Tecniche di clusterizzazione non supervisionata e clusterizzazione di documenti testuali

Daniele Capelli - Matricola 226723

Anno Accademico 2023 - 24

Sommario

L'analisi cluster è una tecnica di classificazione non supervisionata che viene utilizzata per trovare sottogruppi, i cosiddetti cluster, all'interno di un dataset: queste metodologie possono essere utilizzate in svariati ambiti, come ad esempio nel marketing, per cercare gruppi di persone con interessi comuni, o in genetica, dove tecniche di clustering vengono adoperate per cercare quali geni possono essere rilevanti nella predisposizione a determinate malattie. Questi metodi si basano fortemente sulla scelta di una misura di dissimilarità tra osservazioni (o gruppi di osservazioni): nel seguito vedremo nel dettaglio alcuni esempi. Ci focalizzeremo poi sulle due principali vie per realizzare il nostro obiettivo: K-means custering e clustering gerarchico. Infine vedremo come applicare quanto esposto per la classificazione di documenti testuali.
Nel seguito faremo riferimento a [5] e [3] per la nostra analisi.

1 Misura di dissimilarità

1.1 Matrice di dissimilarità

Uno degli aspetti fondamentali da delineare all'interno della definizione di un metodo di clusterizzazione è la scelta di una misura di somiglianza o diversità tra due oggetti. Generalmente raccogliamo le differenze in una matrice $\mathbf{D}$ di dimensione $N \times N$, dove N indica il numero di oggetti che abbiamo a disposizione, costruita in modo che l'elemento $d_{ii'}$ rappresenti la prossimità tra l'oggetto i-mo ed i'-mo. Solitamente si assume che gli elementi di questa matrice siano non negativi e che la diagonale sia formata da soli zero.

1.2 Dissimilarità basata sugli attributi

Solitamente ci vengono fornite misure x_{ij} per $i = 1, 2, \ldots, N$ sulle variabili $j = 1, 2, \ldots, p$, che vengono dette attributi. Per costruire la differenza $D(x_i, x_{i'})$

tra le due osservazioni i ed i' si definisce la dissimilarità $d_j(x_{ij}, x_{i'j})$ sui valori del j-mo attributo per poi usare

$$D(x_i, x_{i'}) = \sum_{j=1}^{p} d_j(x_{ij}, x_{i'j}) \tag{1}$$

La scelta più comune che viene compiuta consiste nell'utilizzare il quadrato della distanza come misura di differenza, ovvero

$$d_j(x_{ij}, x_{i'j}) = (x_{ij} - x_{i'j})^2$$

In alcuni casi questa scelta non risulta la più efficace per risolvere il nostro problema. Vediamo più nel dettaglio alcune alternative.

- Per le variabili quantitative si sceglie una misura che sia una funzione strettamente crescente rispetto al valore assoluto della differenza tra i due valori osservati, ovvero $d(x_i, x_{i'}) = l(|x_i - x_{i'}|)$: ad esempio si può scegliere di usare proprio il valore assoluto di questa differenza. In questo caso, rispetto al quadrato della distanza, si mette meno enfasi sulle differenze maggiori, mentre si accentuano le differenze minori. In alternativa si può utilizzare la correlazione

  $$\rho(x_i, x_{i'}) = \frac{\sum_j (x_{ij} - \bar{x}_i)(x_{i'j} - \bar{x}_{i'})}{\sqrt{\sum_j (x_{ij} - \bar{x}_i)^2 \sum_j (x_{i'j} - \bar{x}_{i'})^2}}$$

  dove $\bar{x}_i = \frac{\sum_j x_{ij}}{p}$ è la media lungo le variabili.

- Se stiamo utilizzando delle variabili ordinali, basta sostituire gli M valori dati con il valore

  $$\frac{i - \frac{1}{2}}{M}, \quad i = 1, \ldots, M$$

  e creare un modello usando questi nuovi dati e una delle misure descritte precedemente.

- Infine con variabili categoriche che non presentano un ordine naturale bisogna delineare esplicitamente il livello di differenza tra due categorie, ad esempio tramite una matrice $M \times M$, il cui elemento di posizione rr' rappresenta il livello di differenza tra le categorie r e r'. Solitamente si sceglie di mettere 1 negli elementi non diagonali per dare un peso uniforme alle differenze fra categorie, ma si può optare di utilizzare quantità diverse per ogni elemento in modo da dare maggiore risalto ad alcune differenze.

1.3 Dissimilarità tra oggetti

Definita la differenza a livello di attributi dobbiamo trovare un modo per fornire la differenza totale di due osservazioni $D(x_i, x_{i'})$. In alternativa alla semplice

formula (1), si possono pesare i vari contributi attraverso la formula

$$D(x_i, x_{i'}) = \sum_{j=1}^{p} \omega_j d_j(x_{ij}, x_{i'j}), \qquad \sum_{j=1}^{p} \omega_j = 1$$

Precisiamo un dettaglio: impostare tutti i pesi ω_j allo stesso valore non equivale ad assegnare lo stesso contributo a ogni attributo. Infatti l'influenza del j-mo attributo sulla quantità $D(x_i, x_{i'})$ dipende dal suo contributo relativo alla dissimilarità media

$$\bar{D} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} D(x_i, x_{i'}) = \sum_{j=1}^{p} \omega_j \bar{d}_j$$

$$\bar{d}_j = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_j(x_{ij}, x_{i'j})$$

Quindi è proprio la quantità $\omega_j \bar{d}_j$ a essere il contributo medio che apporta l'attributo j-mo, e quindi per fare in modo che ogni attributo abbia la stessa influenza sul valore finale va imposto che sia $\omega_j \sim \frac{1}{\bar{d}_j}$. Particolarmente interessante è il caso in cui decidiamo di usare il quadrato della distanza, in quanto otteniamo che

$$\bar{d}_j = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} (x_{ij} - x_{i'j})^2 = 2 \cdot \mathrm{var}_j$$

dove var_j indica la stima della varianza di X_j.
In generale imporre un uguale peso a ogni attributo può sembrare un'idea ragionevole, ma a volte questa strategia si rivela controproduttiva: questo avviene ad esempio nel caso in cui ci siano attributi "più forti" la cui differenza si rivela più importante ai fini della creazione di sottogruppi dei dati che siano più veritieri. Per questo motivo variabili con maggiore rilevanza andrebbero trattate in modo da riservare loro maggiore influenza sul valore di differenza finale.

1.4 Missing values

Talvolta capita che nei dati iniziali siano presenti dei valori mancanti. Possiamo risolvere questo problema in due modi.

- Un modo consiste nell'omettere tutte quelle paia di osservazioni $x_{ij}, x_{i'j}$ per le quali almeno uno dei due valori risulta mancante. Tuttavia questa via presenta un problema nel momento in cui non ci sono osservazioni comuni, in quanto rimaniamo senza dati da analizzare.

- Un'altra via è quella di assegnare ai valori mancanti la media o la mediana dei valori dati. Talvolta, se in presenza di variabili categoriche, si aggiunge la categoria "missing".

2 K-means Clustering

2.1 Presentazione del metodo

Passiamo ora ad analizzare più nel dettaglio i metodi di clusterizzazione più diffusi. Il K-means clustering è uno dei metodi più utilizzati: consiste nel dividere il dataset in un numero prefissato K di cluster che siano distinti e non sovrapposti, in modo che ogni osservazione venga assegnata a uno e un solo cluster. Analizziamo nel dettaglio come funziona: prima di procedere, introdurremo della notazione.
Siano $C_1, \dots, C_K$ gli insiemi contenenti gli indici in ciascuno dei cluster. Vale che:

$$C_1 \cup \cdots \cup C_K = \{1, \dots, n\}$$
$$C_k \cap C_{k'} = \emptyset \quad \forall k \neq k'$$

In particolare cerchiamo una clusterizzazione in corrispondenza della quale otteniamo la minor variazione intra-cluster possibile. Questa è una misura $W(C)$ di quanto le osservazioni all'interno di un cluster differiscono tra di loro. Il problema che cerchiamo di risolvere è dunque

$$\min_{C_1,\dots,C_K} \{W(C)\}$$

Dobbiamo definire esplicitamente W: un modo naturale di fare ciò è come segue

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} d(x_i, x_{i'})$$

ovvero diamo come stima la somma di tutte le possibili distanze tra punti presenti all'interno dello stesso cluster. A questo punto, siccome questo metodo viene utilizzato in presenza di variabili di tipo quantitativo, si utilizza come misura di dissimilarità il quadrato della distanza euclidea

$$d(x_i, x_{i'}) = ||x_i - x_{i'}||^2$$

Quindi possiamo riscrivere

$$\begin{aligned} W(C) &= \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} ||x_i - x_{i'}||^2 \\ &= \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - \bar{x}_k||^2 \end{aligned}$$

dove abbiamo indicato con $\bar{x}_k = (\bar{x}_{1k}, \dots, \bar{x}_{pk})$ il vettore con componenti la media delle componenti nel k-mo cluster e con N_k il numero di indici presenti nel cluster k. L'idea è quindi di minimizzare la media della dissimilarità presenti all'interno dei vari gruppi.

Risolvere questo problema è in generale molto difficile, in quanto abbiamo almeno K^n modi di dividere i nostri n dati in K cluster: per superare ciò si "spezza" in due parti il problema. In primo luogo osserviamo che dato un insieme di osservazioni S otteniamo che la media dei dati soddisfa

$$\bar{x}_S = \underset{m}{\operatorname{argmin}} \sum_{i \in S} ||x_i - m||^2$$

Quindi minimizzare $W(C)$ equivale a minimizzare il seguente problema:

$$\min_{C_1,\ldots,C_K,m_1,\ldots,m_K} \sum_{k=1}^{K} N_k \sum_{C(i)=k} ||x_i - m_k||^2$$

dove i punti $m_1, \ldots, m_K$ sono detti centroidi.
Guardando superficialmente la nostra espressione sembra che abbiamo complicato il problema: tuttavia questa riformulazione può essere risolta velocemente attraverso il seguente algoritmo. In primo luogo assegniamo in maniera casuale ogni indice a un cluster (in alternativa possiamo anche scegliere in maniera casuale i primi centroidi e partire dal passo 2 dell'algoritmo). Successivamente alterniamo i seguenti passi fino a quando l'assegnazione degli indici non rimarrà invariata tra un'iterazione degli step 1-2 e quella successiva.

1. Dati i cluster $C_1, \ldots, C_K$ si calcolano i nuovi centroidi $m_1, \ldots, m_K$ come la media dei punti all'interno dei cluster.

2. Dati i centroidi $\{m_1, \ldots, m_K\}$, assegniamo l'osservazione i al cluster il cui centroide risulta più vicino a i stesso, ovvero assegniamo il cluster $C(i)$ dato da
$$C(i) = \underset{1 \leq k \leq K}{\operatorname{argmin}} ||x_i - m_k||^2$$

Nonostante l'apparente semplicità, questo algoritmo presenta un problema: poichè dipende fortemente dalla scelta dei valori iniziali, può capitare di fermarsi in corrispondenza di punti ottimali solo locali e non globali. Per risolvere ciò basta (una volta scelto il valore appropriato di K), eseguire più volte l'algoritmo scegliendo diverse configurazioni iniziali per poi andare a scegliere quella che presenta caratteristiche migliori.
Un ultimo problema di fondamentale importanza da affrontare è la scelta del parametro K. In alcuni contesti questo può essere già prefissato: forniamone un esempio. Una compagnia pubblicitaria vuole cercare un modo di dividere il pubblico in modo da assegnare a ognuno dei suoi K dipendenti una diversa area di lavoro: in questo caso il numero di cluster K corrisponde al numero di dipendenti, e quindi è già fornito. Capita spesso però di non sapere già a priori quanti gruppi vogliamo formare, quindi dobbiamo trovare una procedura che ne dia una stima adatta. In questo caso solitamente si ripete la clusterizzazione in corrispondenza di una serie di valori di $K \in \{1, 2, \ldots, K_{\max}\}$, e si calcolano i corrispondenti valori di perdita $\{W_1, \ldots, W_{k_{\max}}\}$. Si procede poi con il metodo

"del gomito": si plottano questi valori di perdita come valori dell'ordinata in un grafico dove l'asse delle ascisse è data dai valori K utilizzati, e si cerca il valore K^* in corrispondenza del quale si osserva una piega in questo grafico. L'idea dietro questo ragionamento è la seguente: se il numero di cluster realmente presenti è K^* allora utilizzando un numero di gruppi inferiore a quello ideale il nostro algoritmo commetterà errori elevati. D'altra parte, utilizzando un numero di gruppi superiore a quello necessario, l'errore intra-cluster continuerà solitamente a diminuire (in quanto ogni osservazione all'interno di un gruppo è veramente simile), ma in ogni caso la diminuzione sarà meno marcata rispetto a quanto avviene in corrispondenza dei valori inferiori a K^* in quanto già la naturale divisione permette di ottenere valori molto contenuti per la differenza totale. Si crea quindi un gomito in corrispondenza del valore ottimale, che è ciò che vogliamo sfruttare nella nostra analisi.

2.2 K-means su Iris Dataset

Analizziamo nel concreto quanto abbiamo appena spiegato: studiamo ad esempio l'"Iris Dataset" che possiamo importare sfruttando la libreria di Python "Scikit Learn" (si veda [4]). Grazie a questo dataset possiamo studiare a che specie (setosa, versicolor o virginica) un fiore di Iris appartiene in base alla lunghezza e alla larghezza del petalo (si veda la Figura 1).

Siccome nel dataset ci viene fornita anche la divisione nelle varie classi, pos-

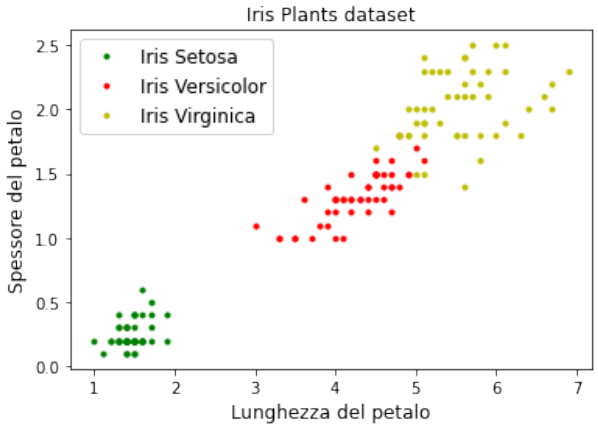

Figura 1: Iris dataset

siamo dire che a livello teorico ci aspettiamo di ottenere tre cluster applicando K-means clustering a questo dataset: tuttavia, osseervando il grafico, notiamo che sarà possibile (e ragionevole) che versicolor e virginica vengano fusi nello stesso cluster in quanto i dati corrispondenti a queste classi risultano molto vicini nello spazio. Eseguendo clusterizzazione in corrispondenza dei valori di $K = 2, 3, 4$ otteniamo i risultati nelle Figure 2, 3 e 4, che ci confermano quanto

abbiamo appena detto. Possiamo anche analizzare come avremmo scelto il nu-

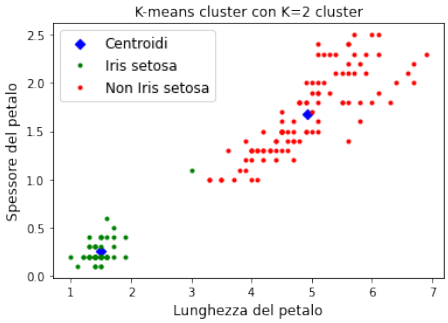

Figura 2: K-means cluster per l'Iris dataset realizzato con $K = 2$ cluster.

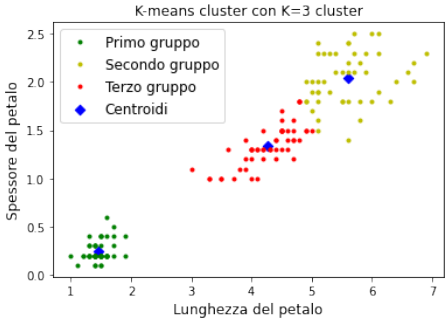

Figura 3: K-means cluster per l'Iris dataset realizzato con $K = 3$ cluster.

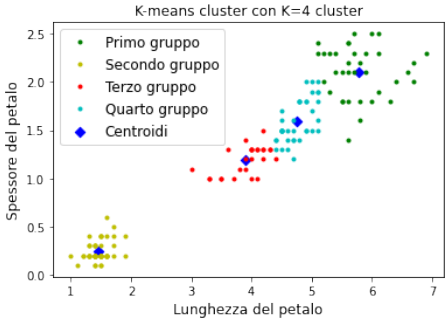

Figura 4: K-means cluster per l'Iris dataset realizzato con $K = 4$ cluster.

mero di cluster senza avere informazioni preliminari: per procedere si esegue il plot della variazione intra-cluster per più valori di K, ottenendo i risultati nella

Figura 5. A questo punto si osserva che in corrispondenza di $K = 2$ otteniamo il gomito descritto precedentemente: quindi, senza avere ulteriori informazioni, si sceglie di utilizzare due cluster. Tutto ciò sembra essere in disaccordo con la presenza delle tre specie nel dataset: bisogna tuttavia tenere in conto che stiamo lavorando con tecniche non supervisionate e quindi il risultato risulta comunque un buon punto di partenza.

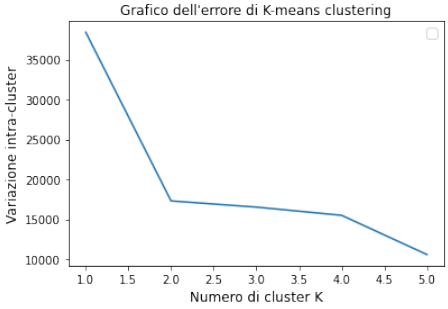

Figura 5: Plot della variazione intra-cluster per l'Iris dataset in corrispondenza di valori di K compresi tra 1 e 5.

3 Clustering gerarchico

Il clustering gerarchico è un tipo di clustering nel quale lo user non si occupa di specificare in anticipo il numero di cluster da utilizzare, ma fornisce solamente una misura di dissimilarità tra gruppi disgiunti di osservazioni. Questo metodo, come suggerisce il nome, fornisce una rappresentazione gerarchica in cui i cluster a ogni livello sono creati unendo cluster al livello inferiore. In particolare, al livello più basso ogni cluster contiene un solo elemento, mentre al livello più alto è presente un solo cluster. Ci sono due tipologie di paradigmi.

- Agglomerativo (bottom up): si parte dal fondo e si uniscono ricorsivamente i due cluster con la dissimilarità più bassa.
- Divisivo (top-down): si parte dalla cima e si divide uno dei cluster già esistenti in due sottogruppi in modo che questi abbiano la maggiore differenza intergruppo possibile.

Completata la costruzione, lo user decide quale livello rappresenta la divisione naturale attraverso un processo che descriveremo nel seguito.
Uno dei modi più semplici di rappresentare questo procedimento è attraverso un albero binario, all'interno del quali i nodi rappresentano i gruppi di osservazioni mentre le radici dell'albero l'intero dataset. In particolare, i nodi terminali rappresentano i singoli dati (cluster singoletto). Ogni nodo non terminale ha due figli, che nel clustering agglomerativo rappresentano i due gruppi che si sono

uniti nel nodo superiore.
I metodi agglomerativi e alcuni metodi divisivi godono di una proprietà di monotonia: la dissimilarità tra due gruppi che si fondono aumenta all'aumentare del livello della fusione. Quindi l'albero può essere plottato in modo che l'altezza dei nodi sia proporzionale alla dissimilarità intergruppo tra i duoi figli. Per rendere graficamente leggibile il tutto, si plottano i dati iniziali ad altezza zero e si costruiscono poi i vari collegamenti scalando l'altezza dei vari collegamenti per rendere quanto appena detto: questo tipo di rappresentazione è detta dendrogramma.
Costruito il dendrogramma, si può dividere il dataset in cluster distinti tagliando quest'ultimo orizzontalmente a una data altezza: in particolare, i vari gruppi sono dati dalle linee verticali che intersecano il nostro taglio. Più un gruppo si unisce ad alto livello, più diventa candidato per una clusterizzazione naturale. Notiamo poi che la divisione potrebbe avvenire a diversi livelli, portando a una gerarchia nei cluster.
Spesso i dendrogrammi vengono usati per ottenere un riassunto grafico dei dati: tuttavia va sottolineato, come vedremo, che la struttura e la resa finale dipendono fortemente dalla misura di dissimilarità scelta. Sono quindi un utile strumento di rappresentazione visuale ma vanno analizzati con cura.

3.1 Clustering Agglomerativo

3.1.1 Presentazione del modello

Il clustering agglomerativo parte dall'assumere che ogni osservazione rappresenti un singolo cluster. Successivamente, a ognuno degli $N-1$ step del procedimento i due cluster che sono considerati più simili vengono uniti in un unico cluster, fino alla creazione di un unico grande gruppo finale. Abbiamo quindi bisogno di definire la dissimilarità tra due gruppi di osservazioni.
Siano G, H due gruppi. Calcoleremo la loro diversità usando la dissimilarità $d_{ii'}$ tra i membri i di G e i' di H: vediamo diversi modi di procedere.

- Single Linkage o nearest-neighbour technique (SL). In questo metodo consideriamo come diversità tra i due cluster quella che corrisponde al paio di osservazioni che sono più vicine tra loro, ovvero
$$d_{SL}(G,H) = \min_{i\in G, i'\in H} d_{ii'}$$
- Complete linkage o furthest-neighbor (CL). Prendiamo come valore per la dissimilarità quello che corrisponde alla dissimilarità tra le due osservazioni più lontane tra loro, ovvero
$$d_{L}(G,H) = \max_{i\in G, i'\in H} d_{ii'}$$
- Group Average (GA). Usiamo come valore finale la media delle dissimilarità, ovvero
$$d_{GA}(G,H) = \frac{1}{N_G N_H}\sum_{i\in G}\sum_{i'\in H} d_{ii'}$$

dove N_G, N_H indicano il numero di elementi in ciascun cluster.

Analizziamo più nel dettaglio i metodi proposti. In primo luogo possiamo osservare che se i dati presentano una forte tendenza alla clusterizzazione i metodi produranno risultati simili. Interessante è invece analizzare cosa succede se ciò non accade. Utilizzando il Single Linkage, due cluster sono considerati vicini se i due elementi più vicini lo sono, indipendentemente dagli altri. Quindi, se si scelgono livelli di soglia bassi, si avrà tendenza a collegare una serie di osservazioni che sono intermediatamente vicine: questo fenomeno è detto chaining, ed è spesso considerato un difetto di questo metodo.
Per effettuare una migliore analisi, definiamo il diametro D_G di un gruppo di osservazioni come

$$D_G = \max_{i \in G, i' \in G} d_{ii'}$$

Abbiamo che il single linkage produce cluster con diametri molto grandi. D'altra parte il complete linkage produce ovviamente l'effetto opposto, creando cluster in cui le osservazioni sono più vicine a elementi di altri gruppi rispetto che ai propri. Il Group Average si prefigge invece di realizzare un compromesso creando cluster relativamente compatti e che sono relativamente distanti fra loro: tuttavia un problema che si presenta utilizzando questo metodo consiste nel fatto che i risultati dipendono fortemente dalla scala numerica con cui si rappresentano le dissimilarità $d_{ii'}$. Infatti applicando una funzione monotona strettamente crescente h alle dissimilarità, i risultati ottenuti con questi nuovi valori mutano andando a modificare il risultato del metodo. Questo fatto non accade utilizzando il Single Linkage e il Complete Linkage, il cui risultato dipende solo dall'ordine delle dissimilarità.

3.1.2 Alcuni esempi pratici

Vediamo ora di spiegare meglio quanto appena visto attraverso alcuni esempi pratici. Consideriamo come primo dataset ancora l'"Iris dataset" utilizzato nella Sezione 2.2. Vediamo inizialmente una rappresentazione grafica di un dendrogramma: per costruire un esempio, ci limitiamo a prendere in considerazione nove dati (tre per classe), nel dettaglio quelli in Figura 6, così che il tutto risulti graficamente leggibile. Utilizzando le tre diverse tipologie di dissimilarità otteniamo i risultati in Figura 7, 8 e 9. In questa particolare configurazione i tre dendrogrammi risultano molto simili, in quanto i tre cluster presenti sono ben distanziati tra loro e i dati al loro interno sono molto vicini.

Torniamo a questo punto all'intero dataset, su cui possiamo eseguire clustering gerarchico: scegliendo di utilizzare 3 cluster (che sono quelli che sappiamo esserci nella realtà), otteniamo i risultati nelle Figure 10, 11 e 12. Possiamo osservare che in questo caso Complete e Average Linkage risultano molto efficaci nel catturare i gruppi realmente presenti, mentre Single Linkage riconosce un singoletto come uno dei cluster: questo segue dal fatto che i due cluster che corrispondono a Iris Versicolor e Virginica presentano dati molto vicini tra loro e quindi, come spiegato precedentemente, questa scelta di dissimilarità risulta essere poco efficace per la nostra analisi.

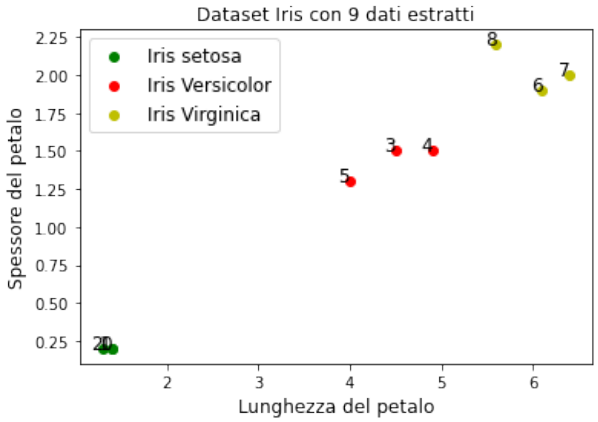

Figura 6: Grafico dei dati che utilizziamo per costruire i dendrogrammi semplificati

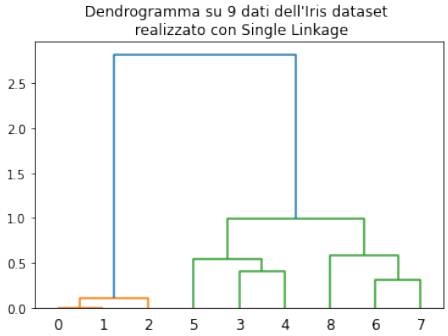

Figura 7: Dendrogramma costruito con Single Linkage sui dati in Figura 6.

Passiamo ora ad analizzare altri due esempi: il primo che prendiamo in considerazione è quello in Figura 13, costruito utilizzando l'apposita funzione `make_moons` presente nella libreria Python "scikit-learn" (si veda [6]). Tenendo in considerazione che abbiamo due gruppi di osservazione per costruzione, possiamo eseguire clustering gerarchico su questo dataset con le tre diverse misure di dissimilarità che abbiamo studiato e, scegliendo di usare due cluster, ottenere i risultati nelle Figure 14, 15 e 16. Osserviamo come in questo caso l'unica misura che risulta efficace nel cogliere la divisione dei cluster è il Single Linkage: questo in quanto i due gruppi presentano dati che sono molto distanti, se considerati tra i due cluster, ma molto vicini all'interno dei cluster stessi.

Infine prendiamo in considerazione il dataset in Figura 17, costruito a partire dall'apposita funzione `make_circles` presente nella libreria Python "scikit-learn" (si veda [1]). Eseguendo clustering gerarchico e dividendo in due cluster otteniamo i risultati nelle Figure 18, 19 e 20. In questo particolare caso nessuna

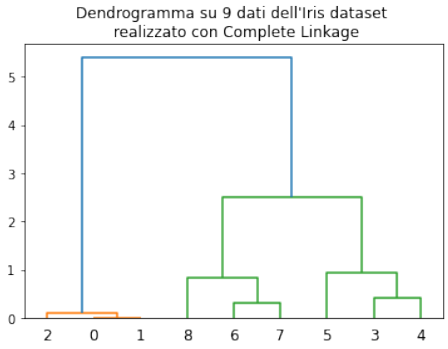

Figura 8: Dendrogramma costruito con Complete Linkage sui dati in Figura 6.

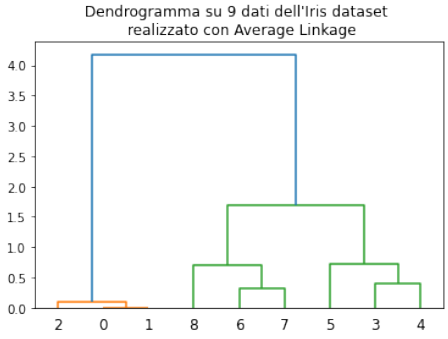

Figura 9: Dendrogramma costruito con Average Linkage sui dati in Figura 6.

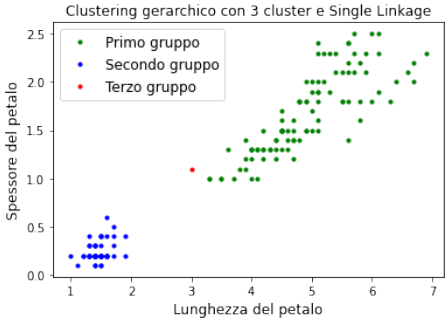

Figura 10: Clustering gerarchico sull'Iris dataset realizzato con Single Linkage.

delle misure presentate sopra risulta efficace nel cogliere la divisione nei due cluster, in quanto i dati risultano essere abbastanza sparsi e vicini sia all'interno dei cluster stessi che in relazione ai dati dell'altro gruppo: si devono quindi utilizzare metodi diversi da quelli qui esposti, come ad esempio Clustering Spettrale.

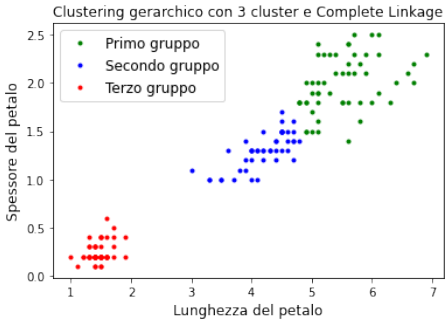

Figura 11: Clustering gerarchico sull'Iris dataset realizzato con Complete Linkage.

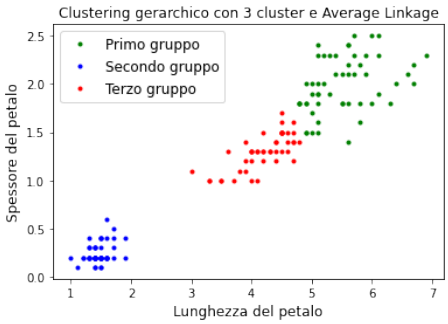

Figura 12: Clustering gerarchico sull'Iris dataset realizzato con Average Linkage.

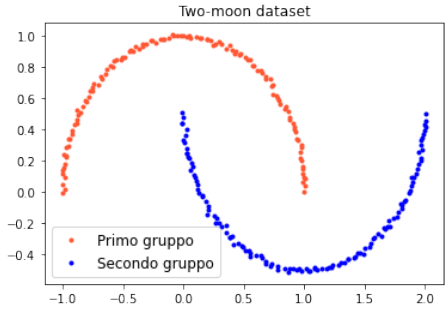

Figura 13: Two Moon dataset costruito con valori dei parametri `n_samples`= 250, `noise`= 0.01 e `random_state`= 0.

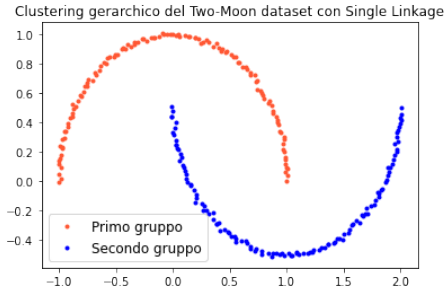

Figura 14: Clustering gerarchico sul dataset Two Moon realizzato con Single Linkage.

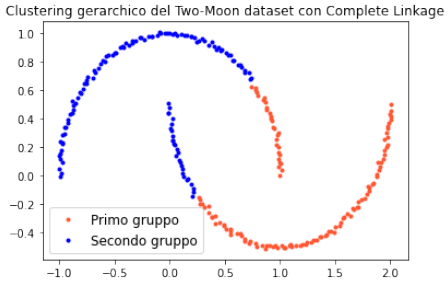

Figura 15: Clustering gerarchico sul dataset Two Moon realizzato con Complete Linkage.

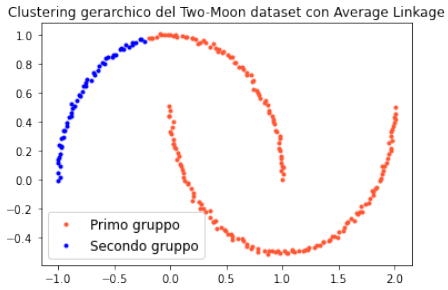

Figura 16: Clustering gerarchico sul dataset Two Moon realizzato con Average Linkage.

3.2 Clustering Divisivo

Il clustering divisivo è un metodo che considera inizialmente l'intero data set come un singolo cluster, e procede dividendo ricorsivamente ogni gruppo in due sottogruppi. Nonostante questo algoritmo sia stato finora poco studiato a livello teorico, è interessante analizzarlo in quanto si rivela utile quando siamo

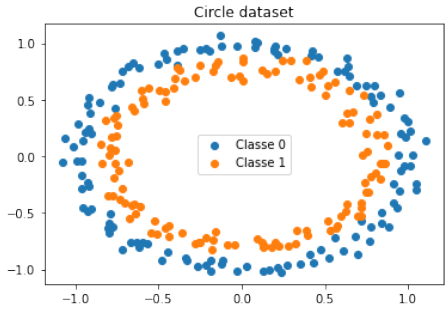

Figura 17: Circles dataset costruito con valori dei parametri `n_samples`= 250, `noise`= 0.05 e `random_state`= 100.

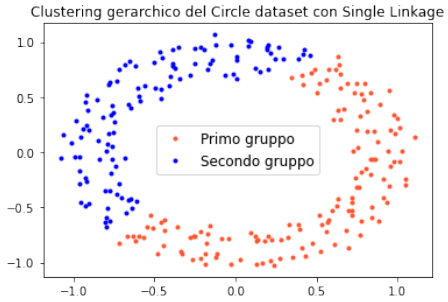

Figura 18: Clustering gerarchico sul dataset Circles realizzato con Single Linkage.

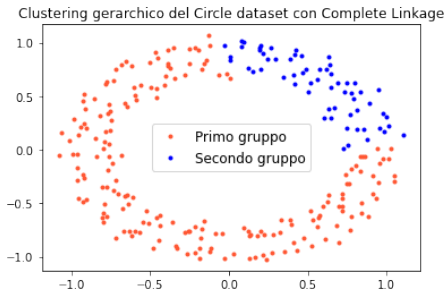

Figura 19: Clustering gerarchico sul dataset Circles realizzato con Complete Linkage.

interessati a dividere i dati in un piccolo numero di gruppi.
Analizziamo nel dettaglio la procedura: dato un cluster, dobbiamo precisare come effettuarne la divisione. Una prima idea potrebbe essere quella di applicare metodi di clusterizzazione già visti come K-means (con $K = 2$): tuttavia questo

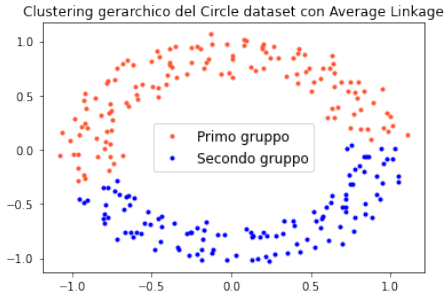

Figura 20: Clustering gerarchico sul dataset Circles realizzato con Average Linkage.

approccio dipende eccessivamente dalla configurazione iniziale di ogni passo e potrebbe non restituire un dendrogramma monotono.
Per superare questo problema, esiste un metodo alternativo proposto da Macnaughton e Smith. Si procede nel seguente modo: all'inizio si mettono tutte le osservazioni in un singolo cluster G, per poi scegliere l'osservazione la cui dissimilarità media dalle altre osservazioni è la più alta. Questa andrà a formare il primo elemento del secondo cluster H: a questo punto ogni osservazione in G la cui distanza media da quelle in H, meno quella dalle altre osservazioni in G è negativa viene trasferita in H. Otteniamo quindi due cluster figli dell'originale che rappresentano il secondo livello della nostra gerarchia. A questo punto si ripete la procedura su uno dei due cluster del livello ottenuto, fino a quando non si ottengono cluster con un solo elemento.
Rimane ora un ultimo problema da analizzare, ovvero la scelta del cluster da dividere: una soluzione consiste nel prendere o il cluster con diametro maggiore o quello che possiede la maggiore dissimilarità media tra i propri elementi, data da

$$\bar{d}_G = \frac{1}{N_G} \sum_{i \in G} \sum_{i' \in G} d_{ii'}$$

4 Clusterizzazione di documenti testuali

Dopo aver elencato alcuni metodi di clusterizzazione a livello teorico, proviamo a vederne un'applicazione pratica attraverso lo studio della clusterizzazione di documenti testuali. Il primo ingrediente necessario per procedere è capire come rappresentare un documento testuale come dato processabile da un computer: uno dei metodi più utilizzati per fare ciò è la rappresentazione "bag of words". Consideriamo un dizionario, come ad esempio quello inglese in quanto più semplice di quello italiano, che contiene M parole: le ordiniamo, e a ogni documento associamo un vettore di lunghezza M il cui elemento i-mo indica il numero di volte in cui compare la parola i-ma del dizionario. Abbiamo due piccole osservazioni da fare: per prima cosa, vista la già grande dimensione dei dati, ci si

limita di solito a registrare la presenza delle parole più frequenti nel dizionario, di solito se ne considerano 10000. Inoltre, con il procedimento appena descritto otteniamo un vettore i cui valori dipendono dalla lunghezza del documento: per superare questo problema si passa alle frequenze relative, ovvero si dividono gli elementi di ogni vettore per il numero di parole presenti nel documento stesso. Un altro espediente che si utilizza per eliminare la dipendenza dei dati dalla lunghezza del documento consiste nell'utilizzare la metrica del coseno al posto di quella euclidea: dati due vettori A, B definiamo la similarità del coseno come

$$S_C(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

dove $\cdot$ indica il prodotto scalare. Questa quantità altro non è che il coseno dell'angolo formato dai due vettori considerati: assume valori nell'intervallo $[-1, 1]$ e non dipende dalla norma dei due vettori, ma solo dalla loro "posizione spaziale".

4.1 Applicazione ai Film Dreamworks

Per concludere la nostra trattazione forniamo un esempio concreto di come funziona la clusterizzazione testuale: per vederne l'implementazione si veda [2]. Prendiamo da Wikipedia la trama di 18 film Dreamworks (in inglese): Shrek, Shrek 2, Shrek 3, Shrek 4, Il Gatto con gli stivali, Il gatto con gli stivali 2, Kung fu Panda, Kung fu Panda 2, Kung fu Panda 3, Cattivissimo Me, Cattivissimo Me 2, Cattivissimo Me 3, Minions, Minions 2, Madagascar, Madagascar 2, Madagascar 3 e Madagascar 4. Il nostro obiettivo è crearne una clusterizzazione, che possiamo ovviamente valutare sulla base di un criterio logico.
Creiamo quindi un vettore che contiene la rappresentazione bag of words dei nostri dati di partenza, dove decidiamo di togliere da (tutti) i vettori le componenti che corrispondono a parole che compaiono in più dell'80 % delle volte. Su questo possiamo eseguire clusterizzazione gerarchica utilizzando sia metrica del coseno che euclidea, entrambe con diverse tipologie di linkage. Otteniamo i risultati nelle Figure 21 - 26.

Possiamo osservare che i risultati ottenuti con metrica del coseno sono molto buoni: effettivamente uniscono i film appartenenti alla stessa saga. In generale anche quelli ottenuti con metrica euclidea sono soddisfacenti, anche se le misure di dissimilarità risultano abbastanza vicine.
A questo punto sappiamo che, per come abbiamo scelto i film, ci sono 4 cluster naturali (che corrispondono alle quattro saghe di Shrek, Kung fu Panda, Cattivissimo Me e Madagascar): possiamo provare a eseguire K-means clustering con $K = 4$. Otteniamo quindi che l'algoritmo divide i film nei gruppi in Tabella 1: i risultati sono ancora molto buoni. Quindi i nostri modelli sono efficaci nel catturare le differenze a partire dai testi delle trame.

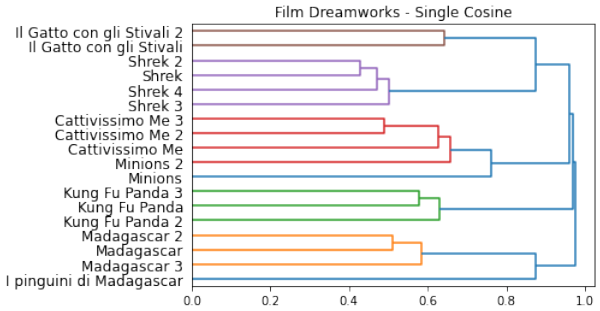

Figura 21: Clusterizzazione gerarchica della trama dei film costruita con metrica del coseno e Single Linkage.

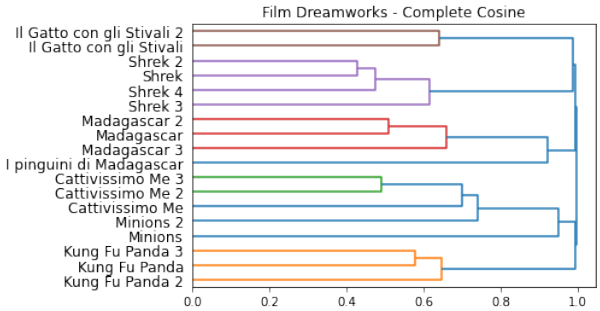

Figura 22: Clusterizzazione gerarchica della trama dei film costruita con metrica del coseno e Complete Linkage.

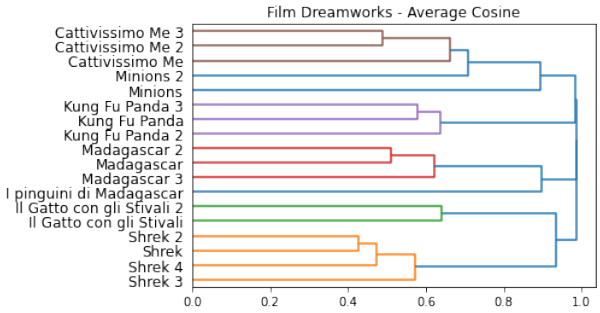

Figura 23: Clusterizzazione gerarchica della trama dei film costruita con metrica del coseno e Average Linkage.

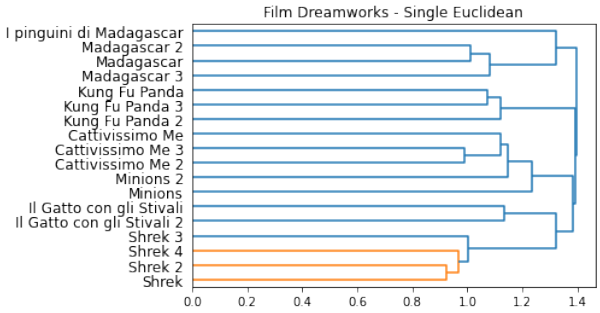

Figura 24: Clusterizzazione gerarchica della trama dei film costruita con metrica euclidea e Single Linkage.

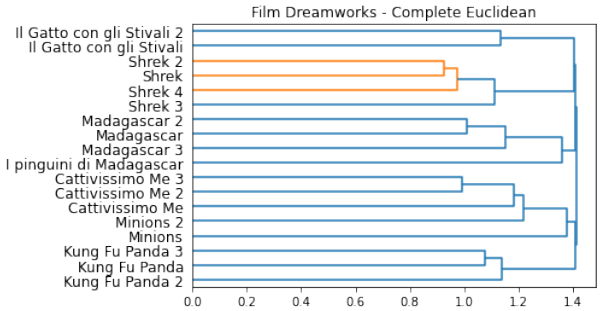

Figura 25: Clusterizzazione gerarchica della trama dei film costruita con metrica euclidea e Complete Linkage.

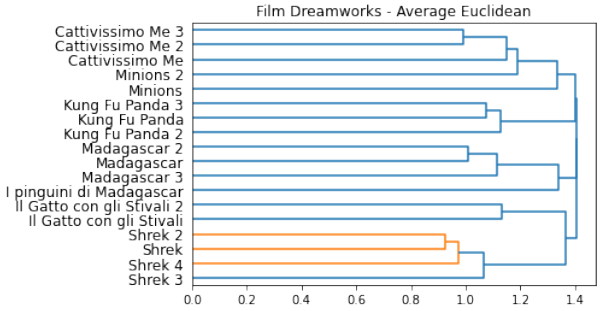

Figura 26: Clusterizzazione gerarchica della trama dei film costruita con metrica euclidea e Average Linkage.

Gruppo	Film
Primo gruppo	Kung fu Panda, Kung fu Panda 2, Kung fu Panda 3
Secondo gruppo	Cattivissimo Me, Cattivissimo Me 2, Cattivissimo Me 3, Minions, Minions 2
Terzo gruppo	Shrek, Shrek 2, Shrek 3, Shrek 4, Il gatto con gli stivali, Il gatto con gli stivali 2
Quarto gruppo	Madagascar, Madagascar 2, Madagascar 3, I pinguini di Madagascar

Tabella 1: Divisione dei film con K-means clustering e $K = 4$ cluster.

Bibliografia

[1] *Circles*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_circles.html.

[2] Capelli Daniele. *Codice per il Percorso di Eccellenza*. URL: https://github.com/dcapelli02/Percorso-di-Eccellenza.git.

[3] Trevor Hastie, Robert Tibshirani e Jerome Friedman. *The elements of statistical learning. Data mining, inference, and prediction*. English. Springer Ser. Stat. New York, NY: Springer, 2001. ISBN: 0-387-95284-5.

[4] *Iris Dataset*. URL: https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html#sphx-glr-auto-examples-datasets-plot-iris-dataset-py.

[5] Gareth James et al. *An introduction to statistical learning. With applications in Python*. English. Springer Texts Stat. Cham: Springer, 2023. ISBN: 978-3-031-38746-3; 978-3-031-39189-7; 978-3-031-38747-0. DOI: 10.1007/978-3-031-38747-0.

[6] *Two Moon*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html.