Apprendimento Automatico
Rappresentazione e Kernels

Fabio Aiolli
www.math.unipd.it/~aiolli

Sito web del corso
www.math.unipd.it/~aiolli/corsi/1617/aa/aa.html

Rappresentazione dei dati con i kernel

- Abbiamo una serie di oggetti $S = \{x_1, x_2, \dots, x_n\}$. Come li rappresentiamo?

- Rappresentazione Classica:
  - $\varphi(x) \to F$

- Rappresentazione con Kernel:
  - $k : X \times X \to R$ (confronti a coppie, funzione simmetrica)
  - Il dataset S è quindi rappresentato come una matrice simmetrica $K_{i,j} = k(x_i, x_j)$

Vantaggi della Rappresentazione dei dati con i kernel

- La rappresentazione dei dati con matrici kernel ha dei vantaggi:
  - Lo stesso algoritmo può analizzare tipologie di dati diversi
  - La progettazione dei kernel e degli algoritmi è modulare
  - Più semplice integrare *viste diverse* dei dati

- La dimensionalità dei dati dipende solo dal numero di oggetti e non dalla loro dimensione vettoriale

- La comparazione tra oggetti può risultare più semplice rispetto a una loro esplicita rappresentazione: algoritmi per il calcolo del kernel vs prodotti scalari

Metodi Kernel

- Molti metodi kernel, comprese le SVM possono essere interpretati come algoritmi che, dato un insieme di oggetti S, risolvono il seguente problema:

$$min_{f \in H} L(f(x_1), f(x_2), \dots, f(x_n)) + \Lambda ||f||_H$$

- L è una funzione costo (loss) associata al rischio empirico (errore sul training) e Λ è un opportuno coefficiente di regolarizzazione
- La norma è legata alla “smoothness” della funzione. Cosa si intenda per ”smoothness” dipende dal kernel considerato (spazio delle features)

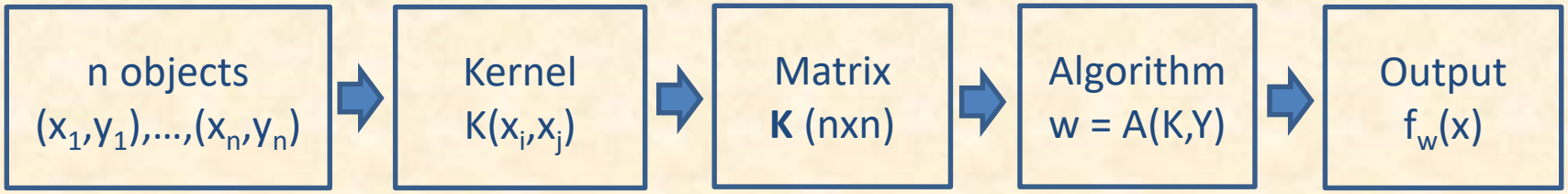
- Si può dimostrare che un problema di ottimizzazione di tale forma ha sempre una soluzione del tipo:

$$f(x) = w \cdot \phi(x) \; = \sum_{i=1}^{n} \alpha_i k(x, x_i)$$

Ovvero, il problema da ottimizzare si può riformulare con n variabili. Se n<<d (la dimensione di $\phi(x)$) si ha un notevole vantaggio computazionale!

Modularità metodi Kernel

Modularità nella progettazione/definizione del kernel (rappresentazione) e dell'algoritmo per il calcolo del modello per classificazione/regressione/ranking, ecc.



Caratteristiche del Kernel

- I metodi kernel operano su matrici semidefinite positive

- Un kernel (semidefinito positivo) è una funzione simmetrica $k(x_i, x_j) = k(x_j, x_i)$ tale che, per ogni n>0, $x_1, x_2, \dots, x_n \in S$ e $c_1, c_2, \dots, c_n \in \mathbb{R}$:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) = \boldsymbol{c}^T K \boldsymbol{c} \geq 0$$

Kernel Trick

Essendo $k(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$

Ogni algoritmo per dati vettoriali che può essere espresso in termini del prodotto scalare tra vettori può essere implicitamente eseguito nello spazio delle feature associato ad un determinato kernel, rimpiazzando I prodotti scalari con valutazioni del kernel

1) Kernelizzazione di metodi lineari o basati su distanze. Per esempio: Perceptron, KNN

2) Applicazione di algoritmi definiti su vettori a dati NON vettoriali usando kernel definiti per dati non vettoriali

Calcolo della distanza nel feature space

Vogliamo calcolare $d(x,z) = ||\phi(x) - \phi(z)||$

$d(x,z)^2 = ||\phi(x) - \phi(z)||^2$
$= \phi(x) \cdot \phi(x) + \phi(z) \cdot \phi(z) - 2\phi(x) \cdot \phi(z)$

$d(x,z) = \sqrt{k(x,x) + k(z,z) - 2k(x,z)}$

Nota che i valori $\phi(x)$ e $\phi(z)$ non vengono *esplicitamente* utilizzati ma solo *implicitamente*!

Come può essere implementato l'algoritmo KNN?

Tipologie di kernel

Kernel per vettori:

Kernel Lineare: $k(x, z) = x \cdot z$

Kernel Polinomiale: $k(x, z) = (x \cdot z + c)^d$

Kernel Gaussiano (RBF): $k(x, z) = exp(-\gamma||x - z||^2)$

Kernel per stringhe:

L'idea di fondo è quella di contare tutte le sequenze fino ad una certa lunghezza e costruire un vettore delle feature delle occorrenze (esistono algoritmi basati su programmazione dinamica molto efficienti per il calcolo del kernel)

Kernel per alberi:

L'idea di fondo è quella di contare tutti i sottoalberi e costruire un vettore delle feature delle occorrenze (anche qui esistono algoritmi basati su programmazione dinamica molto efficienti per il calcolo di questi kernel)

Kernel per grafi:

Simile a sopra ma, per esempio, contando i cammini in comune

Un ottimo libro: Kernel Methods for Pattern Analysis (John Shawe-Taylor and Nello Cristianini)

Operazioni su kernel

Una combinazione lineare positiva di kernel è un kernel (quindi anche la moltiplicazione per uno scalare positivo):

$$k(x,z) = ak_1(x,z) + bk_2(x,z)$$

Se una sequenza di kernel converge puntualmente ad una funzione k, allora k è un kernel

Una moltiplicazione entrywise di due kernel è un kernel:

$$k(x,z) = k_1(x,z)k_2(x,z)$$

La composizione di kernel è un kernel:

$$k(x,z) = k_3(\phi(x),\phi(z))$$

Combinazioni di kernel

Sommare kernel è un semplice stratagemma per integrare informazione eterogenea:

$$k(x,z) = \sum_{s=1}^{Q} \mu_s k_s(x,z)$$

Dove, p.e. μ è un vettore tale che $\sum_{s=1}^{Q} \mu_s = 1$

IDEA Multiple Kernel Learning: definire algoritmi per apprendere i valori μ_s della combinazione che migliorino le prestazioni di una SVM usando il kernel combinato, rispetto alle SVM ottenute usando I kernel individuali.