Introduzione

Un classificatore derivato dalla teoria teoria di apprendimento statistico di Vapnik. Dopo anni di sviluppo teorico, l'SVM diventa famoso quando, usando immagini come input, da un'accuratezza paragonabile al *SotA neural-network*.
Oggi l'SVM è largamente usata intutti i campi del supervised learning e per la regressione

I nostri obiettivi sulle SVM

1. Max Margin Classifier
   Controllo della complessità del modello con un *approccio di ottimizzazione*, per approssimare direttamente il rischio strutturale della minimizzazione.
2. Kernel
   Usare efficientemente l'espansione di base lineare con il kernel, otteniamo quindi un altro approccio flessibile per il supervised learning non lineare
3. Pratice
   Evitare le tipiche malinterpretazioni nell'uso dell'SVM, sono convizioni troppo ottimistiche sull'overfitting e sull'evitamento del corso dimensionale

1. Margin Example

[image pag.11 of 3.5]
[image pag.12 of 3.5]
Non tutti gli hyperplane che risolvono le task sono uguali, variando l'hyperplane di separazione, varia anche il margine.

> Margin
>
> Il margine è (il doppio) la distanza tra la separazione dell'hyperplane e i punti dei dati più vicini

Intuitivamente: la massima distanza tra i punti dei dati

> Support Vector
>
> [image pag.13 of 3.5]
>
> $$x_i \quad t.c. \quad |w^T x_p + b| = 1$$

Tutti i punti sono classificati correttamente se $(w^T x_i + b) y_i \geq 1 \; \forall$ i dati l.

Toward margin example opt. problem & Canonical rap. of hyperplane and SV

Consideriamo il problema di *apprendere un modello lineare* per una classificazione binaria $\xRightarrow{quindi}$ una funzione $h : \mathbb{R}^N \to \{0, 1\}$, $h(x) = sign(wx + b)$ basata sugli esempi (x_p, y_p) nel training set.

> Training Problem
>
> Trovare (w, b) tali che tutti i punti sono classificati correttamente e i margini sono *massimizzati*

Rappresentazione Canonica dell'hyperplane

$$(x_p, y_p) \text{ classificati correttamente } \forall p$$
$$\iff$$
$$0 \text{ if } y_p = 1 \quad \wedge \quad w^T x_p + b < 0 \text{ if } y_p = 1 \quad \forall p$$

Senza perdita di generalità, è possibile *ridimensionare w* così che i punti più vicini all'hyperplane che soddifa $|w^T x_p + b| = 1 \overset{quindi}{\Longrightarrow}$ al Support vector

$$|w^T x_p + b| \geq 1 \text{ if } y_p = 1 \quad \wedge \quad w^T x_p + b \leq 1 \text{ if } y_p = -1 \, \forall p$$
$$\Longleftrightarrow$$
$$(w^T x_p + b) y_p \geq 1 \, \forall p \quad \leftarrow \quad \text{condizioni: tutti i punti sono classificati correttamente}$$

Two useful fact

> **Definition**
>
> $$margin \propto \frac{2}{|w|} \qquad \text{con } |w|^2 = (w^T w) \text{, norma}$$
>
> $$\text{massimizzare i margini} \quad \Longleftrightarrow \quad \text{minimizzare } |w| \quad \Longleftrightarrow \quad \text{minimizzare } \frac{|w|^2}{2} \qquad \text{massimizzare i ı}$$

> **Definition**
>
> **VC-dim** dell'SVM è *inverso al margine* $\overset{quindi}{\Longrightarrow}$ è decrescente con margini alti
>
> - Controllo della complessità del modello con i margini
> - Lo connettiamo con l'**SLT lecture**

L'**hyperplane ottimo** che massimizza i margini e risolve il problema di training.

Hard margin SVM

> **Training Problem**
>
> Trovare (w, b) t.c. tutti i punti di training sono classificati correttamente e i margini sono massimizzati
> **Forma Primordiale:**
> minimizzare $\frac{|w|^2}{2} \overset{quindi}{\Longrightarrow} w^t w$ t.c. $(w^t x_p + b) y_p \geq 1 \, \forall p = 1, \ldots, l$

É una *minimizzazione diretta della complessità del modello*, tenendo la soluzione nei vincoli. La funzione obiettivo è convessa in w.

DUAL PROBLEM

> **Dual formulation of training**
>
> Massimizzare$_\alpha$ $\sum_i \alpha_i - \dfrac{\sum_{ij} \alpha_i \alpha_j y_i y_j x_i^t x_j}{2} \quad i, j = 1, \ldots, l$
> con $\alpha_i \geq 0, \sum_i \alpha_i y_i = 0$

Dobbiamo trovare un α_p ottimale, con $p = 1, \ldots, l$ (*Moltiplicatore di Lagrange*) con la **programmazione quadratica**. Il fatto che sia convessa implica un'*unica soluzione*, inoltre il costo computazionale scala con l invece che con n (con il numero dei dati invece che con la loro dimensione).
La doppia formulazoine (calcolando i valori di α) ci consente di *mostrare i Support Vectors* e una forma speciale della soluzione.

Solution

Con α (calcolata in forma doppia) possiamo calcolare (w, b)

$$w = \sum_p \alpha_p y_p x_p \mid p = 1, \ldots, l \mid b = y_k - w^t x_k \quad \text{for any } \alpha_k > 0$$

> Definition
>
> $$h(x) = sign(w^T x + b) = sign(\sum_{p=1}^{l} \alpha_p y_p x_p^T x + b) = \boxed{sign(\sum_{p \in SV} \alpha_p y_p x_p^T x + b)}$$

> Definition
>
> 1. $\alpha <> 0 \iff$ Support Vectors
>   ($\alpha_p \neq 0 \rightarrow$ is a support vector)
>   La soluzione è (spesso) sparsa e formulata solo in termini di SVs. L'hyperplane dipende solo dal support vector
> 2. *una forma speciale della soluzione:* non è neanche necessario calcolare w, b esplicitamente per classificare i punti

ROLE OF SUPPORT VECTOR

[image pag.19 of 3.5]

$$h(x) = sign(\sum_{p \in SV} \alpha_p y_p x_p^T x + b)$$

L'hyperplane dipende solo dai support vectors

ROLE OF INNER SUPPORT

$$h(x) = sign(\sum_{p \in SV} \alpha_p y_p \boxed{x_p^T x} + b)$$

I dati vengono inseriti sotto forma di prodotti scalari di coppie di punti

Soft Margin

Gli hard margin possono essere troppo restrittivi per tutti i punti, *alcuni errori possono essere ammessi* per la tolleranza del disturbo dei dati e per fornire un margine maggiore. La soluzione è ammettere errori introducendo slack-variables.

> Primal training problem
>
> minimizzare $\frac{|w|^2}{2} + C \cdot \sum_p \xi_p$
> tale che $(wx_p + b)y_p \geq 1 - \xi_p$ e $\xi_p \geq 0 \; \forall p$

ξ_p positivo indica un errore o margini troppo piccoli, $C > 0$ *guida il numero di errori ammessi* (l'indice di ξ_p è calcolato dal risolutore).
C è un hyperparameter definito dall'utente
C basso $\rightarrow$ sono ammessi troppi errori di training $\rightarrow$ possibile overfitting
C alto $\rightarrow$ non sono ammessi errori di training $\rightarrow$ margini piccoli $\rightarrow$ possibile overfitting

2. kernel

Usare efficientemente le espansioni lineari di basis con kernel, quindi ottentiamo un altro approccio flessibile per il supervised learning non lineare.

Mapping to High-Dimensional Space

Mappare i punti dei dati nello spazio di input *in un grande spazio di caratteristiche*, dove possono essere separate linearmente.
[image pag.28 of 3.5]
I separatori lineari qui corrispondono a un separatore non lineare in uno spazio originale.

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$
$$(x_1, x_2)^T \mapsto (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T$$

Usando LBE $\phi(x)$ invece di x per trattare con task non lineari riferito agli input. Comunque sappiamo che usando uno *spazio delle caratteristiche di grandi dimensioni* può essere *computazionalmente irrealizzabile* e più importante può essere *facilmente portata a overfitting* senza controllarne le dimensioni dello spazio e la complessità del classificatore:

$$h_w(x) = sign(\sum_k w_k \phi_k(x))$$

Vedremo l'**approccio kernel** per gestire (**implicitamente**) lo spazio delle caratteristiche nel *contesto di modelizzazione regolarizzata* (dove la complessità dipende dai margini, non strettamente dalla dimensione di input): così, grazie alla regolarizzazione automatizzata dell'SVM, la *complessità del classificatore può essere mantenuta piccola* nonostante la dimensione del nuovo *spazio delle caratteristiche*.

Use Φ(x) instead of x

In SVM *non è necessario calcolare w* e *il dato entra in forma di* **prodotti scalari** *di coppie di punti*.

$$h_w(x) = sign(\textstyle\sum_k w_k \phi_k(x)) \qquad h(x) = sign(\textstyle\sum_{p \in SV} \alpha_p y_p \boxed{x_p^T x} + b)$$

$$\searrow \qquad \swarrow$$

$$h(x) = sign(\textstyle\sum_{p \in SV} \alpha_p y_p \boxed{\phi^T(x_p)\phi(x))}$$

e non è neanche necessario calcolare direttamente ϕ

> Definition
>
> $$h(x) = sign(\sum_{p \in SV} \alpha_p y_p \boxed{K(x_p, x)})$$
>
> $\xRightarrow{quindi}$ possiamo *implicitamente gestire lo spazio delle ipotesi* con una **funzione Kernel**

KERNEL

> Kernel
>
> Un **kernel** $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ è una funzione tale che qualche spazio di Hilbert X (possibilmente di grande dimensione) e una funzione $\phi : \mathbb{R}^n \to X$ esistono con
>
> $$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$
>
> $\xRightarrow{quindi}$ un kernel è potenzialmente una **scorciatoia** per calcolare il prodotto scalare efficientemente anche in spazi di grandi dimensioni

Usiamo la *funzione k* per calcolare direttamente il prodotto scalare nello **spazio delle ipotesi**.

> Esempio
>
> L'esempio di prima può essere efficientemente calcolato in $\mathbb{R}^2$ invece in $\mathbb{R}^3$:
>
> $$\phi : \mathbb{R}^2 \to \mathbb{R}3$$
>
> $$(x_1, x_2)^T \mapsto (x_1^2, \sqrt{2}x_1x_2, x_2^2)^T$$
>
> $$\underset{\text{Computed in 2-dimension (not in 3)}}{\phi(x_i)^T\phi(x_j) = (x_i^T, x_j)}$$

Kernel popolari ben conosciuti

> Kernel Lineare
>
> $K(x_i, x_j) = x_i^T x_j$
>
> - Si mappa $\Phi : x \to \phi(x)$, dove $\phi(x)$ è x stessa

> Kernel Polinomiale
>
> dell'ordine di $p : K(x_i, x_j) = (1 + x_i^T x_j)^k$
>
> - Si mappa $\Phi : x \to \phi(x)$, dove $\phi(x)$ ha dimensione esponenziale in k

> Kernel RBF (funzione radial-basis) Gaussiano
>
> $K(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}$
>
> - Si mappa $\Phi : x \to \phi(x)$, dove $\phi(x)$ è a infinite dimensioni

Il Kernel RBF è una scelta molto popolare: si noti che ha un hyperplane (σ) molto flessibile, può essere usato per fare confini decisionali intorno a ogni punto del training set.

$$\sigma \text{ basso} \implies \begin{matrix}\text{Kernel a}\\ \text{punta stretta}\end{matrix} \implies \begin{matrix}\text{i pattern sono simili solo se molto vicini}\\ \text{e il classificatore risponde}\\ \text{con la classe del punto più vicino}\end{matrix}$$

Il design del nuovo kernel per tipi di dato speciali è tutt'ora un argomento di ricerca.

SVM completata

Scegliamo il *parametro di compromesso C*, e la *funzione kernel K* (e il suo hyper-parametro). Risolviamo poi l'ottimizzazione del problema per trovare α

- Il costo computazionale scala con l invece che con n (con il numero dei dati e non con la loro dimensione)
- Mudolarità: cambiamo solo il Kernel (con lo stesso risolutore)
  Il modello finale risulta:
  $$h(x) = sign(\sum_{p \in SV} a_p y_p K(x_p, x))$$

2. Pratica - evitare misinterpretazioni

- Ci può essere overfitting senza una cauta selezione degli *hyperparametri* dell'SVM ($C, Kernel, Kernel\ parameters$)
- Trattamento implicito degli spazi di grandi dimensioni nello spazio delle caratteristiche e che non sono nello spazio di input.
- La tecnica di validazione vede molto lontano nella selezione del modello (es. $\text{C, Kernel, kernel hyperparametri}$) e la valutazione del modello può essere usata rigorosamente al meglio

DALLA GUIDA LIBSVM

> *Proponiamo che i principianti provino questa procedura per prima:*
>
> - *Trasformare formato del dato di un software SVM*
> - *Condurre un ridimensionamento semplice sul dato*
> - *Considerare il Kernel RBF $K(x, y) = e^{-\gamma||x-y||^2}$, dove $\gamma = \frac{1}{(2\sigma^2)}$*
> - *Usare la valutazione incrociata per trovare il miglior parametro C e γ*
> - *Usare il miglior parametro C e γ per allenare tutto il training set*
> - *Testare su un set separato ed esterno*

DETTAGLI

- Processare i dati
  - $\{red, green, blue\} \to (0,0,1), (0,1,0), (1,0,0)$ $[1 - of - k]$
  - Ridimensionamento dei valori lineare nel range $[-1, +1] or [0, 1]$

> - $es.\ \frac{v-min}{max-min}$

- Griglia di selezione per la selezione del modello ($es.$ C e γ in RBF)
  - Con una tabella sulle combinazioni di tutte le possibili crescite di valori (esponenziali) per trovare buoni intervalli

$$es. \qquad C = 2^{-5}, 2^{-3}, \dots, 2^{15} \quad \gamma = 2^{-15}, 2^{-13}, \dots, 2^{3}$$

```
Quindi può essere eseguita una ricerca su griglia più precisa
```

Conclusioni

L'SVM è uno strumento avanzato del ML molto utile e popolare. Le *prestazioni* sono *spesso buone, ma non sempre il confronto è favorevole*, riferito ad altri metodi di ML.
Combina l'uso efficiente dell'*espansione di base lineare* col **kernel** con l'**approccio di margine massimo** per combinare modelli flessibili e controllo della complessità.
La modularità del kernel apre nuove possibilità, ammettiamo inoltre l'SVM al fuori della valutazione necessaria.