IIA ML Esercitazioni – parte 2 (appunti in sintesi di esercizi svolti in aula)

***WARNING*: E' estremamente utile per voi stessi provare autonomamente a risolvere gli esercizi (nei file testi-domande) PRIMA di vedere queste soluzioni**

Di seguito alcune note (testo degli esercizi e cenni in sintesi delle soluzioni) di alcuni esercizi svolti nelle lezioni di esercitazione:

1 – Esercizio per validation (1):

Data la tabella con i seguenti valori di accuracy (% di classificazione corretta) per un iper-parametro lambda:

λ	TR	VL	TS
0.5	75	70	70
0.1	80	75	74
0.01	90	70	72

1. In che ordine si usano le porzioni di dati per calcolare i valori in tabella?
2. Quale modello (ossia lambda) si sceglie?
3. Che fenomeni si osservano?

Soluzione (cenni)

1. Per ogni riga si calcola l'accuracy sul TR (si costruisce il modello con quel valore di lambda facendo learning sul training set TR) poi si calcola il valore VL (si verifica l'accuracy del modello con un certo lambda sul validation set VL), solo dopo si stima la capacità predittiva sul test set TS. Oppure : TR per tutti, VL per tutti, poi i TS per tutte le righe.
   (L'accuracy sul TS potrebbe anche calcolarsi solo sul secondo modello dopo aver scelto quello)
2. Il secondo, ha il best su VL (ne' il TR ne' il TS non si possono usare a questo scopo).
3. Il primo è in underfitting (con fitting migliore il VL migliora), l'ultimo tende a overfitting (con fitting migliore il VL è peggiorato, e lo vedo senza guardare il risultato del TS).

2 – Esercizio per validation (2), quello suggerito alla lezione su validation (slide "Exercise"):

Data la tabella con i seguenti valori di accuracy (% di classificazione corretta)

λ	TR	VL	TS
0.5	75	70	70
0.1	80	75	70
0.01	90	70	72

1. In che ordine si usano le porzioni di dati per calcolare i valori in tabella?
2. Quale modello (ossia lambda) si sceglie?
3. Che fenomeni si osservano?

Soluzione (cenni)

1. Come per l'esercizio 1: Si calcola TR poi VL per ogni riga, solo dopo TS della riga.
   Oppure TR per tutti, VL per tutti, poi i TS per tutte le righe.
   (Il TS potrebbe anche calcolarsi solo sul secondo modello dopo aver scelto quello)
2. Il secondo , ha il best su VL (il TR e il TS non si possono usare a questo scopo)
3. Il primo è in underfitting (con fitting migliore il VL migliora), l'ultimo tende a overfitting (con fitting migliore il VL è peggiorato). Notare che il terzo, in accordo alla definizione, e con "il senno di poi", non sarebbe proprio in overfitting (rispetto agli altri due) perché li il TS sarebbe migliore, ma voi "non lo sapete" nel senso che se vedete quello e scegliete il terzo fareste un errore di model selection sul TS.

3 - Descrivere cosa si intende per generalizzazione

Provare un risposta intuitiva in base a quanto presente nel corso, raccogliendo i concetti espressi in varie parti per una sintesi.

4- Descrivere il rapporto tra generalizzazione e il fenomeno dell'overfitting

Provare un risposta intuitiva, in base alle def. di overfitting, o in accordo al VC-bound (in accordo a quanto presentato nel corso). Preferire quella di livello più alto.

5 - Quali parametri regolano il controllo di complessità nei vari modelli visti a lezione?

Provare una risposta in base a quanto presente nel corso, raccogliendo i concetti espressi in varie parti per una sintesi. Occasione per una lettura trasversale dello stesso concetto tra parti diverse del corso. Pensare anche all'effetto del loro valore in termini di underfitting e overfitting, specialmente sul VC-bound.

6- Discutere vantaggi e svantaggi relativi tra coppie di modelli viste durante il corso

Pensarlo in base a criteri informatici classici e a quelli più specifici del ML. Pensare ai criteri e poi discuterli e poi discuterli per i vari modelli a confronto. Esercizio in auto-valutazione di ausilio al ripasso ragionato della materia.

Cenni: Considerare ad esempio costi (in training e test), assunzioni (tipi dei dati, bias induttivi), limitazioni, tolleranza al rumore, tipi di task, interpretabilità, proprietà come modelli di ML (flessibilità, meccanismi che ne regolano la complessità, e per il controllo dell'underfitting-overftting e generalizzazione, …)

Repetita: Testo dell'esercizio 5 di esercitazione 1: – Definire un task (il meteo)

Definire un task e un sistema di apprendimento per previsioni di gradimento sulle condizioni meteo, basato su variabili che misurano l'umidità (in percentuale) e la temperatura (in gradi Celsius) di oggi, che stimi se per voi sia o no una giornata piacevole. Si dispone di una serie di misurazioni delle variabili di interesse nel passato.

a) Definire l'input e il tipo delle variabili
b) Definire l'output e il target e il loro tipo
c) Definire il data set
d) Definire il tipo di task
e) Proporre un possibile modello per la soluzione
f) Descrivere cosa significa fare apprendimento e training per il problema.

Nota: Definire un compito (task) di apprendimento significa definire le risposte alle domande a, b, c, d.

Soluzioni: vedi esercitazione 1

7 -Il meteo: CONTINUO (da un compito passato con qualche variazione)

Sul problema del meteo definito nell'esercizio 5 dell'esercitazione 1:

a. Proporre un possibile modello per la soluzione con SVM:
   1. Scrivere la forma (in esteso) delle ipotesi nella forma *h(**x**)*
   2. Come aumentare la capacità espressiva di *h* nel caso che le relazioni lineari sulle variabili non siano sufficienti per il problema? Esprimere la nuova forma (generale) delle *h(**x**)*
   3. Che rapporto c'è tra il modello del punto precedente e il fenomeno dell'overfitting? E come poterlo gestire?
b. Proporre un possibile modello per la soluzione con un K-NN:
   1. Esprimere l'equazione per la previsione del gradimento
   2. Discutere di nuovo il punto sull'aumento della capacità espressiva per il caso K-NN
   3. Discutere di nuovo il punto sul rapporto c'è tra il modello e il fenomeno dell'overfitting. E come poterlo gestire

Soluzioni (cenni):

a.1 $$h(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x}+b) = sign(\sum_{i=1}^{l}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}+b) = sign(\sum_{i\in SV}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}+b)$$

a.2 Si introduce il kernel

$$h(\mathbf{x}) = sign(\sum_{i\in SV}\alpha_i y_i K(\mathbf{x}_i,\mathbf{x}))$$

a.3 Si discute che in SVM c'è un automatismo per la minore VC dimension collegata con il massimo margine, posto nella funzione obiettivo. Inoltre, la presenza dell'iperparametro C regola il rapporto tra fitting e margine. Un C troppo alto può portare all'overfitting.
Un'altra scelta influente riguarda il tipo di kernel *K* e i suoi parametri. Si può gestire con una attenta model selection su un data set di validation (separato da quello di training) per selezionare questi iperparametri (scegliendo quelli con errore minore sul validation set).

b.1 $$avg_k(\mathbf{x}) = 1/k \sum_{\mathbf{x}_i\in N_k(\mathbf{x})} y_i \qquad h(\mathbf{x}) = \begin{cases} 1 & \_if \quad avg_k(\mathbf{x}) > 0.5 \\ 0 & \_\_\_\_\_otherwise \end{cases} \qquad \begin{matrix} for\ y_i = \{0,1\} \\ (targets) \end{matrix}$$

Oppure nel caso di codifica -1/+1 del target *h(**x**)* emetterà +1 se $avg_k(\mathbf{x}) > 0$, -1 altrimenti.

b.2 Il modello è già flessibile, potendo approssimare decision boundary non lineari con K basso.

b.3 Un K troppo basso può condurre a overfitting. Si può gestire con una attenta model selection su un data set di validation (separato da quello di training) per selezionare il miglior K (quello con errore minore sul validation set).

8 --- COMPITINO anni scorsi ---

Esercizio 8.1: Il cliente del modello lineare

Il cliente della vostra start-up vi propone un problema di classificazione a 3 variabili di ingresso per stimare preferenze (positive o negative). Vi offre 1000 dati etichettati di esempio.

a. Scrivere la $h_a(\boldsymbol{x})$ di un modello lineare per il problema.

   $h_a(\boldsymbol{x}) = sign(w_1x_1 + w_2x_2 + w_3x_3 + w_0)$ *ossia*

   $h(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x} + w_0)$ con $\boldsymbol{x} = [x_1, x_2, x_3]$

b. Dopo avere eseguito un training (a discesa di gradiente), avete ottenuto una *accuracy* di training (ossia sul training set) del 70%. Informato del risultato preliminare, il cliente chiede il modello ottenuto e poi chiede di aggiungere anche il logaritmo delle 3 variabili originarie al modello.
   Scrivere la nuova $h_b(\boldsymbol{x})$ e calcolare il gradiente (esplicitamente, mostrando i calcoli) per i pesi (ossia per uno dei pesi) di queste tre nuove variabili.

   $h_b(\boldsymbol{x}) = sign(w_1x_1 + w_2x_2 + w_3x_3 + w_4log(x_1) + w_5log(x_2) + w_6log(x_3) + w_0)$

   Derivazione standard: si ottiene $\frac{\partial E(\mathbf{w})}{\partial w4} = -\sum_{p=1}^{l} 2(y_p - \boldsymbol{w}^T\phi(\boldsymbol{x}_p)) \cdot \boxed{\log(x_{p,1})}$
   Le altre sono analoghe

c. L'accuracy di training con il nuovo modello h_b diviene del 75%. Il cliente, preso anche il secondo modello, chiede di aggiungere anche il quadrato e il cubo delle 3 variabili iniziali.
   Scrivere la nuova $h_c(\boldsymbol{x})$ e dire cosa cambia nella regola di apprendimento del vostro algoritmo.

   Scrivere $h_c(\boldsymbol{x})$ come prima assommandoci $x_1^2, x_2^2, x_3^2, x_1^3 x_2^3, x_3^3$ con i pesi w_7, w_8 etc.
   Diventano 3+3+3+3+1=13 termini.
   Non cambia nulla, si usa analogamente avendo anche i casi con x_1^2 o x_2^2 etc nel posto riquadrato.

d. L'accuracy di training con la h_c cosi completata raggiunge l'80%. Il cliente si ritiene soddisfatto e vi chiede di dirgli cosa si deve aspettare come accuratezza per l'uso del modello: Cosa rispondete? Cosa rischia nel prendere la h_c?

   Che non avete stimato l'accuratezza per l'uso, vi serve di farlo su un test set (separato dal training set) per una stima appropriata.
   Inoltre non deve scegliere sulla base del risultato di training!
   E facendo così rischia di prendere un modello in overfitting (avendo preso il modello con il minimo errore sul training).

e. Il cliente fa una nuova rivelazione. Oltre ai 1000 dati forniti a voi, ha preparato altri 100 dati etichettati, diversi dai primi, con cui prova i modelli ottenuti ai passi a, b, c. Poiché il miglior risultato di queste prove sui 3 modelli lo ottiene sul modello h_b con il 73% di accuracy, misurata sui questi 100 nuovi dati, sceglie definitivamente quello e vi riferisce che è soddisfatto di avere un modello che fornirà predizioni al 73% di accuracy. Che ne pensate di questa stima? E cosa suggerireste di fare al cliente?

   Non ha agito correttamente poiché ha usato il test set per *selezionare il (miglior) modello* (ossia assumendo il 73% come stima dell'accuracy futura mentre ancora sta operando una model selection)! [Oppure si può anche dire che ha usato il validation set per fare stima dell'accuracy futura].
   Quel set è quindi usato per model selection, i.e. come un *validation set* (fatto secondario: ed è anche un po' piccolo nella versione a 100 dati, voi potreste scegliere sulla base di più dati ad esempio con una k-fold cross-validation per ricavare training e validation set).
   [L'essenziale da far notare nella riposta è che:] Non può usare questi 100 dati per stimare l'accuracy futura. Dovrà usare un nuovo test set (e.g. altri 500 dati nuovi) per stimare l'accuracy in modo appropriato.

Esercizio 8.2: Costruire DT per funzioni booleane

a. Costruire un Decision Tree con l'algoritmo ID3 e l'uso di *Gain(S,A)* che realizzi la funzione boolena x_1 AND x_2 (variabili booleane).
   1. Mostrare i calcoli di *Gain(S,A)*
   2. Mostrare l'albero ottenuto
   3. Disegnare nello spazio delle istanze le zone di classificazione che produce l'albero ottenuto
b. Costruire un Decision Tree con l'algoritmo ID3 e l'uso di *Gain(S,A)* che realizzi la funzione espressa in tabella.
   1. Mostrare i calcoli di *Gain(S,A)*
   2. Mostrare l'albero ottenuto
   3. Disegnare nello spazio delle istanze le zone di classificazione che produce l'albero ottenuto

x_1	x_2	Target
0	0	0
0	1	0
1	0	1
1	1	1

c. Se *n* è la dimensione del vettore di ingresso, nel caso pessimo quanti nodi potrebbero esser aggiunti da ID3 e perché?
d. Sapreste fare un esempio di funzione booleana che corrisponda al caso pessimo del punto c con 2 variabili di ingresso booleane (e mostrando l'albero)?

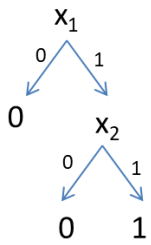
Soluzioni (cenni):

a.1

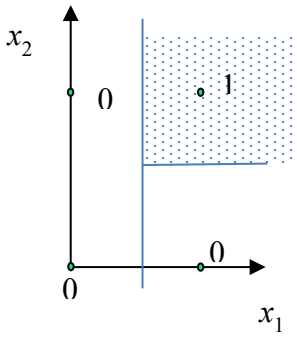
Gain(S, x_1)= E(S) -2/4 (0) -2/4(1)= E(S)-1/2
Gain(S, x_2)= E(S) -2/4 (0) -2/4(1)= E(S)-1/2

Scelta indifferente, diciamo x_1

a.2



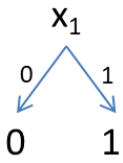
a.3



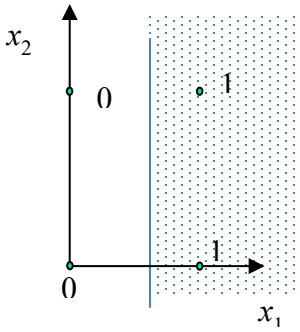
Nota: I riquadri vogliono dare solo un'idea *grafica* della ripartizione (a confronto di quanto farebbe un decisore della classe dei *linear model* che avete già conosciuto) perché con valori booleani DT fa decisioni sui valori stessi, non "*a zone*" (come farebbe per valori continui).

b.1 Sviluppate il calcolo analogamente ottenendo:
Gain(S, x_1)=E(S)-0 = 1 e Gain(S, x_2)=E(S)-2/4-2/4=E(S)-1= 0 → si sceglie x_1

b.2

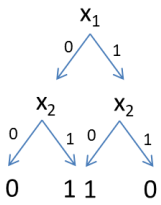


b.3



c. Nel caso pessimo si dovranno inserire tutte le variabili/attributi in un albero bilanciato (per aver ripartito sempre in modo a massima entropia, con pari esempi positivi e negativi), con ogni nodo (di decisione) che ha 2 figli →2^n -1→O(2^n) nodi (le foglie sono output e non le contiamo).

d. Esempio in tabella:
e questo l'albero:



x_1	x_2	Target
0	0	0
0	1	1
1	0	1
1	1	0

Un ultimo suggerimento: evitare *underfitting* o *overfitting* della soluzione di *QUESTI* esercizi o, peggio, da altri da fonti diverse da queste (e.g. prive di affidabilità scientifica) .

.