Data and Web Mining

Raccolta domande teoriche

Quali sono le differenze tra gli algoritmi di supervised learning e quelli di unsupervised learning?

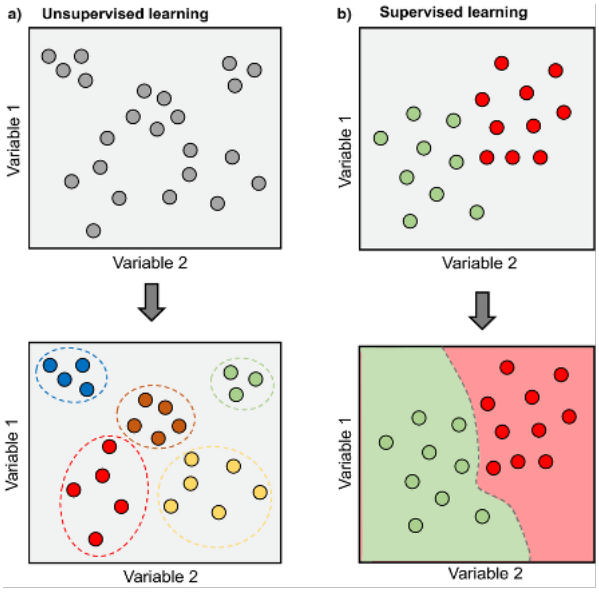
Supervised Learning:

- Obiettivo: Predire un'etichetta di output basata su una o più caratteristiche di input.
- Esempi di Applicazioni: Diagnosi mediche, riconoscimento vocale, riconoscimento di immagini, previsione del prezzo delle azioni.
- Valutazione del Modello: È possibile valutare la precisione del modello confrontando le previsioni del modello con le etichette vere.
- Complessità Computazionale: Tendenzialmente più elevata rispetto all'apprendimento non supervisionato, a causa della necessità di addestrare il modello con un grande numero di esempi etichettati.
- Overfitting: È un rischio maggiore in quanto il modello potrebbe adattarsi troppo ai dati di addestramento e perdere la capacità di generalizzare su dati nuovi e non visti.
- Esempio algoritmi: Decision Trees, Support Vector Machines, Neural Networks, Linear Regression.

Unsupervised Learning:

- Obiettivo: Esplorare la struttura sottostante o le relazioni tra le variabili nei dati.
- Esempi di Applicazioni: Segmentazione del mercato, organizzazione di grandi biblioteche di documenti, compressione di immagini.
- Valutazione del Modello: È più difficile valutare l'efficacia del modello, poiché non ci sono etichette vere con cui confrontare le previsioni o i raggruppamenti del modello.
- Complessità Computazionale: Tendenzialmente meno elevata rispetto all'apprendimento supervisionato, ma può variare a seconda dell'algoritmo e del numero di dati.
- Scoperta di Conoscenza: È particolarmente utile quando non si conoscono le etichette o quando si desidera scoprire relazioni non note tra i dati.
- Esempio algoritmi: K-Means, Hierarchical Clustering, DBSCAN, t-SNE.

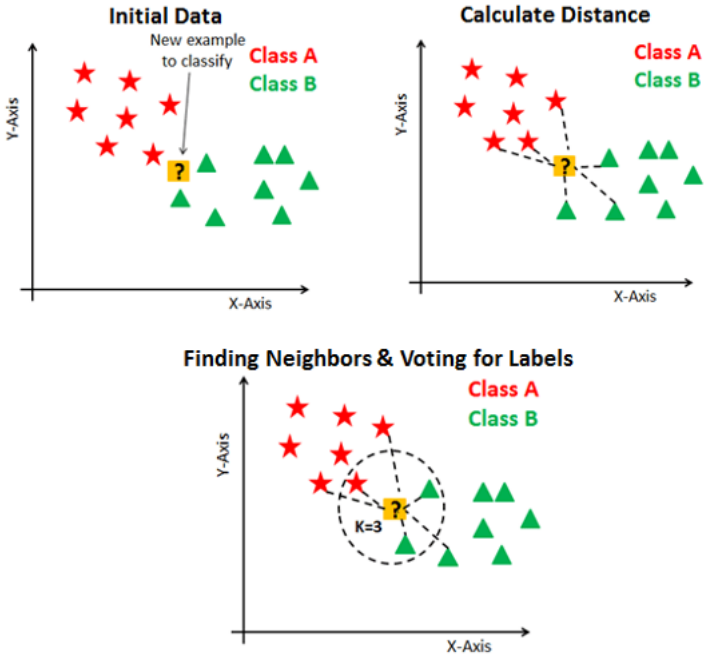
In generale, gli algoritmi di apprendimento supervisionato sono utilizzati per problemi di classificazione e regressione, mentre gli algoritmi di apprendimento non supervisionato sono utilizzati per problemi di clustering e riduzione della dimensionalità.



Descrivere l'algoritmo K-nn e spiegare cosa succede cambiando il parametro k

L'algoritmo K-nn, o K-Nearest Neighbors, è un metodo di apprendimento supervisionato utilizzato sia per la classificazione che per la regressione. Il suo funzionamento è intuitivo: dato un nuovo punto da classificare o da cui prevedere un valore, l'algoritmo identifica i k punti più vicini nel dataset di addestramento e assegna la classe o il valore medio di questi vicini al nuovo punto.
Funzionamento:

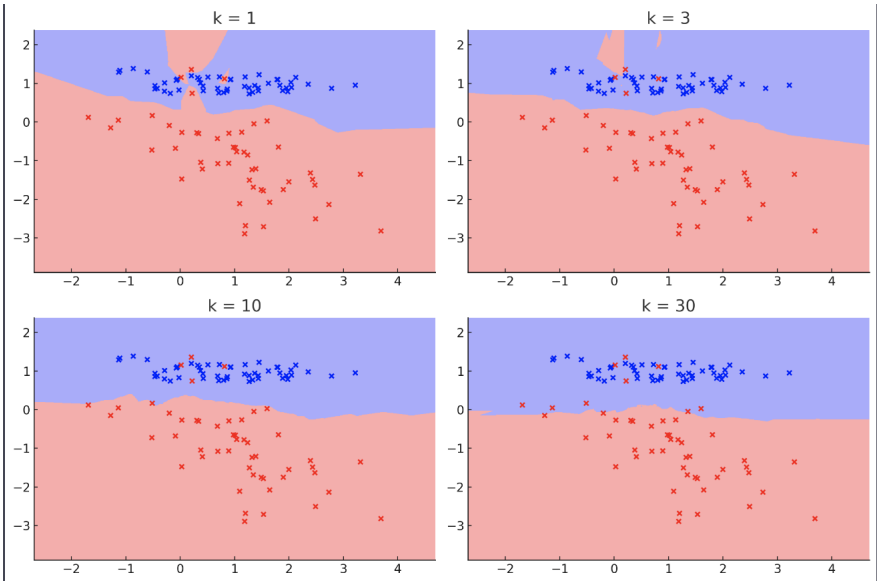
- Initial Data: Si parte con un dataset di addestramento dove ogni punto è etichettato con una classe o un valore.
- Calculate Distance: Per ogni nuovo punto, si calcola la distanza da tutti i punti nel dataset di addestramento. La distanza euclidea è comunemente utilizzata, ma altre metriche di distanza possono essere applicate a seconda del contesto.
- Finding Neighborhood & Voting for Labels: Si selezionano i k punti più vicini e, in base al task:
  - Classificazione: Si assegna la classe più frequente tra i k vicini (majority voting).
  - Regressione: Si assegna la media dei valori dei k vicini.



Il valore di k è cruciale. Un k troppo piccolo rende l'algoritmo sensibile al rumore, mentre un k troppo grande lo rende insensibile alle variazioni locali. È comune utilizzare un numero dispari per k in task di classificazione per evitare pareggi nel voting. Inoltre, è possibile attribuire pesi diversi ai vicini in base alla loro distanza dal nuovo punto, dando più importanza ai punti più vicini.

Per migliorare le prestazioni, è consigliabile scalare le feature in modo che abbiano tutte lo stesso range di variazione, utilizzando tecniche come MinMax Scaler o StandardScaler. Questo perché le feature con variazioni più ampie potrebbero dominare quelle con variazioni più ridotte nel calcolo delle distanze.

L'algoritmo K-nn è semplice ed efficace, soprattutto con dati numerici e quando si dispone di una metrica di distanza appropriata. Tuttavia, ha un costo computazionale elevato, soprattutto con dataset di grandi dimensioni, poiché richiede il calcolo della distanza da tutti i punti del dataset per ogni nuovo punto.



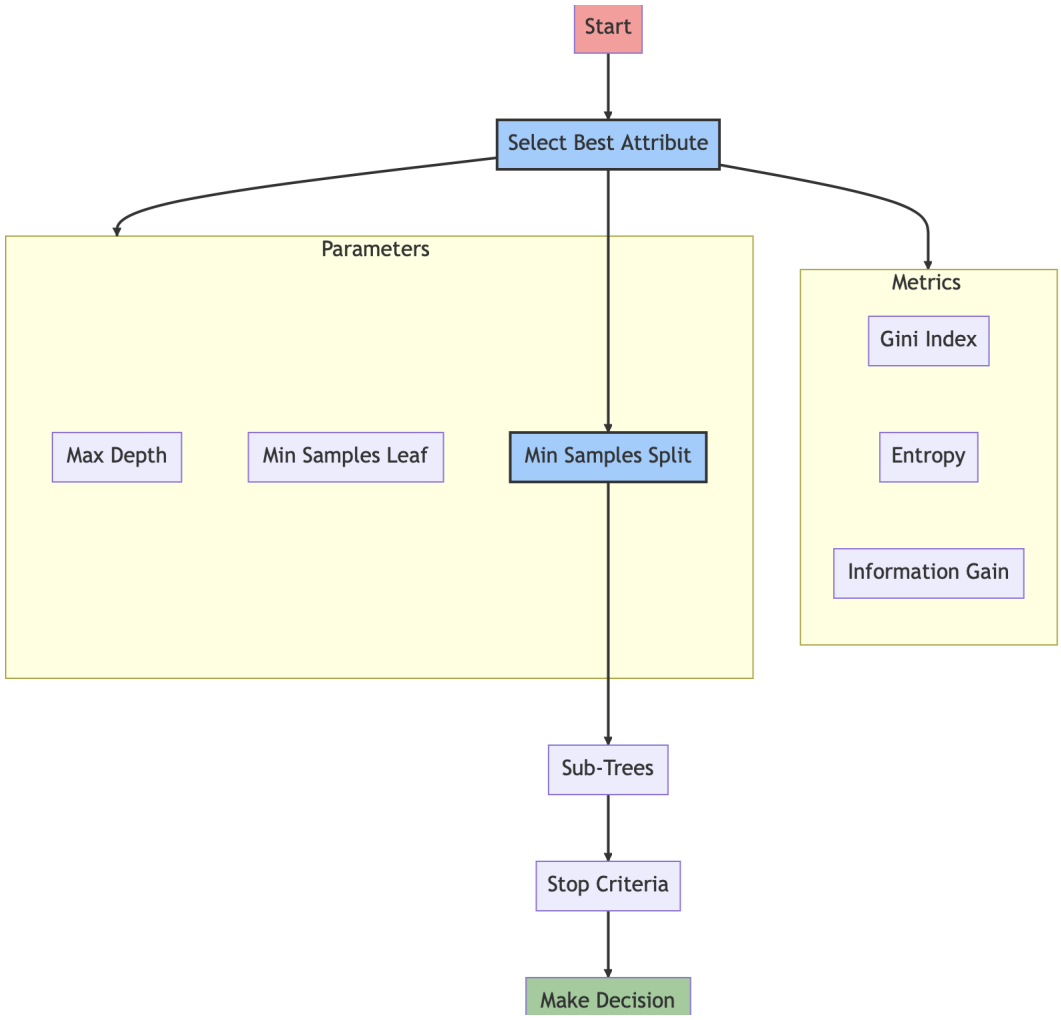
L'immagine mostra come l'algoritmo K-nn varia con diversi valori di k, illustrando l'effetto di overfitting con k piccoli e di underfitting con k grandi. Nello specifico abbiamo ogni grafico che rappresenta un diverso valore di k e mostra come

l'algoritmo classifica i punti in due classi diverse (colori diversi nello sfondo) basandosi sui punti del dataset (punti colorati nel grafico).

- Nel primo grafico, con $k = 1$, si nota un evidente overfitting, con una frontiera di decisione molto irregolare che segue strettamente i punti del dataset.
- Nel secondo grafico, con $k = 3$, la frontiera di decisione è leggermente più liscia ma ancora abbastanza irregolare.
- Nel terzo grafico, con $k = 10$, la frontiera di decisione è più liscia e generalizzata, ma potrebbe ancora catturare bene la struttura dei dati.
- Nel quarto grafico, con $k = 30$ la frontiera di decisione è molto semplice e liscia, il che potrebbe indicare un potenziale underfitting, dove il modello non cattura adeguatamente la struttura sottostante dei dati.

Descrivere l'algoritmo per la costruzione di un Decision Tree e spiegare le metriche e i vari parametri

Gli Alberi di Decisione sono modelli di apprendimento supervisionato utilizzati per problemi di classificazione e regressione. Un albero di decisione divide ricorsivamente lo spazio degli input in regioni omogenee, per poi assegnare una classe (o un valore, in caso di regressione) alla regione in cui cade un nuovo input.



Algoritmo di Costruzione: la costruzione di un albero di decisione inizia con la radice, che contiene l'intero dataset di addestramento. Il dataset viene poi diviso in sottoinsiemi omogenei basandosi su un attributo e un valore di soglia. Questo processo si ripete ricorsivamente su ogni sottoinsieme, creando nuovi nodi, fino a quando tutti i dati in un nodo appartengono alla stessa classe o fino a quando non sono soddisfatti altri criteri di arresto.

Criteri di Divisione: i criteri di divisione, come l'Entropia e l'Indice Gini, misurano l'impurità di un nodo. Un nodo è puro quando tutti i suoi dati appartengono alla stessa classe. Il miglior attributo e valore di soglia da utilizzare per dividere un nodo sono quelli che riducono al massimo l'impurità.

- **Entropia:** Misura dell'incertezza o del disordine in un nodo.

$$\text{Entropia}(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

- **Guadagno di Informazione:** Differenza tra l'entropia del nodo padre e la somma ponderata delle entropie dei nodi figli.

$$\text{Guadagno}(S, A) = \text{Entropia}(S) - \sum_{v \in \text{Val}(A)} \frac{|S_v|}{|S|} \text{Entropia}(S_v)$$

Criteri di Arresto: alcuni dei criteri di arresto includono la profondità massima dell'albero, il numero minimo di campioni per nodo e il numero minimo di campioni per foglia.

Metriche di Valutazione: le metriche di valutazione, come l'Accuracy, la Precision, la Recall e l'F1 Score, aiutano a quantificare le prestazioni di un modello di classificazione.

- **Accuracy:** Rapporto tra le previsioni corrette e il totale delle previsioni.
- **Precision:** Rapporto tra i veri positivi e la somma dei veri positivi e dei falsi positivi.
- **Recall:** Rapporto tra i veri positivi e la somma dei veri positivi e dei falsi negativi.
- **F1 Score:** Media armonica tra precision e recall.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Parametri del Modello: alcuni parametri chiave che influenzano la costruzione dell'albero di decisione sono la profondità massima dell'albero, il numero minimo di campioni richiesti per dividere un nodo interno e il numero minimo di campioni richiesti per essere presenti in un nodo foglia.

Cosa cambia in un decision Tree utilizzato in un task di classificazione a un decision Tree utilizzato per un task di regressione?

Un *Decision Tree* è un modello di apprendimento supervisionato utilizzato sia per problemi di classificazione che di regressione. La principale differenza tra i due tipi di alberi risiede nella natura della variabile obiettivo e nelle metriche di errore utilizzate per effettuare le divisioni.

Decision Tree per Classificazione
Nel contesto della classificazione, diverse metriche possono essere utilizzate per valutare la qualità delle divisioni:

- **Classification Error:**

$$\text{Error} = 1 - \max(p_1, p_2, \ldots, p_m)$$

  dove p_i rappresenta la proporzione di campioni appartenenti alla classe i in un nodo.

- **Information Gain:**

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Val}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

  dove l'entropia è definita come:

$$\text{Entropy}(S) = -\sum_{i=1}^{m} p_i \log_2 p_i$$

- **Gain Ratio:**

$$\text{GainRatio}(S, A) = \frac{\text{Gain}(S, A)}{\text{SplitInformation}(S, A)}$$

  con:

$$\text{SplitInformation}(S, A) = -\sum_{v \in \text{Val}(A)} \frac{|S_v|}{|S|} \log_2 \frac{|S_v|}{|S|}$$

- **Gini Index:**

$$\text{Gini}(S) = 1 - \sum_{i=1}^{m} p_i^2$$

  dove p_i è la proporzione di campioni appartenenti alla classe i in un nodo.

Decision Tree per Regressione
Nel contesto della regressione, l'obiettivo è prevedere un valore continuo piuttosto che una classe. La metrica di errore comunemente utilizzata per la divisione è il *Mean Squared Error* (MSE):

$$\text{MSE}(S) = \frac{1}{|S|} \sum_{i \in S} (y_i - \bar{y})^2$$

dove y_i è il valore obiettivo del campione i e $\bar{y}$ è la media dei valori obiettivo in S.

La principale differenza nella struttura dell'albero tra classificazione e regressione risiede nelle foglie: in un albero di regressione, una foglia contiene la media dei valori obiettivo dei campioni in essa, mentre in un albero di classificazione, contiene la classe maggioritaria.