

Classificazione Supervisionata: caso binario - 25/11

Ci concentriamo nel contesto della classificazione supervisionata sul caso binario, che è un caso più maneggevole e che quindi ci possiamo anche trovare all'esame.

Il caso binario è più semplice ma è anche il più trattato e il più popolare, in alcuni testi si fa la classificazione e poi la regressione logistica si concentra solo sul caso binario.

Tutto ciò che vediamo comunque si generalizza ad un numero di classi arbitrario.

Visto che è caso binario partiamo dicendo che

$$Y \in \{-1, +1\}$$

Le label come sempre si possono chiamare come si vuole, per noi è comodo chiamarle -1 e 1 perché poi ci servono questi valori nelle formule, quindi le label hanno un significato fisico? No, le potevamo chiamare A e B ma poi dovevamo scriverci da qualche parte che dove c'era A bisognava mettere -1 e dove c'era B +1.

Quando vogliamo applicare il criterio MAP, essenzialmente è come fare il rapporto delle due posterior confrontato con 1.

Essenzialmente dovremmo fare il rapporto delle posterior confrontato con 1, o il logaritmo del rapporto confrontato con 0.

$$s_B(x) \triangleq \ln \frac{P_B(+1|x)}{P_B(-1|x)} = \ln \frac{P_B(+1|x)}{1 - P_B(+1|x)} \quad (1)$$

Regola di decisione $s_B(x) \underset{-1}{\overset{+1}{\geq}} 0$ STATISTICA DI DECISIONE

L'oggetto \ln del rapporto delle posterior prende il nome di **Statistica** (funzione dei dati quando sono oggetti aleatori) **di Decisione** e porta ad una **Regola di Decisione**.

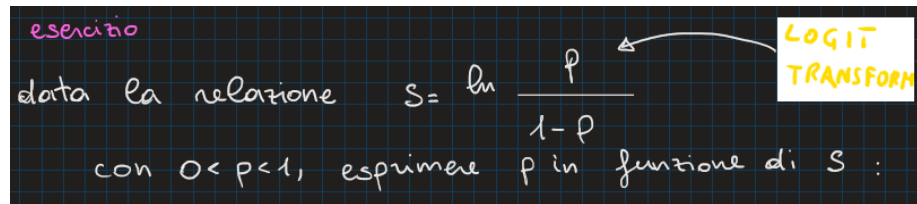
La ragione per la quale introduciamo questa statistica di decisione richiede qualche riflessione.

Quando abbiamo una PMF su h valori quanti gradi di libertà abbiamo? La somma dei valori della PMF fa 1 quindi per forza uno dei valori è fissato se sono noti gli altri, di conseguenza ha senso che la

statistica di decisione in caso binario è una sola funzione, il rapporto tra le due funzioni, ma una delle due si può scrivere l'altra meno 1 e quindi c'è una sola funzione.

Se avessimo avuto h classi avremmo avuto $h - 1$ di questi confronti da fare, invece ne abbiamo 1 perché il problema è binario ($h=2$). Per dire quale di h classi è più probabile ci vogliono tanti confronti, non ne basta 1.

Esercizio



esercizio

data la relazione $s = \ln \frac{p}{1-p}$ con $0 < p < 1$, esprimere p in funzione di s :

LOGIT TRANSFORM

Con s e p che sono numeri generici senza significato fisico.



In gergo quando le varie componenti hanno questa forma tutto il pezzo a destra prende il nome di **Logit Transform**.

Abbiamo s in funzione di p .

Vogliamo trovare p in funzione di s .

$$e^s = \frac{p}{1-p} \Leftrightarrow e^s - pe^s = p \Leftrightarrow p = \frac{e^s}{1+e^s}, \quad 1-p = \frac{1}{1+e^s}$$

$$\text{è conveniente scrivere } p = \frac{1}{1+e^{-s}}$$

Abbiamo in questo modo ottenuto sia p che $1 - p$ entrambi in una forma molto simile l'una all'altra.

Ora applichiamo queste formule al problema originario.

Otteniamo la **SOFTMAX FORM**

$$P_{\beta}(+1|x) = \frac{1}{1 + e^{-s_{\beta}(x)}}$$

$$P_{\beta}(-1|x) = \frac{1}{1 + e^{s_{\beta}(x)}}$$

Si chiama Softmax form perché se lo vedo come un rapporto di esponenziali, quando abbiamo un esponenziale diviso una somma di altri termini domina l'esponenziale con esponente più grande quindi è come se stessimo facendo una sorta di massimo.

Ora che abbiamo le due formule per +1 e -1 possiamo scrivere la formula per la generica Y.

$$P_{\beta}(y|x) = \frac{1}{1 + e^{-y s_{\beta}(x)}}$$

Questi passaggi ci hanno spiegato perché ci conveniva chiamarli +1 e -1 ma ancora non spiega a cosa ci serve tutto questo.

Per un problema generico di classificazione binaria la posteriori, sia vera che parametrizzata (dipende da chi è s_{β}), si può scrivere con la forma che abbiamo scritto.

Non abbiamo fatto nulla, abbiamo solo **trovato una scrittura alternativa della p_{β}** .

Quindi il β ottimo diventa il seguente:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \mathbb{E}_P \left[\ln \left(1 + e^{-y s_{\beta}(z)} \right) \right]$$

Il β empirico invece diventa:

$$\hat{\hat{\beta}} = \underset{\beta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n \ln \left(1 + e^{-y_i s_{\beta}(x_i)} \right)$$

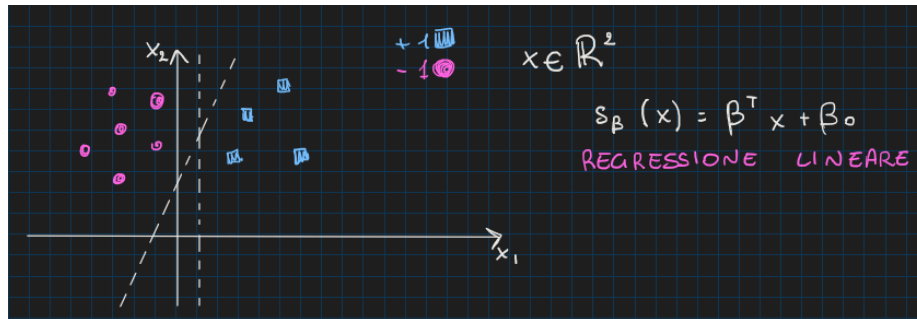
A cosa serve tutto questo?

Tutto quello che abbiamo fatto fino ad ora potrebbe essere bypassato, già la forma precedente andava bene.

Nelle formule precedenti per usarle al calcolatore avremmo dovuto esplicitare p_β , con queste nuove formule invece dobbiamo esplicitare la statistica di decisione s_β .

Ma comunque la definizione di s_β contiene p_β quindi cos'è cambiato? Facciamo un esempio.

Immaginiamo di avere le x che sono features che appartengono ad uno spazio bidimensionale, il nostro training set è fatto da dati con etichetta.



Se ci venisse chiesto come separare le due classi si potrebbe pensare ad una retta che rappresenta la soglia, questa è una regola lineare.

Ora pensiamo in un altro modo, ci viene chiesto di costruire la posteriori, avremmo dovuto calcolare la probabilità che un certo x che sta in un certo punto provenga dalla classe rosa e quella che provenga dalla classe blu.

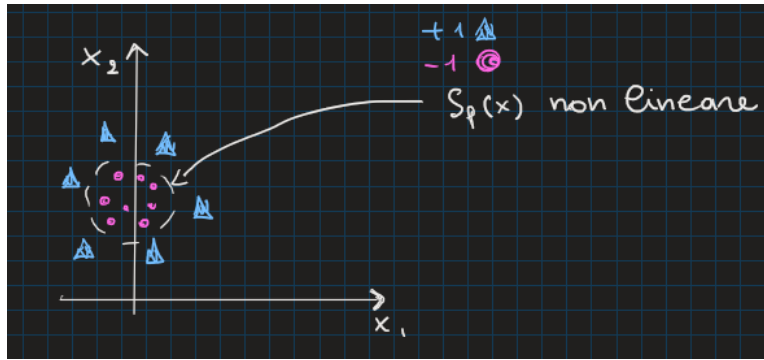
Costruire la posteriori, dovendo assegnare queste probabilità, è più innaturale, è spesso più naturale stabilire una regola che dice stai di qua o stai di là, quella che abbiamo definito Regola di Decisione.

Quindi la ragione per la quale siamo arrivati ad una nuova formulazione è solo strumentale, talvolta è più facile immaginare più o meno qual è il criterio con il quale distinguerò le classi.

Poi è utile capire che questa formula è equivalente, è un mapping 1 a 1 con la posteriori, quindi non abbiamo perso nè guadagnato nulla in principio a cambiare formulazione ma ci abbiamo guadagnato qualcosa se, come in questo caso, ci viene semplice la regola per distinguere le due classi, la formula per separare.

Ovviamente non tutti i problemi sono così, non tutte le classi sono linearmente separabili.

Ad esempio in questo caso la s_β non sarà lineare.



s_β

Non è detto che spesso si è in grado di fare un'analisi di questo tipo, sicuramente qualche plot dei dati è utile farlo, se sono dati ad alta dimensionalità si plotteranno sottoinsiemi delle componenti per capire se ad esempio la separabilità lineare ha senso e se non lo ha per vedere se ci sono pattern particolari. Questo come prima guida per cercare di capire come istruire il classificatore.

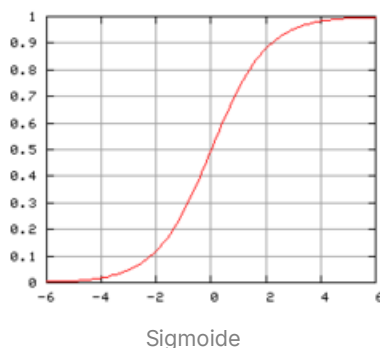
Riassumendo, la classificazione binaria la possiamo fare, definendo la Logit Transform, mettendo la posterior nella forma che abbiamo raggiunto, il ruolo dell' s_β che abbiamo trovato è un ruolo di comodo perché laddove troviamo un modo comodo di separare le classi allora abbiamo una regola 1 a 1 che dice "se hai la regola costruisci anche la posteriori", se non si trova la regola brancolo nel buio uguale a come sarebbe stato se non ci fossimo calcolati s_β .

In molti testi per farla breve la classificazione binaria viene presentata così, si prende la regola della regressione lineare ($s_\beta(x) = \beta^T x + \beta_0$) e siccome esce un numero tra $-\infty$ e $+\infty$ e a noi serve un numero tra 0 e 1 (perché deve essere una probabilità) prendiamo la formula della p_β che sta nel rettangolo rosa questa e la si applica visto che porta ad ottenere un numero tra 0 e 1.

Questo però è assurdo, il processo è al contrario, la classificazione è un mondo a parte rispetto alla regressione, è impensabile pensare che si fa la regressione e si usa una formula per aggiustare il risultato. Il principio di partenza è lo stesso, cioè si vuole minimizzare un costo, il costo però è diverso,

si vede la probabilità, non l'errore quadratico. Da dei calcoli si arriva al fatto che bisogna minimizzare la cross-entropy e poi si mette in una forma che ricorda la regressione.

Questa cosa di partire dalla regressione lineare è assolutamente **ingiustificata** e dire una cosa del genere è un **errore enorme all'esame**, va bene per una persona che deve capire questo concetto in 15 minuti perché poi deve fare altro.



La formula nel rettangolo è spesso chiamata Sigmoide, perché trasforma un numero nell'intervallo $[-\infty, +\infty]$ in un numero nell'intervallo $[0, 1]$.

Supponiamo di essere arrivati alla fine di questi ragionamenti, abbiamo scelto una regola lineare, la forma la abbiamo, s_β ha preso forma, **cosa manca per risolvere il problema?**

Supponiamo di sostituire la s_β individuata nella seguente formula:

$$\hat{\beta} = \arg \min_{\beta} \frac{1}{n} \sum_{i=1}^n \ln(1 + e^{-y_i s_\beta(x_i)})$$

Cosa ci manca per risolvere il problema?

Il calcolatore deve trovare il β migliore, quello che minimizza, come fa?

Si usa la derivata e la si eguaglia a 0. Qui non si può fare perché esce trascendente, quindi è solo per l'MSE che si fa, non esistono casi ragionevoli in cui si risolve carta e penna.

Quindi lo deve fare il calcolatore e servono degli algoritmi per trovare il minimo, l'approccio forza bruta può essere fattibile se β è scalare ma sicuramente non lo possiamo fare se β è in tante dimensioni.

Ci sono, fortunatamente, algoritmi molto furbi per farlo, sono algoritmi che hanno a che fare con la teoria dell'ottimizzazione, cioè con la teoria che si occupa di trovare i minimi o i massimi delle funzioni.

Il primo algoritmo interessante è l'algoritmo del gradiente.

Vediamo elementi di base dell'ottimizzazione e vogliamo imparare il necessario per eseguire questo algoritmo importantissimo.