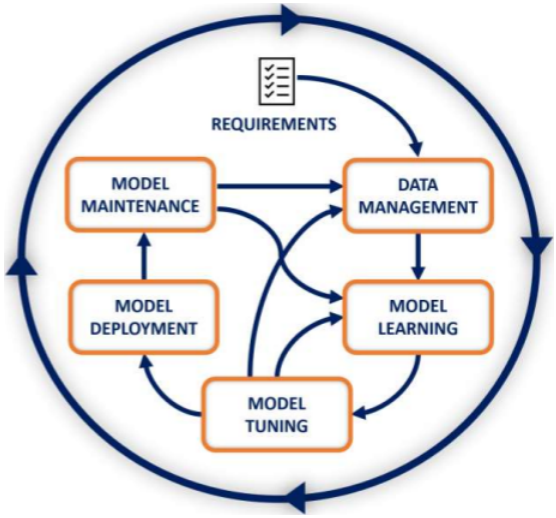


1.1.5 Ciclo di vita di un modello di ML

L'addestramento è solo una delle parti di vita di un modello di machine learning:

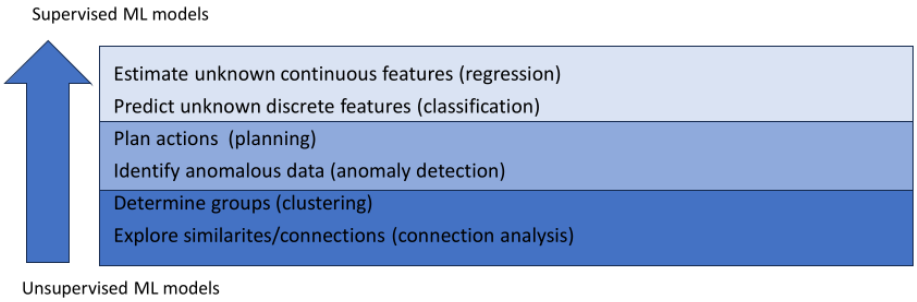
- **Data Management:** costruzione del dataset, controlli di vario tipo, eliminare colonne correlate tra loro ...
- **Model Learning:** addestramento del modello
- **Model Tuning:** fase post-addestramento in cui si danno al modello dati che non ha mai visto
- **Modelo Deployment:** il modello viene messo in produzione nel caso in cui soddisfi i requisiti; altrimenti, viene riportato indietro alle fasi precedenti
- **Model Maintenance:** si può avere un degradamento delle prestazioni perchè cambiano le distribuzioni dei dati in ingresso (è preferibile che siano gaussiane e stazionarie); devo fare gli opportuni controlli

Ciascuna di queste fasi può subire un attacco.



1.2 Introduzione a Machine Learning

- I modelli supervisionati vanno addestrati; sono soggetti ad attacco in fase di inferenza ed in fase di training
- I modelli non supervisionati processano i dati sulla base delle loro proprietà, non su esempi che hanno visto; ad esempio, le tecniche di clustering sono basate sulla definizione di distanze intragruppo e intergruppo. Non esiste una fase di training in cui possono essere attaccati



Regressore e classificatore fanno parte delle task di completamento.

1.2.1 Paradigma Supervised Learning

Il problema è **imparare una relazione tra input e output a partire da degli esempi**; viene prima addestrato il modello e poi se ne ottiene uno (adde-

strato) che può essere messo in produzione. In sede di addestramento il modello viene regolato per cercare di **minimizzare l'errore**.
Può essere attaccato, per cercare di violare alcune proprietà (accuratezza, ma non solo), posso cercare di fargli sbagliare un input in particolare, posso fare violazioni di privacy cercando se un determinato elemento appartiene al training set ...
L'idea per creare un modello è:

- viene dato un input, e il modello produrrà un output sicuramente sbagliato
- si va a modificare i pesi per cercare di ridurre l'errore
- si continua così fino a che ottengo dei buoni risultati che minimizzano l'errore
- una volta finito questo processo, il modello non ancora pronto, perchè funziona bene sui dati che ha già visto

  $\rightarrow$ devo fare una fase di testing sui dei valori che non ha mai visto (*test set*)

Supervised ML Training in a nutshell

- Il modello interno più semplice è una somma pesata; si ha un input multivariato, dove ogni variabile viene moltiplicata per un peso per poi fare la sommatoria
- A questa somma pesata viene poi applicata una soglia (ad esempio, se è sopra la soglia considero il risultato benigno mentre se è sotto considero maligno)
- Quando ottengo il risultato, viene confrontato con quello reale (siamo nella fase di training), e se ho fatto un errore si va a modificare i pesi cercando di diminuire l'errore
- Si continua tante volte in questo modo sul training set, sperando che l'errore continui a diminuire sempre di più fino a fermarmi nel punto di minimo ($gradiente = 0$, ovvero la derivata delle funzioni multivariate)

All'input viene aggiunta una costante che prende il nome di **bias**, che aiuta a migliorare l'output riducendo la varianza e beneficiando all'accuratezza. Il training *decide la quantità di bias* che massimizza l'accuratezza.
Mettendo più strati di neuroni (l'output di uno va in input ad un altro) si riesce a fare un addestramento più rapido; con 3 o più strati si dice che la rete è *Touring completa.*

Error backpropagation

Uno dei problemi della rete neurale tradizonale è quello di non riuscire a fare una corretta *error backpropagation*, ovvero non si riesce a distribuire l'errore anche sugli strati interni.
Per arginare questo problema si usano i primi strati per modificare la rappresentazione degli input, comprimendoli in una parte più piccola, mentre sono solamente gli ultimi quelle che vengono effettivamente addestrati.

Overfitting

Bisogna fare attenzione all'*overfitting*: significa che il modello commette errore pari a 0 sui dati del training set, ma che su dati mai visti prima commette un grande errore; questo accade perché il modello non è in grado di generalizzare.
Per evitare che un modello vada in overfitting, generalmente un il training viene fermato prima di raggiungere un'accuratezza perfetta sui dati di training.

1.2.2 Diversi modelli di Machine Learning

- **Classificazione:** mappa vettori di caratteristiche in *categorie* o *classi*
- **Anomaly Detection:** serve a capire se un input è anomalo rispetto a quello atteso
- **Predizione:** un classificatore può anche agire da predittore, come ad esempio prevedere l'esito di un azione. Per fare il training gli do sole la parte iniziale dei dati e li faccio classificare come successo/insuccesso
- **Planning:** sono adatti in cui non riesco a calcolare l'errore per ogni singolo input; ad esempio, se gioco a scacchi è difficile classificare una singola mossa come buona o cattiva...

  $\rightarrow$ si usa un sistema di *reward*, non calcolato sulla singola mossa ma su una sequenza di mosse
- **Connection Analytics:** analizza la relazione tra utenti/sistemi; fondamentale per la sicurezza (ad esempio, individuare nodi centrali o vulnerabili)

$\Rightarrow$ lo stesso problema può essere risolto con modelli diversi; la scelta dipende dai dati disponibili