

OpenLab 2018 - Machine Learning

Claudio Chiappetta

April 27, 2018

Che cos'è l'intelligenza?

Che cos'è l'intelligenza?

Definizione operativa

Manipolazione di stringhe al fine di risolvere problemi

Perche' intelligenza artificiale?

- Automatizzare problemi gia' risolvibili da umani
- Risolvere problemi troppo difficili per il cervello umano
- Migliorare la comprensione dei processi mentali

Storia

- 1956: Conferenza di Dartmouth
 - Perceptron
 - Reasoning as search
- AI winter 1974-1980
- Second AI Boom 1980-1987
- Second AI winter 1987-1993
- Third AI boom 1993-2011
- Situazione attuale
 - Deep Learning

Perceptron

Perceptron

- $f(\underline{x}) = \Theta(\underline{w} \cdot \underline{x} + b)$

Perceptron

- $f(\underline{x}) = \Theta(\underline{w} \cdot \underline{x} + b)$
- Training (data set x^j classificato):
 - 1 valori random a w_i
 - 2 valuto $f(x_i)$
 - 3 se la classificazione è corretta, torno a 2)
 - 4 se no $w_i = w_i + \eta \cdot d \cdot x_i^j$

Perceptron

- $f(\underline{x}) = \Theta(\underline{w} \cdot \underline{x} + b)$
- Training (data set x^j classificato):
 - 1 valori random a w_i
 - 2 valuto $f(x_i)$
 - 3 se la classificazione è corretta, torno a 2)
 - 4 se no $w_i = w_i + \eta \cdot d \cdot x_i^j$

È un neurone!

Reti neurali

Insieme di percertroni, la cui topologia si può descrivere con una matrice W_{ij} (connectivity matrix)

- Feed Forward
- Recurrent
- Convolutional

Reti neurali

Insieme di percertroni, la cui topologia si può descrivere con una matrice W_{ij} (connectivity matrix)

- Feed Forward
- Recurrent
- Convolutional

→ Deep learning

Reti neurali

Insieme di percertroni, la cui topologia si può descrivere con una matrice W_{ij} (connectivity matrix)

- Feed Forward
- Recurrent
- Convolutional

→ Deep learning

Training

Per reti feed forward: Backpropagation

In generale è un problema di ottimizzazione

Storia

- 1956: Conferenza di Dartmouth
 - Perceptron
 - Reasoning as search
- AI winter 1974-1980
- Second AI winter
- Situazione attuale
 - Deep Learning

Machine Learning

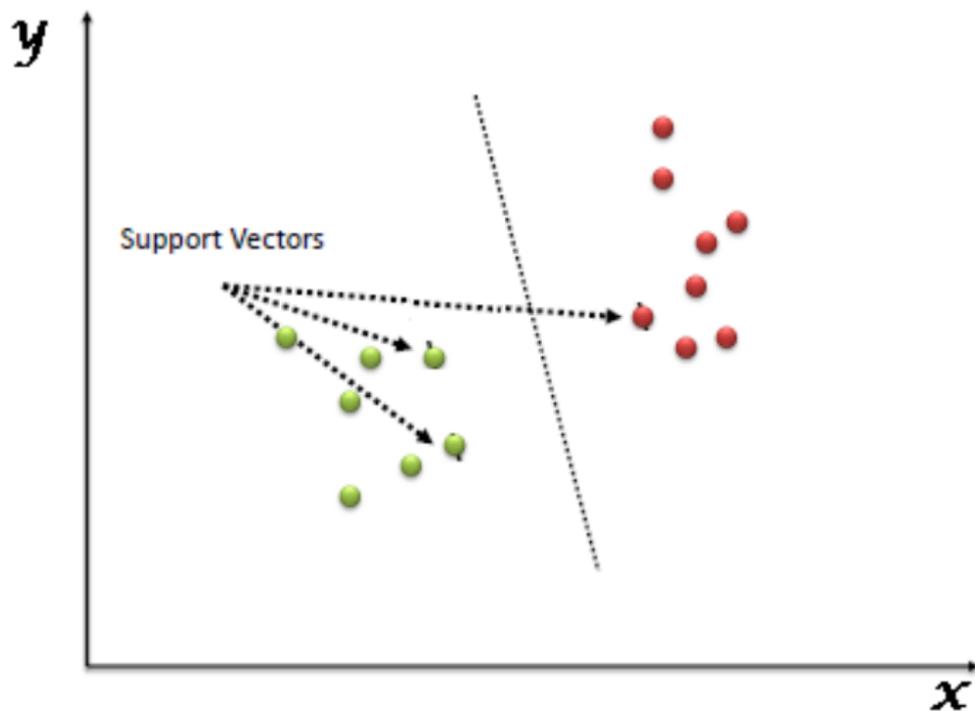
Programmi che utilizzano un corpus di dati per migliorare la propria performance

- Supervised learning
- Unsupervised learning
- Reinforcement learning

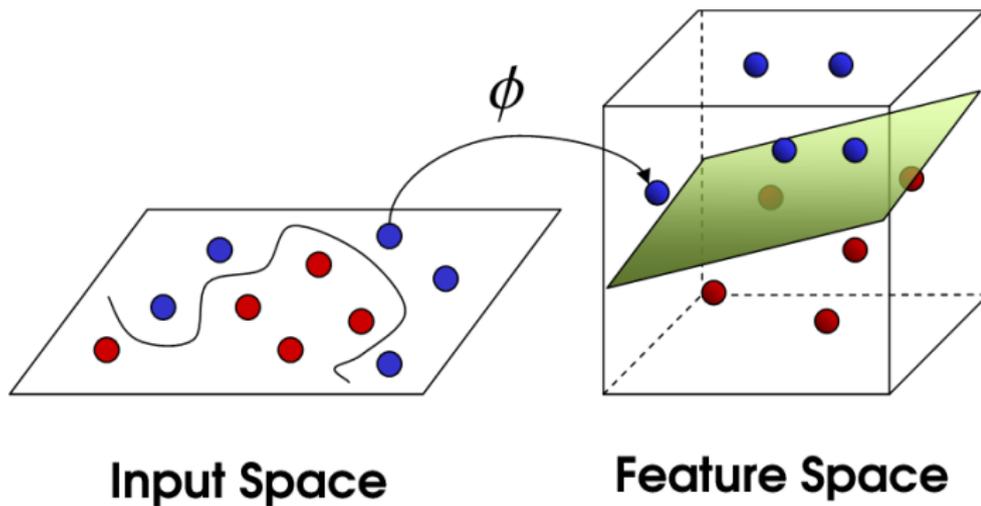
Supervised learning (Classificazione)

- L'algoritmo vuole "imitare" un set di input-output noti
- Usato un po' ovunque
 - Spam
 - Diagnosi automatiche
 - Classificazione eventi di un acceleratore
 - Previsione andamenti di mercato

SVM



SVM



Bayes

- Assunzione di base: features indipendenti
- Calcolare $P(C_k | \underline{x}) \xrightarrow{\text{Bayes}} \frac{1}{Z} P(C_k) \prod_i P(x_i | C_k)$

Bayes

- Assunzione di base: features indipendenti
- Calcolare $P(C_k | \underline{x}) \xrightarrow{\text{Bayes}} \frac{1}{Z} P(C_k) \prod_i P(x_i | C_k)$
- Scegliere: $f(\underline{x}) = \underset{k \in \{1 \dots K\}}{\text{argmax}} P(C_k) \prod_i P(x_i | C_k)$
- Molto veloce, scala linearmente con i dati

Unsupervised learning (Clustering)

- Trovare pattern nelle features
- Esempi pratici:
 - Organizzare i prodotti in un supermercato (fisico o virtuale)
 - Sicurezza informatica (behavioral detection)
 - Linguistica

Algoritmi

- Hierarchical linkage
 - Creo una gerarchia di cluster aggregando o dividendo
- K-means
 - 1 Parto scegliendo K clusters come intorno di K vettori random
 - 2 Metto ogni pto del dataset nel cluster più vicino a lui
 - 3 Ridefinisco il K cluster come l'insieme del centroide del K cluster attuale
 - 4 Ripeto per ridurre l'"errore" a mio piacere
- Reti neurali

Rinforcement learning

- Non un supervisore onnisciente, ma una risposta ad ogni azione
- Bilancio fra guadagno a breve e a lungo termine
- Applicazioni molto fantasiose:
 - Finanza
 - Robot
 - Videogiochi

Definizioni

- A : azioni disponibili per l'agente
- S : insieme di stati dell'ambiente
- R : rewards per coppie stato-azione
- $P_a(s,s')$: probabilita' di transizione
- π : Policy (come scelgo le azioni)
- $V_\pi(s)$: ritorno atteso
- $Q_\pi(s,a)$: ritorno atteso in seguito ad a

Q-learning

- Scelgo un'azione in base ad una tabella $S \times A$ (Q-table)

Q-learning

- Scelgo un'azione in base ad una tabella $S \times A$ (Q-table)
- Equazione di Bellmann

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{r_t + \gamma \cdot \max_a Q(s_{t+1}, a)}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

reward discount factor estimate of optimal future value

Codice

Implementiamo qualcuno di questi algoritmi con le librerie di Python3

Prerequisiti:

- `pip3 install -U sklearn`

Lecture consigliate

- Google google google
- Enrico Prati, "Mente artificiale" (2017), EGEA