UNIVERSITÀ DEGLI STUDI DI MILANO
FACOLTÀ DI SCIENZE E TECNOLOGIE

CORSO DI LAUREA IN INFORMATICA

TITOLO TESI

Relatore:
Prof. Dario Malchiodi

Correlatore:
Prof. Anna Maria Zanaboni

Candidato:
Rita Pia Folisi

Matr. 908815

ANNO ACCADEMICO 2019/2020

Indice

1 Introduzione

2 Apprendimento automatico e logica fuzzy

Nel seguente capitolo si passano in rassegna i concetti di apprendimento automatico e di logica fuzzy, illustrando come i due argomenti possano essere coniugati insieme. Nel paragrafo 2.1 ci si soffermerà sull'apprendimento supervisionato, di cui i metodi proposti in seguito faranno parte. Nel paragrafo 2.2 si riprenderanno i concetti relativi ai *fuzzy set*, fino ad illustrare cosa significa indurre la funzione di appartenenza.

2.1 Apprendimento supervisionato

Con apprendimento automatico si intende l'abilità di una macchina di apprendere in maniera autonoma, attraverso diversi algoritmi ed esempi di addestramento. Ciò che la macchina impara è una funzione, una struttura computazionale attraverso il cui apprendimento la macchina è in grado di rispondere a problemi di vario genere, quali la classificazione, la regressione e il clustering.

Da un punto di vista formale[1] si considerino una funzione *f*, rappresentante il compito al quale si è interessati, e una funzione *h*, che approssima al meglio *f*. Le funzioni f e h sono definite sul vettore $\mathrm{X} = x_1 ... x_n$ composto da n elementi. Viene assunto a priori che h viene selezionata da una classe di funzioni H, alla quale f può appartenere. All'interno della classe di funzioni, h viene scelta in base a Ξ, un insieme contenente m esempi di apprendimento. Una volta scelta, h viene implementata da una macchina, che avrà dunque input X e in output $h(\mathrm{X})$.

L'apprendimento automatico può essere in generale di due tipi: modalità supervisionata e non supervisionata, la cui trattazione non verrà tuttavia effettuata in sede. Nel caso di modalità superivisionata, si conoscono anticipatamente gli output di f per i valori di Ξ. Se il modello di apprendimento riesce a selezionare una funzione h che approssima al meglio f per i valori di Ξ, allora si può ipotizzare che h sia un'ottima approssimazione per f anche in generale. In altri termini, attraverso esempi di apprendimento costituiti da coppie di input e di output, il modello impara una funzione capace di fornire output corretti in relazione a nuovi input. Quando tale output è rappresentato da una variabile discreta si parla di classificazione, se invece è una variabile continua si parla di regressione. Il campo esaminato riguarda i problemi di classificazione e pertanto l'obiettivo dei metodi presentati in seguito sarà, dato un vettore di input, fornire la *label* corrispondente e corretta. Un esempio di problema di classificazione è il filtraggio antispam. D'altra parte, il vettore di input X può essere definito anche come *feature vector* e le componenti x_i come feature o attributi, i cui valori possono essere di tre tipologie: numeri reali, discreti o valori categorici. L'input viene spesso rappresentato in forma non ordinata affiancando il valore dell'attributo all'attributo stesso. Ad esempio, si può avere (forma: rettangolo, perimetro: 14, colore: rosso), dove forma, perimetro e colore sono attributi, mentre rettangolo, 14 e rosso sono i valori degli attributi.

Una volta addestrato il modello con gli esempi di apprendimento, occorre valutare le performance per comprendere se questo è in grado di eseguire correttamente il suo

compito. L'idea è quindi misurare quanto le predizioni eseguite dal modello siano vicine all'output teorico. I criteri più noti sono l'accuratezza e le funzioni di errore. L'accuratezza è il numero di predizioni corrette divise per il totale delle predizioni effettuate e viene espresso in percentuale. Tuttavia, l'accuratezza non viene indicata come buon valutatore, dal momento che non riassume bene la robustezza assunta dal modello. Con robustezza s'intende la sensibilità del modello rispetto a variazioni dei dati. Si introducono pertanto una funzione di errore da minimizzare, come il *mean square error*, e una soglia sotto la quale il margine di errore è considerato accettabile. L'errore quadratico medio quantifica la differenza tra valore teorico e risposta predetta dal modello attraverso questa formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - h(x_i))^2$$

dove y_i è la label teorica, mentre $h(x_i)$ è la label predetta dal modello. Un'altra funzione d'errore utilizzata è RMSE ovvero la radice quadrata dell'errore quadratico medio:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - h(x_i))^2}$$

La valutazione, tuttavia, non verrà effettuata sugli esempi utilizzati per l'apprendimento, dal momento che l'obiettivo del modello è generalizzare con dati diversi da quelli osservati in precedenza. Una volta scelto il campo su cui è definito il compito che la macchina dovrà imparare, occorre distinguere una parte di dati dedita ad addestrare il modello e una parte di dati con cui valutarne la prestazione. In altre parole, dato un dataset, sarà opportuno distinguere tra *training set* e *testing set*, ognuno con compiti sensibilmente diversi. Il primo verrà utilizzato in fase di apprendimento, mentre il secondo verrà utilizzato esclusivamente per decretare le performance del modello. I due nuovi set possono presentare delle criticità tali da poter alterare l'intero processo di apprendimento e valutazione. Ad esempio, possono avere una dimensione molto elevata di features, comportando un aumento della complessità computazionale dell'algoritmo di apprendimento; per risolvere questo problema, si utilizzano algoritmi di selezione delle features, capaci di selezionare quelle più rilevanti per l'assegnazione dell'output corretto. Un altra criticità è causata dal rumore nei dati. Nel caso sia presente nei dati di training, il rumore di classe altera i valori di *f*, mentre il rumore di attributo altera i valori delle componenti dei vettori di input, compromettendo dunque la corretta associazione input-output. Inoltre, è bene osservare quanto il training set sia rappresentativo del campo in analisi: infatti, può accadere che questo presenti poca varietà e che il metodo dunque non sia in grado di generalizzare. Ad esempio, se nel training set sono presenti, sotto la label di "orologio", solo immagini di orologi rotondi e di color rosso, quando il modello si ritroverà a dover predire la label di un orologio quadrato e giallo probabilmente non darà la risposta corretta. Questo fenomeno prende il nome di *overfitting* ed è facilmente visibile quando il modello si adatta molto ai dati di training ma riporta performance sensibilmente inferiori con i dati di testing.

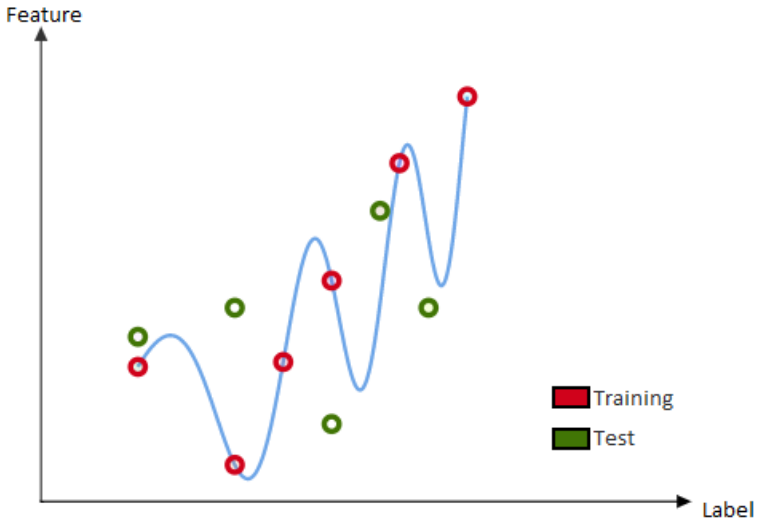

Figura 1: Illustrazione grafica di overfitting

Graficamente, si avrà la situazione presente in Figura 1, ovvero la funzione in blu h(x), appresa dal modello, si adatterà perfettamente ai dati di training (segnati in rosso), ma non riuscirà a ricoprire i dati di testing. In altri termini, se il modello allenato viene eseguito sui dati di training, il modello produrrà un numero sensibilmente minore di errori, rispetto a quando viene eseguito sui dati di testing.

Un buon modo per ridurre gli effetti dell'overfitting consiste nell'introdurre il *validation set*, cioè un sottinsieme del training set, atto a fornire una panoramica di come il modello stia lavorando, senza però fare valutazioni più precise come accade con il test set. In questo modo è già possibile vedere, durante la fase di addestramento, se il modello sappia generalizzare. L'overfitting, infatti, può avvenire anche per una configurazione errata degli iperparametri del modello, cioè parametri che non vengono appresi durante il processo di apprendimento ma che vengono definiti a priori. Il validation set permette di vedere se con i parametri impostati prima del training si stanno ottenendo buoni risultati e, in caso contrario, di modificarli senza dover attendere i risultati sul test set. Si introduce quindi il *tuning dei parametri*, ovvero una tecnica di ottimizzazione degli iperparametri. Viene definito il *range* degli iperparametri, il modello viene addestrato con tutte le combinazioni possibili di iperparametri e viene decretato il *validation score*, che generalmente coincide con l'accuratezza. Alla fine, si selezionano i parametri che massimizzano il validation score e vengono assegnati al modello. Questo processo prende il nome di *model selection*.

Spesso, per poter eseguire tuning dei parametri e addestramento, si ricorre alla *k-fold cross validation*. Più in generale, la cross validation è una tecnica utilizzata per la

suddivisione del dataset in training set, validation set e testing set. Si definisce il numero *fold* k e si divide il dataset Ξ in k sottinsiemi mutualmente esclusivi e di ugual dimensione: $\Xi_1 ... \Xi_k$. In seguito, si reitera l'algoritmo di apprendimento k volte considerando ad ogni iterazione il sottinsieme Ξ_k come training set e i rimanenti come validation o testing set. Il motivo per cui si utilizza questa tecnica e non si fa un solo semplice *split* del dataset è da ricercarsi nell'aleatorietà della suddivisione. Infatti, può succedere che uno split casuale inserisca nel training set osservazioni non generali, comportando dunque il già sopracitato problema dell'overfitting. In questo modo, alternando di volta in volta il sottinsieme di training, validation e testing, il modello risulterà più generalizzato e dunque robusto. La valutazione finale viene effettuata facendo la media delle valutazioni parziali dei sottinsiemi. In Figura 2 è presente una visualizzazione grafica della cross validation, con suddivisione in training set e validation set.

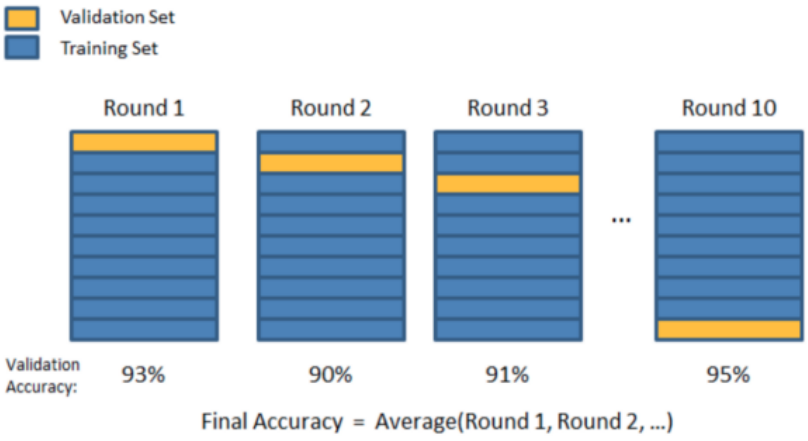

Figura 2: Cross validation

2.2 Logica fuzzy e fuzzy set

Il linguaggio e il ragionamento umano, spesso imprecisi e complessi, difficilmente sono riassumibili dalla logica classica basata sull'assegnazione di un valore di verità binario (vero o falso). Infatti, nel linguaggio naturale, molte espressioni risultano ambigue e, per la loro mancanza di formalità, diventano difficilmente trattabili con i classici strumenti della matematica. Una proposizione, in particolare, viene definita imprecisa nel momento in cui contiene predicati che possono essere veri o falsi solo parzialmente o solo ad un certo grado. Un esempio concreto è la frase "Il caffè è abbastanza dolce". In questa frase il concetto di dolce non è formalizzato e non ha un significato fisso, in quanto non viene indicata una certa quantità di zucchero affinché il caffè possa considerarsi effettivamente dolce. Inoltre, è accompagnata dall'avverbio "abbastanza", che non esprime a sua volta un grado di certezza ben definito. Pertanto, alla proposizione è impossibile affidare un valore di verità binario, dal momento che il grado di dolcezza è soggettivo e, dunque, non interpretabile universalmente. Per poter trattare queste situazioni si introduce la *logica fuzzy*[2], un'estensione della logica booleana. La logica fuzzy attribuisce ad ogni proposizione un valore di verità compreso tra 0 e 1, estremi inclusi: pertanto, è capace di trattare proposizioni vere (valore pari ad 1), false (valore pari a 0) e parzialmente vere o false (valore compreso tra 0 e 1). Permettendo quindi alle proposizioni di trovarsi in uno stato diverso dal vero o falso, la logica fuzzy consente di descrivere con flessibilità il ragionamento umano e gestire l'incertezza e l'inesattezza. Per questo motivo è ampliamente utilizzata in molti domini di applicazioni, come processi decisionali, analisi testuale, *data analysis*, *feature extraction* in un'immagine. Un esempio recente di applicabilità è stato presentato in [3], dove vengono elaborate immagini di risonanze magnetiche del cervello per poterne rilevare anomalie.

2.2.1 Fuzzy set e funzione di appartenenza

La logica fuzzy si basa sul concetto di *fuzzy set*, una generalizzazione della teoria classica degli insiemi. Un fuzzy set è un raggruppamento di oggetti, caratterizzato da una funzione che sintetizza il grado di appartenenza di ogni oggetto a tale fuzzy set. Formalmente, si avranno un insieme X, un generico elemento x dell'insieme X e un fuzzy set A contenuto in X. L'insieme A è caratterizzato da una funzione f_A, che prende il nome di *membership function*, che associa ad ogni punto x un numero reale dell'intervallo [0,1]. Il valore $f_A(x)$ rappresenta il grado di appartenenza (*membership value*) di x ad A. Più questo valore sarà prossimo ad 1, maggiore sarà il grado di appartenenza di x ad A. Quindi 1 indicherà la piena appartenenza all'insieme A, mentre 0 sarà la piena non-appartenenza. I valori compresi tra 1 e 0 indicheranno invece la parziale appartenenza o meno all'insieme A. Gli insiemi classici possono essere visti come un caso particolare di fuzzy set, dove gli unici valori ammessi di membership sono 1 e 0, corrispondenti a vero e falso. Questo tipo di insiemi prende il nome di *crisp set* o *boolean set* ed $f_A(x)$ avrà solo valori 0 o 1 (vedi Figura 3).

In Figura 4 è possibile osservare la rappresentazione grafica di un crisp set (a sinistra) e di un fuzzy set (a destra). Il crisp set ha i contorni ben delineati e presenta due soli

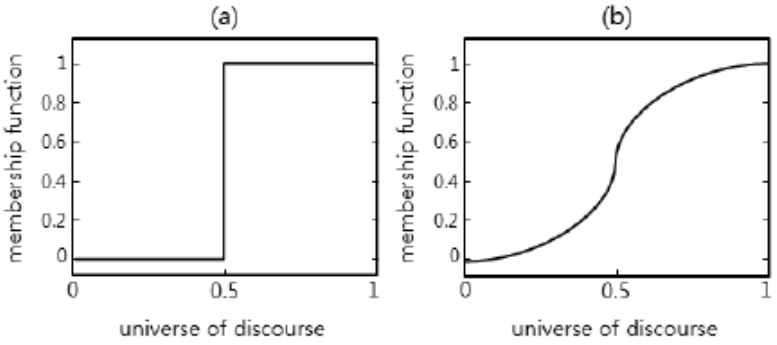

Figura 3: Membership function di un crisp set e di un fuzzy set

colori, bianco o nero: ciò accade poiché un punto può o appartenere al set e quindi trovarsi nello spazio in nero, oppure non appartenere al set e dunque trovarsi nello spazio bianco. Il fuzzy set, invece, avrà i contorni più sfumati e zone in grigio, dal momento che contempla l'imprecisione e l'incertezza e pertanto un punto può parzialmente appartenere ad un set.

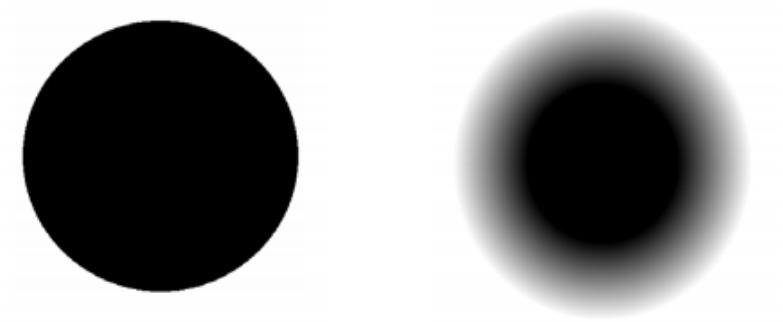

Figura 4: Rappresentazione grafica di un crisp set e di un fuzzy set

2.2.2 Variabile linguistica e FIS

Un altro concetto sul quale si basa la logica fuzzy è quello di variabili linguistiche, attraverso cui è possibile avvicinarsi al linguaggio naturale. Infatti, una variabile linguistica ha come valori attributi linguistici, ovvero parole utilizzate nel linguaggio naturale: in particolare, si potranno generare variabili da termini primari (ad esempio "alto"), modificatori (avverbi come "molto", "sicuramente", "abbastanza" etc.) e connettori ("and", "or", "not"). Secondo la definizione di Zadeh, una variabile linguistica viene definita come una quintupla $(X, T(X), U, G, M)$ dove:

- X è il nome della variabile linguistica
- $T(X)$ è l'insieme dei termini dei valori linguistici della variabile X (*term set*)
- U è l'universo del discorso, cioè il dominio in cui sono definite le variabili
- G è la regola sintattica che genera i nomi in $T(X)$ applicando modificatori linguistici ai termini primari
- M è la regola semantica che assegna a ciascun nome il suo significato, cioè un insieme fuzzy attraverso una funzione di compatibilità $c : U - - > [0, 1]$

Ad esempio, data una variabile linguistica X = età, avremo come universo U il range dei valori assumibili da X, ad esempio U=[0, 122]. Un insieme $T(X)$ può essere "vecchio, molto vecchio, giovane, poco giovane". La regola G genera nomi applicando modificatori linguistici ("molto", "poco") ai termini primari ("vecchio", "giovane"). Infine, la regola M assegnerà ad esempio al valore 27 all'etichetta "giovane" con un grado di 0.7. I modificatori, divisi in varie categorie come quelli di concentrazione, di dilatazione e di contrasto, riescono così a rimodellare la regola semantica M, di modo da poter rispecchiare e rappresentare la vaghezza caratterizzante il linguaggio naturale.

Le variabili linguistiche possono essere utilizzate all'interno dei FIS (*fuzzy interference systems*)[4], che consentono il mapping tra input-output crisp attraverso funzioni di appartenenza, operatori logici e regole fuzzy. I FIS vengono utilizzati per classificazione, *data analysis*, *decision-making problems* e ragionamento approssimativo. Si avvalgono di un processo di *fuzzification*[5], cioè di costruzione di un insieme fuzzy a partire da valori di una data variabile, e di *defuzzification*, che trasforma un insieme fuzzy in un valore numerico. La fuzzification è dunque lo step che determina il grado con cui ogni input appartiene ad ogni fuzzy set e si avvale della funzione di appartenenza.

2.2.3 Apprendimento automatico supervisionato e fuzzy set

Per molto tempo, all'interno della teoria dei fuzzy set, grande peso ha assunto la *knowledge representation*, soprattutto in relazione allo sviluppo di sistemi intelligenti e intelligenza artificiale. Tuttavia, negli ultimi anni sono emersi sempre più problemi con l'approccio puramente *knowledge-driven*, che pone le basi sulla logica per poter formalizzare modelli, dal momento che risulta essere difficoltoso, intricato e spesso poco produttivo in termini di risultati. Di conseguenza, si è pensato di adattare i concetti della teoria dei fuzzy set ad un approccio *data-driven*, che invece utilizza i dati empirici per estrarre modelli. In tal senso dunque si può osservare come la teoria dei fuzzy set sia stata applicata all'interno dell'apprendimento automatico supervisionato, portando così alla nascita della *fuzzy machine learning*[6]. D'altronde, la teoria dei fuzzy set riesce a rispondere a problemi orientati verso la classificazione o la data analysis, risultando pertanto un ottimo strumento per l'ambito dell'apprendimento automatico supervisionato. Difatti, la logica fuzzy risulta idonea per, ad esempio, effettuare *data selection*, come operare con la vaghezza risultante dai dati empirici, che risentono di linguaggio

naturale[7]. Gli algoritmi classici del machine learning vengono pertanto estesi ai corrispettivi fuzzy: saranno presenti dunque alberi di decisione fuzzy[8], fuzzy clustering[9], fuzzy k-nearest neighbors[10] e così via. In questo modo è dunque possibile combinare punti di forza dei metodi classici e della teoria fuzzy, per ottenere un metodo che riesce ad ottimizzare parametri altrimenti più complessi da migliorare, come l'interpretabilità dei dati e la gradualità.

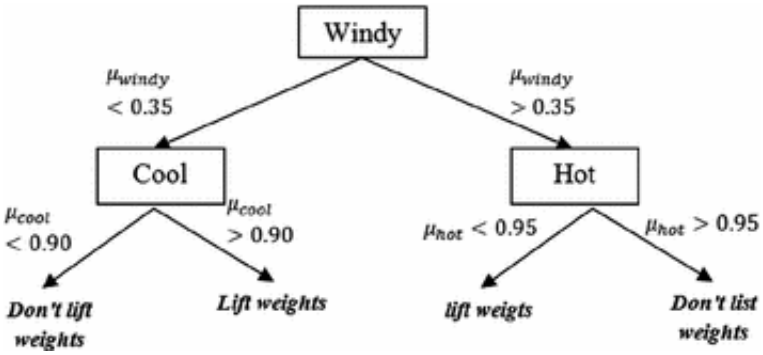

Figura 5: Esempio di decision tree fuzzy

Relativamente al fuzzy machine learning, il focus di questa tesi è l'induzione di funzioni di appartenenza, cioè la produzione di modelli che, date variabili in input, riescano a predire i gradi di appartenenza ai diversi insiemi fuzzy. In generale la costruzione della funzione di appartenenza può essere effettuata attraverso metodi deduttivi e induttivi. I primi sono knowledge-driven, mentre i secondi presentano un approccio data-driven. Nel prossimo capitolo ci si soffermerà su algoritmi induttivi, con il fine di risolvere problemi di classificazione attraverso l'induzione della funzione di appartenenza.

Riferimenti bibliografici

[1] J. G. Smith and H. K. Weston, *Introduction to Machine Learning: An Early Draft of a Proposed Textbook*. Stanford University, 1998.

[2] F. Dernoncourt, "Introduction to fuzzy logic," 01 2013.

[3] H. Deng, W. Deng, X. Sun, Y. Chaohui, and X. Zhou, "Adaptive intuitionistic fuzzy enhancement of brain tumor mr images," *Scientific Reports*, vol. 6, p. 35760, 10 2016.

[4] S. Guillaume, "Designing fuzzy inference systems from data: An interpret ability-oriented review," *Fuzzy Systems, IEEE Transactions on*, vol. 9, pp. 426 – 443, 07 2001.

[5] E. Kayacan and M. A. Khanesar, "Chapter 2 - fundamentals of type-1 fuzzy logic theory," in *Fuzzy Neural Networks for Real Time Control Applications* (E. Kayacan and M. A. Khanesar, eds.), pp. 13 – 24, Butterworth-Heinemann, 2016.

[6] E. Hüllermeier, "Fuzzy methods in machine learning and data mining: Status and prospects," *Fuzzy Sets and Systems*, vol. 156, no. 3, pp. 387 – 406, 2005. 40th Anniversary of Fuzzy Sets.

[7] R. Viertl, *Statistical Methods for Non-Precise Data*, pp. 1442–1444. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[8] C. Olaru and L. Wehenkel, "A complete fuzzy decision tree technique," *Fuzzy sets and systems*, vol. 138, no. 2, pp. 221–254, 2003.

[9] T. Lei, X. Jia, Y. Zhang, S. Liu, H. Meng, and A. K. Nandi, "Superpixel-based fast fuzzy c-means clustering for color image segmentation," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 9, pp. 1753–1766, 2018.

[10] J. Maillo, J. Luengo, S. García, F. Herrera, and I. Triguero, "Exact fuzzy k-nearest neighbor classification for big datasets," in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6, IEEE, 2017.