Università degli Studi di Milano
Facoltà di Scienze e Tecnologie
Corso di Laurea Triennale in Informatica

IMPLEMENTAZIONE E CONFRONTO DI ALGORITMI DI OTTIMIZZAZIONE PER L'APPRENDIMENTO DI INSIEMI FUZZY

Relatore: Prof. Dario MALCHIODI

Correlatore: Prof. Alberto CESELLI

Tesi di:
Giacomo Intagliata
Matricola: 873511

Anno Accademico 2020-2021

Indice

Introduzione

Insiemi fuzzy e come è strutturata la tesi

Capitolo 1

Induzione insiemi fuzzy

1.1 Logica fuzzy

La logica fuzzy è una logica in cui possiamo attribuire a ciascuna proposizione un grado di verità compreso tra 0 e 1. Questa è un'estensione della logica booleana. Nella logica booleana le variabili possono assumere solamente i valori *vero* e *falso*, generalmente denotati con 0 e 1. Nella logica fuzzy le variabili possono assumere valori compresi nell'intervallo [0,1], dove gli estremi corrispondono rispettivamente a *falso* e *vero*. Il valore assunto è chiamato *valore di appartenenza* o *grado di verità*, e denota quanto è vera una proprietà, che può essere oltre che vera o falsa, anche parzialmente vera e parzialmente falsa.

Per capire meglio questo concetto possiamo fare qualche esempio che rispecchia la vita reale, dove molte cose non vengono valutate in maniera netta e non sempre è tutto o niente:

- l'acqua che esce dal rubinetto è *fredda* con un grado di verità 0.4;
- un diciottenne è *giovane* con un grado di verità 0.8;
- una persona di 180cm è *alta* con un grado di verità 0.7;

1.2 Insiemi Fuzzy

Quando x appartiene ad un insieme e non è più valutato in termini di logica classica ma di logica fuzzy otteniamo l'insieme fuzzy. Formalmente il grado di verità è determinato da un' opportuna *funzione di appartenenza* '$\mu_f(x) = \mu$'. La x rappresenta i predicati da valutare ed appartenenti ad un insieme di predicati X. La μ rappresenta il valore di appartenenza del predicato all'insieme fuzzy considerato ed è un valore

reale compreso tra 0 e 1. Per x pari a 1 l'elemento è certamente incluso nell' insieme, per x pari a 0 l'elemento non è per niente incluso nell' insieme (questi due valori corrispondono alla teoria classica degli insiemi), mentre per tutti i valori compresi tra 0 e 1 l'appartenenza può essere più o meno forte.

Consideriamo per esempio lo spazio U, come l' universo delle persone e un insieme A che include tutte le persone giovani. Per ognuna di queste categorie in U:

- ottantenne;
- ventenne;
- neonato;

possiamo definire un grado di appartenenza all'insieme A. Per esempio:

- l'ottantenne appartiene ad A con un valore pari a 0.1;
- il ventenne appartiene ad A con un valore pari a 0.8;
- il neonato appartiene ad A con un valore pari a 1;

Possiamo formalizzare questo ragionamento. Dato un universo U e una funzione di appartenenza, che definisce il grado di appartenenza ad A per ogni elemento nell'universo dell'insieme U :

$$\mu_A : U \to [0, 1]$$

si definisce l'insieme fuzzy A l'insieme delle coppie:

$$A = (u, \mu_A(u)) \quad | \quad u \in U$$

Operazioni tra insiemi fuzzy

Gli insiemi fuzzy non godono di relazioni di univocità e biunivocità far gli elementi di insiemi diversi. Pertanto, gli insiemi fuzzy sono un' estensione degli insiemi della teoria classica, ovvero sono una teoria che allarga ma è inclusa in quella degli insiemi. Sugli insiemi fuzzy valgono quindi gli operatori insiemistici: unione, intersezione e complementare. Tali operazioni vengono definite attraverso le relative funzioni di appartenenza.

Unione

Dati due insiemi fuzzy A e B di U, la loro unione è indicata da $A + B$ o $A \cup B$ e risulta anche essa sottoinsieme fuzzy di U, con una funzione di appartenenza pari a:

$$\mu_{A+B}(u) = max\{\mu_A(u), \mu_B(u)\} \qquad u \in U$$

$A \cup B$ è quindi rappresentabile come :

$$A \cup B = \int_U \frac{max(\mu_A(u), \mu_B(u))}{u}$$

Intersezione

Dati due insiemi fuzzy A e B di U, la loro intersezione è indicata da $A * B$ o $A \cap B$ e risulta anche essa sottoinsieme fuzzy di U, con una funzione di appartenenza pari a:

$$\mu_{A*B}(u) = min\{\mu_A(u), \mu_B(u)\} \qquad u \in U$$

$A \cap B$ è quindi rappresentabile come :

$$A \cup B = \int_U \frac{max(\mu_A(u), \mu_B(u))}{u}$$

Complemento

Il complemento di un sottoinsieme fuzzy A di U è un insieme fuzzy indicato con $\neg A$ e con funzione di appartenenza:

$$\mu_{\neg A}(u) = 1 - \mu_A(u) \qquad u \in U$$

1.3 Machine Learning

Il Machine Learning è una branca dell' informatica nella quale in una macchina si predispone l' abilità di apprendere qualcosa dai dati in maniera autonoma.

Il Machine Learning si riferisce al processo tramite cui i computer sviluppano il riconoscimento dei modelli, ovvero la capacità di apprendere continuamente ed effettuare previsioni utilizzando i dati per poi apportare modifiche in autonomia, senza essere programmati specificatamente per farlo. Il Machine Learning automatizza in modo efficiente il processo di costruzione di modelli analitici e consente alle macchine di adattarsi a nuovi scenari in modo autonomo.

Gli algoritmi di Machine Learning differiscono tra loro a secondo dell'approccio, dei dati che utilizzano, che producono, e nel tipo di problema da risolvere. Possiamo avere degli approcci di diverso tipo:

- supervisionato;
- non supervisionato;
- semi-supervisionato;
- apprendimento con rinforzo.

Approccio Supervisionato

L' approccio supervisionato è una tecnica di apprendimento automatico che mira ad istruire un sistema informatico in modo da consentirgli di elaborare automaticamente previsioni sui valori di uscita di un sistema rispetto ad un input sulla base di una serie di esempi ideali, costituiti da coppie di input e di output, che gli vengono inizialmente forniti. Questa tecnica lavora su un insieme di dati associati ad etichette definite dall'utente. Sapendo che ogni dato è associato a un'etichetta si vuole arrivare a cogliere la relazione che vi è tra dati ed etichette, così da poter predire le etichette a partire dai dati, anche lavorando con dati non visti durante la fase di apprendimento. Questa tecnica può fornire due diversi tipi di risultati: discreti o continui.

Per fare un esempio, consideriamo delle diagnosi mediche fatte su una serie di pazienti.Analizzando le diagnosi un medico è in grado di definire se il paziente è in salute o meno. Da qui possiamo estrapolare quindi due etichette differenti per il nostro caso: "in salute" e "malato". Fornendo come input a un classificatore questo insieme di dati con le rispettive etichette appena definite, il calcolatore, tramite un'algoritmo supervisionato, sarà in grado di fornire delle predizioni sulla possibile etichetta da attribuire ad ogni nuova diagnosi.

Però è necessario che vi siano molti dati da analizzare, perché quando questo non succede la capacità di predizione del sistema che si ottiene è spesso di bassa qualità.

In questo esempio abbiamo utilizzato solamente le classi "in salute" e "malato", ma ad esempio se volessimo quantificare l'aspettativa di vita non è più possibile ricorrere ai classificatori. Nasce quindi la necessità di passare da un valore discreto ad uno continuo, pertanto si utilizzano i *regressori* che costituiscono un modello per predire i valori continui. Ad esempio, potremmo voler quantificare, data una specifica diagnosi, il tempo di guarigione per un paziente malato che per definizione si definisce su una scala di numeri continui.

Per riuscire ad eseguire una predizione su dati nuovi, mai visti prima, con la maggior precisione possibile, dobbiamo assicurarci che il modello produca stati lontani sia dal sovra-adattamento (*overfitting*) che dal sotto-adattamento (*underfitting*).

L'*overfitting* si verifica quando il modello tende ad adattarsi in maniera eccessiva ai dati che gli sono stati forniti per allenarsi, non permettendo la generalizzazione a nuovi insiemi di dati. L'*underfitting* invece, si verifica nel caso contrario,ovvero quando il modello si basa su schemi troppo semplici e poco robusti, il che comporta la definizione di regole con scarsa qualità per la predizione di nuovi elementi.

Quello che vogliamo è trovare un modello che si posizioni tra l' *overfitting* e l' *underfitting*. Il numero di parametri e variabili utilizzate sono aspetti importanti da considerare.

Con un modello troppo semplice si rischia di utilizzare dati poco significativi, con un troppo complesso rischiamo invece di utilizzare troppe variabili e non concentrarci su quelle davvero significative.

Bisogna trovare lo *Sweet spot*, ovvero il punto che rappresenta il miglior compromesso tra precisione nella predizione e complessità del modello in caso di dati mai visti.

Vediamo nello specifico alcuni algoritmi di apprendimento supervisionato.

k-Nearest Neighbors

Il *k-nearest neighbors* è un algoritmo utilizzato nel riconoscimento di pattern per la classificazione o regressione di oggetti basandosi sulle caratteristiche degli oggetti vicini a quello considerato. Questo algoritmo si bassa su un k fissato, che indica il numero di vicini da prendere in considerazione per fare la predizione.

La scelta di k dipende dalle caratteristiche dei dati. Generalmente all'aumentare di k si riduce il rumore che compromette la classificazione, ma il criterio di scelta per la classe diventa più labile.

Nel caso della classificazione è meglio scegliere un k dispari per escludere casi di indecisione e quindi poter sempre definire la classe del nuovo dato.

Nel caso di regressione il risultato sarà pari alla media dei valori target dei k più vicini.

Modelli lineari

I modelli lineari cercano di effettuare predizioni utilizzando una funzione lineare basata sull' insieme delle caratteristiche (*feature*) dell' elemento da analizzare.

Nel caso della regressione, la funzione è definita come segue:

$$y = w_0x_0 + w_1x_1 + \cdots + w_nx_n + b$$

dove n è il numero di *feature*, x_i le *feature*, w_i i pesi da attribuire ad esse, e b un termine noto.

I modelli lineari si possono applicare anche al contesto della classificazione, introducendo degli intervalli per definire a quale classe appartiene il singolo caso. Nel caso della classificazione binaria, ad esempio, avremmo una formula del tipo:

$$y = w_0x_0 + w_1x_1 + \cdots + w_nx_n + b > 0$$

dove, avendo le classi C_1 e C_0 se la y fosse maggiore dei 0, l'oggetto descritto dagli x_i verrebbe classificato come C_1 altrimenti come C_0

Alberi di decisione

Un albero di decisione è un modello predittivo, dove ogni nodo interno rappresenta una variabile, un arco verso un nodo figlio rappresenta un possibile valore per quella proprietà e una foglia il valore predetto per la variabile obiettivo a partire dai valori delle altre proprietà, che nell' albero è rappresentato dal cammino dal nodo radice al nodo foglia. Nei nodi troviamo delle domande la cui risposta è di tipo binario (vero o falso) mentre le foglie rappresentano le classi che vogliamo predire.

I problemi di *overfitting* e *underfitting* sono problemi ricorrenti che si presentano anche nel caso degli alberi di decisione. Infatti se viene costruito un albero troppo dettagliato, e quindi con un elevato livello di profondità, il modello tende ad adattarsi in maniera eccessiva ai dati usati in fase di allenamento generando un problema di *overfitting*.

Aumentando la profondità dell' albero, infatti, l'errore su un insieme di dati non incluso in quello di allenamento cresce.

Più permettiamo all' albero di avere tanti livelli, più lui ha la capacità di adattarsi meglio ai dati di training, e a partire da una certa lunghezza inizia a farlo a detrimento della sua capacità di generalizzazione. Questo succede proprio perché il modello si è adattato in maniera eccessiva all' insieme di dati di allenamento.

Per risolvere questo problema esistono due strategie:

- limitare a priori la profondità dell'albero,
- eliminare i nodi che contengono informazioni poco significative.

Support Vector Machine

Le *Support Vector Machine* sono dei modelli di apprendimento supervisionato associati ad algoritmi di apprendimento per la regressione e la classificazione.

Una *Support Vector Machine* individua un iperpiano o un insieme di iperpiani per separare i punti in uno spazio e quindi dividerli in diversi gruppi.

Mentre il problema originale può essere definito in uno spazio di finite dimensioni, spesso succede che gli insiemi da distinguere non siano linearmente separabili in quello

spazio. Per risolvere questo problema si ricorre alle funzioni *kernel* che sono in grado di mappare dei vettori da uno spazio n-dimensionale a uno spazio m-dimensionale.

Approccio Non Supervisionato

L'apprendimento non supervisionato è una tecnica di apprendimento automatico che consiste nel fornire al sistema informatico una serie di input che verranno riclassificati e organizzati sulla base di caratteristiche comuni per cercare di effettuare ragionamenti e previsioni sugli input successivi. Al contrario dell'apprendimento supervisionato, durante l'apprendimento vengono forniti all'apprendista solo esempi non annotati, in quanto le classi non sono note a priori ma devono essere apprese automaticamente.

Il principale algoritmo di questa categoria è il *clustering*, ovvero una tecnica in grado di suddividere in gruppi distinti elementi che hanno dati e caratteristiche in comune.

Un esempio di utilizzo ricade nell' ambito della sicurezza informatica. I tipi di attacco conosciuti sono probabilmente solo la punta dell' iceberg. Utilizzando tecniche di clustering è possibile individuare e bloccare attacchi ancora sconosciuti. Il Machine Learning non supervisionato potrebbe utilizzare informazioni dei clienti sugli utilizzi abituali dei servizi di una banca. Se un malintenzionato provasse ad effettuare delle operazioni tramite il nostro conto bancario dall' altra parte del mondo ad un orario differente da quello abituale, il calcolatore tramite tecniche di clustering, è in grado di riconoscere le operazioni abituali e mandare un messaggio di allarme se individua casi anomali.

Approccio Semi-Supervisionato

A metà strada tra l'apprendimento supervisionato e quello non supervisionato c'è l'apprendimento semi-supervisionato.

Questo approccio consiste nel combinare le due tecniche e fornire un risultato basandosi su un input eterogeneo costituito da dati etichettati e dati non etichettati. Questo perché i dati etichettati a volte possono essere difficili, costosi e dispendiosi in termini di tempo da recuperare, perché spesso i dati vengono etichettati manualmente da utenti specializzati. Allo stesso tempo i dati non etichettati possono essere relativamente facile da raccogliere, ma ci sono pochi modi di utilizzarli.

L' approccio semi-supervisionato risolve questi problemi utilizzando una grande quantità di dati non etichettati, insieme a una porzione di dati etichettati, per costruire classificatori migliori.

Apprendimento con rinforzo

Il quarto e ultimo approccio, chiamato approccio con rinforzo, punta a realizzare degli agenti autonomi in grado di scegliere azioni da compiere per il conseguimento di determinati obiettivi tramite interazione con l'ambiente in cui sono immersi.

A differenza degli approcci visti in precedenza, questo si occupa di problemi di decisioni sequenziali, in cui l'azione da compiere dipende dallo stato attuale del sistema e ne determina quello futuro.

Quando la macchina prende una decisione otterrà successivamente una "ricompensa", ispirata al concetto di rinforzo, sotto forma di punteggio che sarà alto o basso a seconda che la decisione presa sia giusta o sbagliata. Così l'agente cercherà di fare sempre meglio per arrivare ad ottenere il punteggio più alto possibile, prendendo decisioni corrette.

1.4 μ-learn

Nel capitolo precedente abbiamo visto le diverse tecniche utilizzate nel ML. L'algoritmo che andremo a descrivere, denominato *μ-learn*, si basa sull' induzione di insiemi fuzzy e ricade nell'approccio supervisionato, ovvero quella tecnica che necessita di dati preventivamente valutati per effettuare predizioni. In questo caso la predizione è il grado di appartenenza.

Supponiamo di avere un campione $x_1, \ldots, x_m$ in un dominio X, ed un insieme di gradi di appartenenza $\mu_1, \ldots, \mu_m$ associato ad un generico insieme fuzzy A.

Il problema di apprendere μ_A può essere suddiviso in due parti:

- determinare la forma di A;
- dedurre i parametri della funzione di membership μ_A

Per fare ciò dobbiamo partire dalle seguenti ipotesi:

- L'insieme $A_1 = \{x \in X \mid \mu_A(x) = 1\}$ deve contenere tutti i punti in X la cui trasposizione attraverso una mappatura Φ appartenga alla sfera di centro a e raggio R.
- L'appartenenza $\mu_A(x)$ dipende solo dalla distanza tra $\Phi(x)$ e a

Fatte queste ipotesi è possibile definire il problema: trovare la più piccola sfera avente centro a e raggio R che include la maggior parte delle immagini tramite Φ degli elementi $x \in X$.

La formulazione matematica di questo problema determina un problema di ottimizzazione vincolata, la cui funzione obiettivo è:

$$minR^2 + C\sum_{i=1}^{n}(\xi_i + \tau_i)$$

sottoposta ai seguenti vincoli:

$$\mu_i\|\Phi(x_i) - a\|^2 \leq \mu_i R^2 + \xi_i \tag{1}$$

$$(1-\mu_i)\|\Phi(x_i) - a\|^2 \geq (1-\mu_i)R^2 - \tau_i \tag{2}$$

$$\xi_i \geq 0, \tau_i \geq 0 \tag{3}$$

dove C è una costante di cui discuteremo successivamente, mentre ξ e τ sono le variabili di scarto utilizzate nel problema di ottimizzazione. ξ è la variabile slack legata al posizionamento dei punti all'interno della sfera, mentre τ è riferita al posizionamento dei punti all'esterno della sfera.

Possiamo notare che quando $\mu_i = 1$ otteniamo che:

$$\|\Phi(x_i) - a\|^2 \leq R^2 + \xi_i$$

ovvero che la distanza della trasformazione di x_i da a è minore o uguale al raggio della sfera e che il secondo vincolo diventa:

$$\tau_i \geq 0$$

che è già incorporata nel terzo vincolo, e quindi il nostro x_i sarà posizionato all'interno della sfera.

Allo stesso modo, quando $\mu_i = 0$ otteniamo che il vincolo 2 diventa:

$$\|\Phi(x_i) - a\|^2 \geq R^2 - \tau_i$$

ovvero che la distanza della trasformazione di x_i da a è maggiore o uguale al raggio della sfera e il primo vincolo così diventa:

$$\xi_i \geq 0$$

che è già incorporato nel terzo vincolo, e quindi il nostro x_i sarà posizionato all' esterno della sfera.

Utilizzando la formulazione duale di Wolfe, quello che otteniamo è il seguente

problema:

$$max \sum_{i=1}^{m}(\alpha_i\mu_i - \beta_i(1-\mu_i))k(x_i,x_i) - \sum_{i,j=1}^{m}(\alpha_i\mu_i - \beta_i(1-\mu_i))(\alpha_j\mu_j - \beta_j(1-\mu_j))k(x_i,x_j)$$

sottoposto ai vincoli:

$$\sum_{i=1}^{n}(\alpha_i\mu_i - \beta_i(1-\mu_i)) = 1 \tag{4}$$

$$0 \leq \alpha_i, \beta_i \leq C \tag{5}$$

dove k denota la funzione kernel associata al mapping nello spazio dell' immagine di Φ (ovvero $k(x_i,x_j) = \Phi(x_i) \cdot \Phi(x_j)$).

Indicando con * il valore ottimale per una variabile, è possibile dimostrare che:

$$R^2(x) = k(x,x) - 2\sum_{i=1}^{m}(\alpha i^*\mu_i - \beta_i^*(1-\mu_i))k(x,x_i) + \sum_{i,j=1}^{m}(\alpha_i^* - \beta_i^*(1-\mu_i))(\alpha_j^* - \beta_j^*(1-\mu_j))k(x_i,x_j)$$

così facendo è possibile calcolare la distanza tra il centro della sfera e l'immagine del punto dato x. In particolare, tutti i punti x con una membership $\mu_A(x) = 1$ soddisfanno $R^2(x) \leq R_1^2$ dove $R_1^2 = R^2(x_i)$ per qualunque vettore di supporto.

Apprendere la funzione di membership richiede di trovare il giusto compromesso nella scelta del parametro C, così come nei parametri del kernel.

Capitolo 2

Tecnologie e metodologie

Nel capitolo precedente ho un algoritmo di ottimizzazione vincolata e ci sono vari modi per risolvere questo problema numerico. Parlo di CVXOP, CVXPY, GUROBI.

2.1 Gurobi

2.2 CVXOPT

2.3 CVXPY

2.4 TensorFlow con Rilassamento Lagrangiano

2.4.1 Rilassamento Lagrangiano

Capitolo 3

Esperimenti

Spiegare come sono stati costruiti ed eventuali esperimenti Cercare un filo logico tra gli esperimenti, esperimenti miei e perché li ho fatti così.

3.1 Esperimenti a confronto

3.1.1 CVXPY

3.1.2 CVXOPT

3.1.3 TensorFlow

Confronto con esperimenti precedenti

Capitolo 4

Conclusioni

Ringraziamenti

Bibliografia

[1]