Appunti di Machine Learning

4 ottobre 2022

Rosso Carlo

Contents

1 Decision Trees

L'apprendimento mediante alberi di decisione è uno dei metodi più utilizzati e pratici per l'apprendimento induttivo. L'apprendimento mediante alberi di decisione cerca in uno spazio di ipotesi completamente espressivo. Il loro bias è la preferenza per alberi piccoli rispetto ad alberi grandi.

Gli alberi di decisione approssima una funzione discreta, dove la funzione appresa è un albero di decisione; possono anche essere rappresentati come una serie di *if-then rules*, per migliorare la comprensione. Ciascun nodo dell'albero coincide con il testi di un attributo, e ciascun arco, che connette i figli, indica il valore dell'attributo. I nodi foglia indicano la classe dell'istanza. In effetti si tratta di una tecnica di classificazione. Fondamentalmente, ciascun nodo dell'albero rappresenta una condizione ("if"), e quindi una congiunzione ("and") di condizioni; mentre ciascun arco rappresenta un il valore delle condizioni possibili ("is a"), e quindi una disgiunzione ("or"). Per questo motivo, si capisce bene che questo approccio è possibile solamente se le variabili sono discrete.

1.1 Algoritmo ID3

L'algoritmo ID3 è l'algoritmo di base, da cui partiamo. In questo caso l'albero è costruito a partire dalla root e quindi ha una costruizione bottom-up. Funziona testando l'attributo che contiene la maggior informazione. C'è quindi bisogno di chiarire come selezionare l'attributo più utile per classificare gli esempi. L'*information gain* è una proprietà statistica che misora quanto un attributo separa gli esempi di allenamento in base alla loro classe di appartenenza.

Def. 1.1 *Sia S una collezione, contenente esempi positivi e negativi, l'entropia di S relativa alla rappresentazione booleana è così definita:*

$$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i \tag{1}$$

dove p_i è la proporzione di esempi in S che appartengono alla classe i. L'entropia vale 0 se tutti gli esempi sono dello stesso tipo, e 1 se gli esempi sono divisi equamente tra le classi. L'entropia è il numero di bit necessari per rappresentare un esempio di S, per questo motivo è utilizzato il logaritmo in base 2.

Def. 1.2 *Sia l'entropia una misura di impurità in una collezione di esempi. Possiamo ora misurare quanta informazione contiene ciascun attributo per classificare gli esempi. L'information gain, banalmente, è la partizione di esempi secondo l'attributo. Più precisamente, l'information gain, $G(S, A)$ di un attributo A, relativo ad una collezione di esempi S, è così definita:*

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{2}$$

dove $Values(A)$ sono tutti i possibili valori che può assumere un attributo A, e S_v è il suttoinsieme di S per cui l'attributo A assume valore v.

- L'information gain è proprio la misura usata dall'algoritmo ID3 per effettuare la scelta greedy e decidere quale attributo controllare per primo;
- L'albero è costruito in modo tale che lo stesso attributo compare solo una volta in ciascun percorso dell'albero decisionale; Ciascun processo continua per ciascun nuovo nodo foglia fino a che o tutti gli attributi sono stati inclusi nel percorso, oppure tutti gli esempi associati alla foglia hanno medesima classificazione;
- L'algoritmo ID3 è uno spazio completo di funzioni finite a valori discreti;
- L'ID3 non permette il backtracking, le decisioni di costruzione dell'albero iniziale permangono valide anche se gli esempi di training cambiano. L'albero cresce per ottenere una soluzione ottima locale, ma non globale. La soluzione in merito a questo è utilizzare l'algoritmo di *post-pruning*, per cui l'albero viene come ribilanciato.

Una buona approssimazione dell'ID3 è la seguente: alberi più piccoli sono preferiti ad alberi più grandi. Sono preferiti gli alberi che pongono gli attributi con maggior information gain vicino alla radice.

Per cui il bias è dato dalla sua strategia di ricerca. In genere questo tipo di bias è chiamato preference bias.

2 lezioni

- **hold-out**: sono trattenuti v esempi dal training set. Questi sono utilizzati come validation set, ovvero per misurare l'accuratezza del modello allenato;
- **k-fold cross validation**: si suddivide il training set in k partizioni, di cui una viene utilizzata come validation set, mentre le altre k-1 vengono utilizzate per l'allenamento. Questo processo viene ripetuto k volte, in modo da utilizzare ogni volta una partizione diversa come validation set. L'accuratezza del modello viene calcolata mediando le k misurazioni ottenute;

3 metriche

- **accuracy**: è la percentuale di esempi correttamente classificati:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{P + N} \tag{3}$$

  dove T sta per true, F per false, P per positive, N per negative;

- **precision**: è la percentuale di esempi positivi classificati rispetto ai TP e ai FP:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

- **recall**: è la percentuale di esempi positivi classificati rispetto ai TP e ai FN:

$$recall = \frac{TP}{TP + FN} \tag{5}$$

- **F1-score**: è la media armonica tra precision e recall:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{6}$$

- **precision-recall curve**: è una curva che rappresenta la precision in funzione del recall. Questa curva è utile per visualizzare il comportamento di un classificatore rispetto a diversi valori di soglia. In particolare, è possibile scegliere il valore di soglia che massimizza la precision, oppure quello che massimizza il recall. Inoltre, è possibile calcolare l'area sotto la curva, che è un'indicazione della qualità del classificatore;
- **ROC curve**: è una curva che rappresenta la percentuale di TP in funzione della percentuale di FP. Questa curva è utile per visualizzare il comportamento di un classificatore rispetto a diversi valori di soglia. In particolare, è possibile scegliere il valore di soglia che massimizza la precision, oppure quello che massimizza il recall. Inoltre, è possibile calcolare l'area sotto la curva, che è un'indicazione della qualità del classificatore. ROC curve si tratta dell'integrale della *precision-recall curve*;
- **average precision**: è una misura che combina la recall e la precisione per risultati già ordinati in base all'accuratezza del modello. In particolare, è definita come:

$$AP = \frac{1}{n} \sum_{i=1}^{n} precision_i \cdot rel_i \tag{7}$$

  dove n è il numero di esempi positivi, $precision_i$ è la precisione fino all'i-esimo esempio positivo, e rel_i è una funzione che vale 1 se l'i-esimo esempio è rilevante (positivo), 0 altrimenti.

Diagnosticare bias o varianza: si possono utilizzare le curve di apprendimento. Le curve di apprendimento habno sulle y la misura dell'errore, e sull'asse x un paremetro che indica la complessità del modello che stiamo allenando. Mettendo assieme le curve di apprendimento di training e validation, si può ottenere una stima dell'errore di generalizzazione (graficamente). La differenza tra le due curve è il valore che ci interessa.

4 Support Vector Machine

Le SVM sono modelli di machine learning con supervisono. Sono non-probabilistici e sfruttano la classificazione lineare. La funzione per aggiornare i pesi è:

$$\Theta = min_{\Theta} \frac{1}{\lambda} J(\Theta) + \frac{1}{2} \sum_{j=1}^{n} \Theta_j^2 \tag{8}$$

tale che:

$$J(\Theta) = \sum_{i=1}^{m} cost(h_{\Theta}(x^{(i)}), y^{(i)}) \tag{9}$$

dove *cost* è la funzione di costo così definita:

$$cost(h_{\Theta}(x^{(i)}), y^{(i)}) = \max(0, 1 - y^{(i)} \cdot (\Theta^T x^{(i)})) \tag{10}$$

tale che $y^{(i)} \in \{-1, 1\}$. Notiamo dalla funzione di costo che nel momento in cui $sgn(\Theta^T x^{(i)}) = y^{(i)}$ e $|\Theta^T x^{(i)}| \geq 1$, allora $cost = 0$. In questo caso il margine che separa le due class ha dimensione $M = \frac{2}{||\Theta||}$.
Non è detto che tutti i punti siano linearmente separabili, per questo motivo introduciamo la tecnica del soft-margin, che permette di avere un margine maggiore, infatti, per tutti quei punti che si trovano sul margine o oltre di esso è introdotto qualche parametro di penalità, che permette di individuare un iperpiano per le due classi più accurato, appunto.

4.1 Kernel

Le SVM sono modelli molto comodi per operare anche classificazioni non lineari. Infatti, è possibile applicare una kernel function, che mappa i dati in uno spazio di dimensione maggiore. Aumentare le dimensioni potrebbe permettere di rendere i dati linearmente separabili, tuttavia è necessario indovinare la funzione di mappatura corretta.

5 non-parametric models

Si tratta di modelli che hanno poche features. I modelli non parametrci più comuni sono del tipo nearest neighbor. Il modello non è allenato: la prediction è calcolata in base all'intero dataset, per questo richiede molta memoria ed è lenta. I modelli non parametrci sono in grado di approssimare ogni funzione: in effetti viene mappato l'intero spazio degli esempi, nel senso che si costruisice una sorta di mappa che approssima gli ipervolumi conenenti gli esempi e a ciascuno di essi è assegnata una classe. L'esempio di test è classificato riconducendolo a esempi del training set.

6 neirest neighbor

Ciascun esempio è codificato come un vettore in un iperspazio. Si cerca l'elemento nel training set più vicino all'esempio di test utilizzando qualche distanza (euclidea, di manhattan, etc). I limiti degli iper-volumi non sono calcolati esplicitamente, ma possono essere dedotti:

Def. 6.1 (diagramma di Voronoi) *L'iperspazio è diviso in ipervolumi, in cui ciascuna ipersuperficie è equidistante, per qualche distanza, tra due esempi di classi diverse.*

cons:

- computazionalmente costoso;
- è sensibile a punti anomali;

7 k-nearest neighbor

Al posto di considerare l'esempio più prossimo, sono cercati i k esempi più prossimi ed è applicato un sistema di votazione (conteggio) per determinare la classe dell'esempio di test. Regola di decisione: si assegna la classe di maggiornanza tra le classi dei k esempi più vicini.
NB k troppo piccolo rende il modello molto sensibile a esempi anomali. Un k troppo grande classifica qualunque esempio come appartenente alla classe più frequente. Si può utilizzare la tecnica della cross-validation per determinare il valore ottimale di k. In genere $k < \sqrt{m}$, dove m è il numero di esempi nel training set. Bias induttivo: assume che esempi vicini appartengano alla stessa classe, può essere necessario un lavoro di preprocessing per normalizzare i dati.

8 decision tree

Ciascun nodo corrisponde ad un test su un attributo. Il branching è determinato dal valore dell'attributo. Le foglie rappresentano l'output, ovvero il risultato. Per cui, in pratica a seconda del test effettuato il dataset è diviso in subset che va formare il dataset del nuovo nodo figlio, fino a che non si individua una classe comune a tutti gli esempi del subset. Questo metodo è chiamato *recursive partitioning*.
Un albero di decisione rappresenta una funzione booleana: ciascun percorso dalla root alla leaf è rappresentato da congiunzioni di test sugli attributi. Percorsi che arrivano alla stessa classe sono combinate con disgiunzioni. Questa struttura da origine alla *disjunctive normal form*. Per rappresentare l'albero per intero, si ottiene una di queste funzioni per ogni classe.

9 ID3

Ha in input S, A, dove S è il training set e A è l'insieme degli attributi.

1. crea la radice r;
2. se $\forall s \in S, \exists c \in C, class(s) = c$ allora assegna r a c e termina;
3. se $A = \emptyset$ allora assegna r alla classe più frequente in S e termina;
4. seleziona $a \in A$ che massimizza l'information gain;
5. partiziona S in $S_1, \ldots, S_n$ in base al valore di a;
6. crea un nodo sull'attributo a e lo aggiunge ad r;
7. ripete per ciascun sottoinsieme così generato, utilizzando $A - \{a\}$.

Spighiamo che cos'è l'information gain, che è il cuore dell'algoritmo.
L'entropia è:

$$H(S) = -\sum_{c \in C} p(c) \log_2 p(c) \tag{11}$$

dove $p(c)$ è la probabilità che un esempio di S appartenga alla classe c, ovvero $p(c) = \frac{|S_c|}{|S|}$, dove $S_c = a | a \in c$ per qualche classe $c \in C$.
L'entropia misura la distribuzione delle classi in S. Per cui se in S le classi sono equidistribuite, l'entropia è massima e vale 1. Questo vuol dire che è più difficile assegnare una classe al dataset S e occorre dividerlo in subset.

$$IG(S, a) = H(S) - \sum_{t \in T} p(t) H(t) = H(S) - H(S|a) \tag{12}$$

dove T è l'insieme dei subset che si ottengono dividendo S sull'attributo a. L'informazion gain compara l'entropia di S con l'entropia dei subset $S_1, \ldots, S_n$. Vuol dire che si ha information gain massima, quando con l'attributo considerato si riesce a dividere il dataset in subset che hanno entropia minimale.
Inductive Bias: sono preferiti alberi piccoli che pongono gli attributi con maggior information gain in alto.

10 Random Forest

Si tratta di un modello composto da diversi decision tree. Il risultato restituito è la classe più frequente tra i risultati dei singoli alberi.