Dipartimento
di Matematica
e Informatica

Corso di Laurea
in Informatica

Cervello e computer: bellezza e segreti dei bit di tutti i giorni

cybersecurity • intelligenza artificiale • realtà virtuale • Android

Ciclo di seminari di divulgazione informatica
in collaborazione con NOVA a.p.s.

NOVA

Ferrara, 8–12 Giugno 2020

Programma

Lunedì 8 Giugno: Carlo Giannelli
Cyber security: istruzioni per l'uso – Principi di sicurezza informatica

Martedì 9 Giugno: Guido Sciavicco
Come imparano le macchine – Principi di intelligenza artificiale

Mercoledì 10 Giugno: Marco Alberti
Neuroni di bit – Reti neurali e applicazioni

Giovedì 11 Giugno: Antonino Casile
Informatica e percezione sensoriale – L'ultima frontiera della realtà virtuale

Venerdì 12 Giugno: M. Roma, G. Turri, L. Travaglia – NOVA Ferrara
La nascita di un'app Android – Programmazione in Android

Disclaimer

- La presentazione, il filmato, i materiali e i contenuti in essi inclusi sono di proprietà dell'Università di Ferrara
- Il diritto morale d'autore ("Proprietà Intellettuale") appartiene ai singoli docenti/relatori dell'evento
- L'utilizzo è concesso per uso esclusivo e personale
- Nessun altro utilizzo può essere legittimamente esercitato senza la previa autorizzazione scritta dell'Ateneo e dei proprietari del diritto morale d'autore
- Qualunque abuso verrà perseguito a norma di legge
- Per ulteriori informazioni visitare il sito dmi.unife.it/stageInformatica

Come imparano le macchine

Guido Sciavicco

Dipartimento
di Matematica
e Informatica

Apprendimento automatico e IA

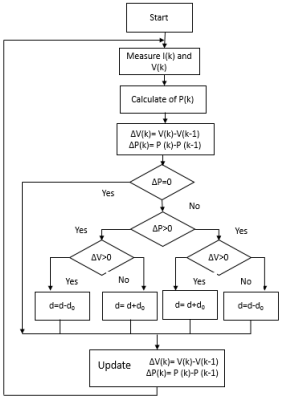
In questo seminario parleremo di intelligenza artificiale, e cercheremo di impararne i principi fondamentali. Il primo passo, però, è quello di comprendere la corretta **terminologia** per poterla, poi, utilizzare. Cominciamo con una lista di termini ben noti: **intelligenza artificiale**, **machine learning**, **big data**, **reti neurali**. Come si relazionano tra loro, e come si relazionano con l'informatica classica che conosciamo bene?

Apprendimento automatico e IA

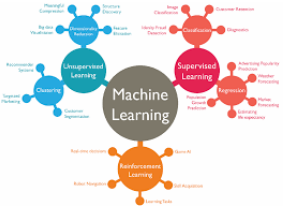
Il mondo reale ci presenta quotidianamente **problemi computazionali** da risolvere. Alcuni di questi hanno la caratteristica di essere *formulati matematicamente* (es.: calcolare una funzione, ordinare una lista, o calcolare la strada migliore per arrivare ad una destinazione). Altri, invece, si presentano in *maniera vaga ed imprecisa* (es.: effettuare una diagnosi medica a partire da immagini, riconoscere la firma autografa di una persona, imparare a giocare a un gioco da tavolo).

Apprendimento automatico e IA

I primi sono quelli di cui si occupa l'**informatica classica** (che produce la grandissima maggioranza dei sistemi che utilizziamo quotidianamente). La soluzione ad un problema formulato in maniera precisa è un algoritmo, cioè una procedura ben definite, composta da passi semplici, non equivocabili, ed implementabili.



Apprendimento automatico e IA



I problemi la cui formulazione è inerentemente imprecisa, invece, sono quelli di cui si occupa l'**intelligenza artificiale moderna** (**IA**) (che sta progressivamente integrandosi con i sistemi informatici classici, non per sostituirli, ma per, in certi casi, migliorarli e dotarli di capacitá altrimenti impensabili).

Apprendimento automatico e IA

Per sopperire alla mancanza di una formulazione rigorosa dei problemi che affronta, la IA si basa su **schemi di apprendimento**, che permettono di approssimare la conoscenza attraverso l'analisi dei dati. L'apprendimento sarà tanto più affidabile quanto più ampi e rappresentativi sono i dati da cui si impara (**big data hypothesis**).

Apprendimento automatico e IA

Approccio classico:

- esiste una teoria soggiacente;
- non impariamo dai dati;
- dobbiamo essere precisi e non ammettiamo errori.

Approccio basato su IA:

- non esiste una teoria soggiacente;
- impariamo dai dati;
- ammettiamo una certa imprecisione.

I due approcci non sono in contrapposizione, ma ortogonali. In molti casi, sistemi classici sono *arricchiti* con sistemi basati su IA, che li integrano e li completano. E' fondamentale comprendere che un approccio basato su IA può, potenzialmente, risolvere problemi altrimenti irrisolvibili, ma è inerentemente approssimato, e non c'è modo di prevedere la magnitudine dei potenziali errori nelle soluzioni proposte.

Apprendimento automatico e IA

Gli schemi di apprendimento possibili sono molti. Alcuni di questi sono stati resi più famosi di altri dai mezzi di informazione divulgativa, come le **reti neurali** ad esempio. Per comprendere il funzionamento degli schemi, però, è conveniente concentrarci su schemi più semplici, che è quello che faremo noi, e studiando, concretamente, uno schema chiamato **albero decisionale**.

Apprendimento automatico: tassonomia

Diremo che un algoritmo di apprendimento è un algoritmo di **classificazione con supervisione** quando utilizza dati già classificati. Supponiamo, ad esempio, di voler costruire una applicazione di IA che determina, in base alla situazione meterologica in un certo giorno, ed in una certa area specifica, (dati opportunamente codificati), se una partita di calcio, programmata in quel giorno, potrá venir giocata.

È noto che, sebbene esistano delle linee guida, la scelta è arbitraria e dipende dalla sensibilitá dell'arbitro. Quindi, si tratta di una situazione che non ha una caratterizzazione matematica (almeno non una chiara).

Apprendimento automatico: tassonomia

Un algoritmo di classificazione con supervisione (che è una categoria di schemi di apprendimento) per questo caso si basarebbe sul un insieme di coppie:

$$(\text{situazione}, \text{giocata/non giocata})$$

che vengono dette **istanze**. In questo semplice esempio, l'algoritmo che cerchiamo deve decidere tra un si o un no, quindi la classificazione è binaria. L'informazione sul fatto che la partita è stata giocata oppure no viene detta **classe** del problema. In un caso più generale si possono avere più classi.

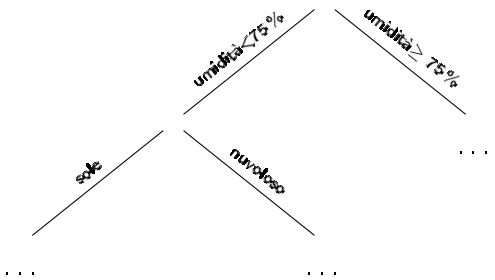
Alberi decisionali

In questo esempio partiremmo da una situazione che può essere descritta in forma tabellare (**dataset**):

tempo	*temperatura*	*umiditá*	*vento*	*si gioca*
sole	*caldo*	*alta*	*no*	*no*
sole	*caldo*	*alta*	*si*	*no*
nuvoloso	*caldo*	*alta*	*no*	*si*
pioggia	*media*	*normale*	*no*	*si*
pioggia	*freddo*	*normale*	*no*	*si*
pioggia	*freddo*	*normale*	*si*	*no*
nuvoloso	*freddo*	*normale*	*si*	*no*
sole	*media*	*alta*	*no*	*no*
sole	*freddo*	*normale*	*no*	*si*
pioggia	*media*	*normale*	*no*	*si*
sole	*media*	*normale*	*si*	*si*
nuvoloso	*media*	*alta*	*si*	*si*

Alberi decisionali

Un **albero decisionale** è uno schema di apprendimento che, dato un caso come quello precedente, **impara** un oggetto con forma di albero:



Le colonne del data set sono le variabili che codificano ogni istanza, e vengono chiamate **attributi**. Nel nostro esempio abbiamo attributi come 'tempo' (soleggiato o nuvoloso), 'umidità relativa' (in percentuale), ecc. Alle foglie di questo albero si troverà una **decisione**.

Alberi decisionali

Tutti gli schemi di apprendimeno danno luogo a problemi computazionali cosiddetti **hard**, che nella teoria della computazione classica non possono essere risolti in maniera efficiente. Le tecniche accettate di apprendimento si basano su **approssimazioni statistiche** alle soluzioni. Guardiamo quindi cosa succede:

problema è troppo complesso.

- Uno schema di apprendimento come un albero *approssima* un problema reale che non può essere descritto in maniera matematicamente precisa;
- Imparare un caso concreto a sua volta *approssima* lo schema stesso perchè il

Imparare un albero decisionale

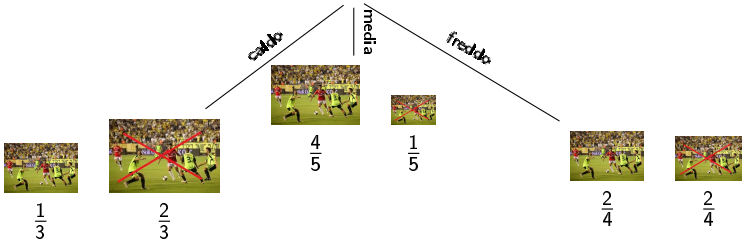
Ora che abbiamo tutti i concetti principali in mano, possiamo capire in maniera molto astratta come si impara un albero decisionale. L'idea più importante è quella della *confusione* di un data set. La confusione di un data set è il grado di non-omogeneità della classi delle istanze. Nostro esempio avevamo 5 istanze in cui la partita era sospesa, e 7 istanze in cui si giocava. Il livello di confusione è calcolato a partire dalle quantità $\frac{5}{12}$ e $\frac{7}{12}$. Più queste sono vicine tra loro, meno chiara è la situazione, e viceversa.

$\frac{7}{12}$

$\frac{5}{12}$

Imparare un albero decisionale

Ma se ci basiamo su 'temperatura', e dividiamo (effettuiamo uno **split**) sui tre valori possibili, otteniamo la seguente situazione:



L'idea è: ad ogni iterazione, scelgo l'attributo e lo split che mi fa guadagnare più informazione *globale* (cioè in media su tutti i rami).

Imparare un albero decisionale

Una volta che il processo è terminato, avremo un albero decisionale che contiene al suo interno l'informazione del data set dal quale ha imparato, in maniera 'usabile'. *Classificare* una nuova istanza è l'operazione per la quale abbiamo fatto tutto ciò: una nuova istanza, nel nostro esempio, sarebbe una nuova situazione di gioco/non gioco; di fronte a questa, la possiamo leggere attraverso l'albero, e la decisione alla quale arriviamo sarebbe quella predetta. Questo si può interpretare in vari modi. Il più naturale è: l'albero ha in se l'esperienza di chi ha creato il data set, e la decisione che prende sarebbe *probabilmente* quella che prenderebbe la stessa entità. Quindi, la macchina ha *imparato* la teoria non scritta delle decisioni di gioco/non gioco imitando la/le persona/e che hanno creato il data set.

Un albero decisionale al lavoro

Come facciamo a rendere *usabile* un albero decisionale, per costruire una machina che abbia a tutti gli effetti un comportamento migliorativo? Immaginando di poter usare un albero decisionale già pronto (una libreria per alberi decisionali), allora possono inserirlo in un algoritmo classico.

Un albero decisionale al lavoro

Ecco un semplice sistema 'intelligente' che impara la teoria delle condizioni di un campo da gioco usando esperienza che viene aggiornata ad ogni passo:

- Chiedo di descrivere una nuova situazione di gioco;
- Controllo quante situazioni diverse conosco già;
- Se sono abbastanza, allora:
  - Creo un albero decisionale;
  - Provo a classificare la situazione attuale;
- Inserisco la situazione attale e la risposta **vera** tra quelle che conosco;
- Torno al passo 1.

Un albero decisionale al lavoro

La libreria open source *Weka* ci permette di fare questo con poche linee di codice, usando una classe già esistente per la creazione di alberi decisionali, e utilizzando un sistema proprietario di codifica delle istanze.

Un albero decisionale al lavoro

Nel nostro esercizio faremo proprio questo, e lo proveremo sul problema di insegnare alla macchina a riconoscere il gruppo di appartenenza di un *vertebrato* in base alle sue caratteristiche. Le tipiche caratteristiche che si prendono in considerazione sono il fatto di avere piume oppure no, il fatto di essere capaci di nuotare oppure no, il fatto di essere capaci di volare oppure no... I gruppi possibili sono: *pesci*, *anfibi*, *uccelli*, *rettili*, e *mammiferi*. Questi concetti si insegnano a bambini delle scuole medie: quanto sarà complesso per un computer impararli?