Oggetto: Algoritmo K-means clustering

Introduzione

Il k-means clustering è un algoritmo non supervisionato utile per la segmentazione dei dati. L'obiettivo è quello di raggruppare le osservazioni simili per scoprire pattern nascosti dividendo i dati in cluster.

Struttura

L'algoritmo prevede la creazione di un punto di riferimento denominato centroide per un numero desiderato di classi e quindi assegnando i punti dati ai cluster di classi in base al punto di riferimento più vicino. Prima di tutto si definiscono i cluster che sono solo gruppi di elementi ed i raggruppamenti consistono semplicemente nell'associare elementi in quei gruppi, pertanto gli algoritmi di clustering mirano a fare due cose:

- Assicurati che tutti i punti dati in un cluster siano il più simili possibile tra loro;
- Assicurati che tutti i punti dati nei diversi cluster siano il più possibile dissimili tra loro.

In generale gli algoritmi di clustering raggruppano gli elementi in base a una metrica di somiglianza, questo viene spesso identificato trovando il "centroide" dei diversi gruppi possibili nel dataset. Esistono diversi algoritmi di clustering, ma l'obiettivo è lo stesso determinare i gruppi intrinseci a un insieme di dati.
Il K-Means Clustering è uno dei tipi di algoritmi più comunemente usati e funziona in base alla quantizzazione vettoriale ossia viene identificato un punto nello spazio selezionato come origine, quindi i vettori vengono disegnati dall'origine a tutti i punti dati in input. In generale, il clustering K-means può essere suddiviso nelle seguenti fasi:

1. Posizionare tutte le istanze in sottoinsiemi, dove il numero di sottoinsiemi è uguale a K;
2. Trovare il punto/centroide medio delle partizioni del cluster appena create;
3. Sulla base di questi centroidi, assegna ogni punto a un cluster specifico;
4. Calcolare le distanze da ogni punto ai centroidi e assegnare punti ai cluster in cui la distanza dal centroide è la minima.
5. Dopo che i punti sono stati assegnati ai cluster, trovare il nuovo centroide dei cluster.

Questi passaggi vengono ripetuti fino al termine del processo di formazione.

In alternativa, dopo che i centroidi sono stati posizionati, si può concepire il clustering di K-means come uno scambio avanti e indietro tra due diverse fasi: l'etichettatura dei punti dati e l'aggiornamento dei centroidi. Nella fase di etichettatura del punto dati, ad ogni punto dati viene assegnata un'etichetta che lo colloca nel cluster appartenente al centroide più vicino, questo viene in genere determinato utilizzando la distanza euclidea al quadrato, sebbene sia possibile utilizzare altre metriche di distanza a seconda del tipo di dati inseriti nell'algoritmo di clustering. Mentre nella fase di aggiornamento del centroide questo viene calcolato trovando la distanza media tra tutti i punti dati attualmente contenuti in un cluster.

Un decisione iniziale molto importante riguarda la scelta del numero di classi K da considerare, visto che tale dato non è noto a priori, vi sono diverse tecniche che permettono una sua determinazione. Una tecnica per selezionare il numero dei gruppi da definire è chiamata "tecnica del gomito", questa consiste nell'esecuzione di un algoritmo di clustering K-means per un intervallo di diversi valori K e nell'utilizzo di una metrica di precisione, tipicamente la somma dell'errore quadratico, per determinare quali valori di K danno i risultati migliori. La somma dell'errore quadratico viene calcolata con la distanza media tra il centroide di un cluster e i punti dati in quel cluster.
Il clustering K-means può essere utilizzato in modo sicuro in qualsiasi situazione in cui i punti dati possono essere segmentati in gruppi/classi distinti. Di seguito sono riportati alcuni esempi di casi d'uso comuni come ad esempio nella classificazione dei documenti, raggruppando i documenti in base a funzionalità come argomenti, tag, utilizzo delle parole, metadati e altre funzionalità del documento; potrebbe anche essere utilizzato per classificare gli utenti come bot o meno in base a modelli di attività come post e commenti. Inoltre può essere utilizzato anche per mettere le persone in gruppi in base ai livelli di preoccupazione durante il monitoraggio della loro salute, in base a caratteristiche come comorbidità, età, anamnesi del paziente; altresì può essere utilizzato per la creazione di sistemi di "raccomandazione" come ad esempio nello streaming video dove gli utenti possono essere raggruppati in base a modelli di visualizzazione e contenuti simili consigliati. Infine potrebbe essere utilizzato per attività di rilevamento di anomalie, evidenziando potenziali casi di frode o articoli difettosi.

Implementazioni

Sono state implementate, come richiesto, due versioni dell'algoritmo: una sequenziale ed una parallela. In questi algoritmi, il numero di cluster è fissato a 2, ma può essere modificato cambiando il valore della costante *k* mentre il vettore *data* contiene i punti da clusterizzate.

Inizialmente i centroidi sono assegnati ai primi due punti del vettore *data* mentre la funzione *distance* calcola la distanza euclidea tra due punti. Infine il ciclo *while* esegue l'algoritmo K-means finché i centroidi non convergono, l'assegnazione dei cluster ai punti avviene nel primo ciclo, mentre l'aggiornamento dei centroidi avviene nel secondo ciclo. Il tempo di elaborazione è misurato utilizzando la funzione *clock* della libreria *ctime*.

Il programma utilizza un dataset di dati denominato *data*, il numero di cluster *k*, i vettori *centroids* e *clusters* per contenere rispettivamente i centroidi e l'assegnazione di ogni punto ad un cluster; la funzione *kMeansClustering()* viene chiamata per eseguire l'algoritmo K-means clustering sui dati.

La variabile *start* contiene il numero di cicli di clock al momento dell'avvio dell'algoritmo, mentre la variabile *end* contiene il numero di cicli di clock al termine dell'algoritmo. Il tempo di elaborazione viene calcolato tramite differenza tra i valori *end* e *start* e dividendo il risultato per il valore di *CLOCKS_PER_SEC*, che indica il numero di cicli di clock al secondo. I risultati vengono stampati a schermo per mostrare i centroidi e l'assegnazione del cluster per ogni punto ed il tempo di elaborazione.

Risultati

L'output ricevuto circa i centroidi identificati e le assegnazioni ai cluster sono i medesimi per entrambe le implementazioni ed il dettaglio dei risultati sono presenti nei relativi file denominati *OutputSequenziale.txt* ed *OutputParallelo.txt*. Ciò che è variato in maniera significativa all'aumentare della dimensione del dataset in ingresso è stato il tempo di esecuzione dell'algoritmo *kMeansClustering(),* poiché, già con una matrice di dati in input di dimensioni 100x2 nel caso di struttura sequenziale il tempo di elaborazione è stato di 0,000174 secondi mentre nel caso di implementazione parallela utilizzando la libreria OpenMP il tempo di esecuzione è sceso a 0,000126 secondi, questa differenza si è accentuata utilizzando un dataset con 500 punti in cui l'elaborazione sequenziale è stata effettuata in 0,001308 secondi mentre quella parallela di appena 0,000937. Di seguito una tabella con le prove effettuate:

Osservazioni	Sequenziale	Parallelo
10	0,000025	0,000023
20	0,000055	0,000044

50	0,000114	0,000105
100	0,000336	0,000227
500	0,001308	0,000937

In conclusione l’algoritmo parallelo risulta più efficiente al crescere dei dati in input di una sequenziale, poiché riesce ad effettuare l’elaborazione in un numero di cicli di clock inferiore.