Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Matematica

IL CLUSTERING NELL'INFORMATION RETRIEVAL

Tesi di Laurea in Analisi Numerica

Relatrice:
Chiar.ma Prof.ssa
Valeria Simoncini

Presentata da:
Chiara Fusaroli Casadei

Anno Accademico 2023/2024

Ai miei genitori e a mio fratello Enrico,
ad Anna, Margherita, Marianna, Alice,
Corinna e Rachele, le amiche di sempre,
agli amici del Dipartimento, il cuore di questo percorso

Introduzione

Il crescente sviluppo di Internet e l'avanzamento delle tecnologie hanno messo a disposizione una enorme quantità di dati in continuo aumento e aggiornamento, siano essi in forma di testi, immagini, video o audio. Questo va di pari passo con la necessità di metodi sempre più efficienti per ottenere informazioni utili da tali risorse disponibili. Negli anni si è dunque sviluppato il campo dell'*Information Retrieval* (IR) che, tradotto letteralmente, significa "reperimento di informazione" e che si può definire come "l'insieme delle tecniche per il reperimento di materiale (tipicamente documenti) di natura non strutturata (tipicamente testi), proveniente da un'ampia collezione, che soddisfi un bisogno informativo dell'utente" [3]. Pertanto, fanno parte di questo ambito anche tutte quelle tecniche volte alla rappresentazione, memorizzazione, organizzazione dei dati, oltre ai metodi di accesso a quelli rilevanti per il bisogno informativo.

Il presente elaborato si propone di applicare la cluster analysis all'information retrieval. Più precisamente, verrà trattata la rappresentazione vettoriale dei documenti e si spiegherà come alcune tecniche di clustering siano più adatte di altre in questo contesto (con una particolare attenzione per l'algoritmo delle k-medie sferiche), nonché come una suddivisione in cluster possa essere sfruttata per una più facile gestione di grandi moli di dati. Siccome i documenti, così come il bisogno informativo dell'utente, rientrano in quella categoria di dati detti "non strutturati", cioè dati che non possiedono una struttura tale da essere facilmente analizzabile e trattabile da una macchina, è necessario sviluppare dei modelli che ne rendano possibile l'elaborazione. Uno dei metodi più diffusi per la rappresentazione dei documenti è il modello vettoriale, il quale considera ogni documento e il bisogno informativo come un insieme di parole chiave, codificabile in un vettore; ne segue che la collezione di documenti dalla quale si vuole estrarre l'informazione può essere rappresentata da una matrice in cui ciascuna colonna corrisponde ad un documento. Mentre tale modello è soggetto ai problemi derivanti dalla complessità e dall'ambiguità del linguaggio naturale, una sua variante, la Latent Semantic Indexing (LSI), mira a cogliere le relazioni semantiche all'interno della collezione sfruttando la Decomposizione

ai Valori Singolari (SVD) della matrice rappresentante la collezione. D'altra parte, il modello vettoriale permette di fare uso di strutture e proprietà algebriche adatte alla risoluzione dei problemi di document retrieval, clustering e classificazione, ovvero la ricerca dei documenti più rilevanti per la richiesta dell'utente, la determinazione di raggruppamenti omogenei (*cluster*) dei documenti sulla base del loro contenuto e l'assegnazione di un nuovo elemento al gruppo più opportuno. In particolare, all'interno di un sistema di information retrieval il clustering permette di organizzare la collezione di documenti per facilitarne la comprensione all'utente e la navigazione, ma anche di rendere più efficiente il document retrieval conducendo la ricerca dei documenti rilevanti solo all'interno del cluster più attinente alla richiesta, anziché sull'intera vasta collezione. Infine, la conoscenza di una suddivisione in cluster della collezione di documenti può essere utilizzata per ridurre le dimensioni dei dati che il sistema di information retrieval deve gestire, il che è fondamentale per poter manipolare in modo efficiente le grandi quantità di dati odierne.

La tesi è strutturata nel seguente modo:

- Il primo capitolo analizza il modello vettoriale per la rappresentazione dei documenti: la costruzione, i limiti e il metodo di confronto tra documenti. Vengono inoltre riportati alcuni risultati sulla Decomposizione ai Valori Singolari e trattata la tecnica della Latent Semantic Indexing.

- Il secondo capitolo descrive dapprima due delle principali tecniche di clustering (clustering gerarchico e clustering partizionale) e le mette a confronto nel contesto dei documenti, riportando come il clustering di tipo partizionale si riveli migliore dal punto di vista della complessità computazionale, della qualità del clustering e dell'interpretazione dei risultati. In particolare, viene posta l'attenzione sull'algoritmo di clustering partizionale delle k-medie sferiche, una variante del noto algoritmo delle k-medie, che utilizza la più adatta misura di similarità del coseno.

- Il terzo capitolo mostra come sfruttare il clustering di una collezione di documenti per ridurre le dimensioni dei dati mediante un problema ai minimi quadrati e la matrice dei centroidi. Inoltre, alcuni risultati sperimentali evidenziano come la riduzione di dimensioni così effettuata sia particolarmente efficace e migliore della SVD per il problema della classificazione di documenti.

Indice

Prerequisiti

Si introducono alcune definizioni e risultati che verranno utilizzati nella tesi.

Definizione (Norma di matrice)**.** *Una funzione* $\|\cdot\| : \mathbb{R}^{m\times n} \longrightarrow \mathbb{R}$ *è una norma di matrice se per ogni* $A, B \in \mathbb{R}^{m\times n}$ *soddisfa le seguenti proprietà:*

1. $\|A\| \geq 0$ *e* $\|A\| = 0$ *se e solo se* $A = 0$*, dove* 0 *indica la matrice nulla;*
2. $\|\alpha A\| = |\alpha|\|A\|, \forall \alpha \in \mathbb{R}$*;*
3. $\|A + B\| \leq \|A\| + \|B\|$*;*
4. $\|AB\| \leq \|A\|\|B\|$*.*

Definita la *norma Euclidea* (o norma-2) di un vettore reale $x = \big(x_1, \ldots, x_n\big)^T$ come

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2},$$

si definisce la norma-2 di una matrice $A \in \mathbb{R}^{m\times n}$ come:

$$\|A\|_2 = \max_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{\|Ax\|_2}{\|x\|_2}.$$

In generale, si può parlare di norma-p matriciale indotta dalla norma-p vettoriale:

$$\|A\|_p = \max_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{\|Ax\|_p}{\|x\|_p},$$

dove la norma-p vettoriale è

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p\Big)^{\frac{1}{p}}.$$

Definizione (Matrice ortogonale)**.** *Una matrice* $Q \in \mathbb{R}^{n\times n}$ *si dice ortogonale se* $Q^TQ = QQ^T = I_n$*.*

In particolare, se $Q \in \mathbb{R}^{n\times n}$ è una matrice ortogonale, le sue colonne costituiscono una base ortonormale di $\mathbb{R}^n$. Una importante proprietà delle matrici ortogonali è che esse sono isometrie: sia $Q \in \mathbb{R}^{n\times n}$ matrice ortogonale, allora per ogni $x \in \mathbb{R}^n$ si ha $\|Qx\|_2 = \|x\|_2$. Infatti, $\|Qx\|_2^2 = x^T Q^T Q x = x^T x = \|x\|_2^2$.

Si indica $\text{spec}(A) = \{\lambda \in \mathbb{C} : \lambda \text{ autovalore di } A\}$.

Proposizione. *Siano $A \in \mathbb{R}^{n\times n}$ simmetrica e $\lambda_{\max} = \arg\max_{\lambda \in spec(A)} |\lambda|$. Allora*

$$\|A\|_2 = |\lambda_{\max}|.$$

Dimostrazione. Si consideri la decomposizione spettrale della matrice A: $A = Q\Lambda Q^T$ con Q matrice ortogonale e $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ matrice diagonale contenente gli autovalori di A. Allora, sfruttando l'ortogonalità di Q, si ha

$$\|A\|_2 = \max_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{\|Ax\|_2}{\|x\|_2} = \max_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{\|Q\Lambda Q^T x\|_2}{\|Q^T x\|_2} = \max_{\substack{y\in\mathbb{R}^n \\ y\neq 0}} \frac{\|\Lambda y\|_2}{\|y\|_2}.$$

Per ogni $x \in \mathbb{R}^n, x \neq 0$ si ottiene

$$\frac{\|Ax\|_2^2}{\|x\|_2^2} = \frac{\|\Lambda y\|_2^2}{\|y\|_2^2} = \frac{\sum_{i=1}^n (\lambda_i y_i)^2}{\sum_{i=1}^n y_i^2} \leq \lambda_{\max}^2$$

cioè

$$\frac{\|Ax\|_2}{\|x\|_2} \leq |\lambda_{\max}|.$$

Dunque,

$$\|A\|_2 = \max_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{\|Ax\|_2}{\|x\|_2} \leq |\lambda_{\max}|$$

e il massimo è raggiunto per $q_{\max}$ autovettore associato a $\lambda_{\max}$. □

In particolare, da questa proposizione si deduce che se $\Lambda \in \mathbb{R}^{n\times n}, \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ con $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$ allora $\|\Lambda\|_2 = \lambda_1$.

Definizione (Distanza)**.** *Dato un insieme X, una distanza su X è una funzione $d : X \times X \longrightarrow \mathbb{R}$ tale che per ogni $P, Q, R \in X$ valgono le seguenti proprietà:*

1. *$d(P, Q) \geq 0$ e $d(P, Q) = 0$ se e solo se $P = Q$;*
2. *$d(P, Q) = d(Q, P)$;*
3. *$d(P, Q) \leq d(P, R) + d(R, Q)$.*

Per giudicare il grado di somiglianza di due oggetti P e Q di un insieme X, si può anche utilizzare una *misura di similarità*, la quale sarà una funzione $s : X \times X \longrightarrow \mathbb{R}$ di valore maggiore quanto più P e Q sono simili. Inoltre, devono valere le seguenti proprietà:

1. $s(P, Q) \geq 0$;
2. $s(P, Q) = s(Q, P)$;

Nel caso in cui s sia una misura di similarità a valori in $[0, 1]$, è possibile ricavare una misura di distanza d tramite la relazione

$$d(P, Q) = 1 - s(P, Q).$$

Capitolo 1

Rappresentazione dei documenti

I testi fanno parte di quella tipologia di dati detti "non strutturati", ovvero dati che non possiedono una struttura ben definita, facilmente analizzabile e trattabile da una macchina. Pertanto, se si vogliono estrarre informazioni da una vasta collezione di documenti testuali occorrono modelli che formalizzino la procedura di reperimento delle informazioni, specificando come rappresentare i documenti e le modalità di confronto fra essi. I più noti modelli utilizzati nell'information retrieval sono: modello booleano, modello vettoriale e modello probabilistico. Fra questi, di particolare rilievo è il modello vettoriale: esso, infatti, permette di sfruttare strutture e proprietà algebriche che si rivelano utili per problemi quali document retrieval, clustering e classificazione. Il document retrieval è la ricerca dei documenti più rilevanti per una "query" fra quelli presenti in una collezione, dove con il termine *query* ci si riferisce alla formalizzazione della richiesta dell'utente in una frase di ricerca. Il clustering consiste nel determinare raggruppamenti omogenei dei dati sulla base di una misura di similarità (o di distanza). La classificazione ha come scopo quello di assegnare un nuovo dato al gruppo più opportuno.

1.1 Il modello vettoriale

Il primo passo per definire il modello vettoriale prevede l'individuazione di termini significativi (parole chiave) estrapolati dai documenti di una data collezione e la costruzione, con questi, di un indice. La creazione di un tale indice richiede una fase di preprocessing dei testi, ossia una serie di passaggi quali:

- *Tokenization* ovvero la suddivisione del testo in unità più piccole (token), come frasi o singole parole;

- *Filtering* ovvero la rimozione di punteggiatura e stopwords (cioè quelle parole come articoli, congiunzioni, preposizioni, avverbi e similari che non hanno un particolare significato all'interno del documento);

- *Stemming* e *lemmatization* ovvero, rispettivamente, la rimozione dei suffissi da ciascuna parola e la conversione delle parole nelle loro forme base del vocabolario.

Ad ogni parola chiave è poi assegnato un peso in relazione a ciascun documento. A questo punto il modello vettoriale prevede di rappresentare i documenti come vettori in uno spazio vettoriale multidimensionale, in cui ciascuna dimensione corrisponde ad uno dei termini utilizzati per costruire l'indice e le componenti dei vettori sono i pesi. Dunque, dati m termini, un documento d viene codificato nel vettore $(w_1, \ldots, w_m)$ dove $w_i > 0$ se l'i-esimo termine è presente in d, $w_i = 0$ altrimenti. Il più semplice esempio è il peso binario, attribuendo peso 1 se la parola chiave è contenuta nel documento, 0 se è assente. Tipicamente, il peso è una funzione della frequenza[1] con cui il termine compare nel documento. Più precisamente, ciascun peso può essere:

- la sola frequenza della parola chiave nel documento (*term frequency*, TF) e quindi $d_{tf} = (tf_1, \ldots, tf_m)$, dove tf_i è la frequenza dell'i-esimo termine all'interno del documento;

- la frequenza della parola chiave nel documento moltiplicata per un fattore che riflette il valore globale della parola all'interno dell'intera collezione di documenti (*inverse document frequency*, IDF) andando a penalizzare la parola se essa ricorre in un gran numero di documenti . Quindi, $d_{tf\text{-}idf} = (tf\text{-}idf_1, \ldots, tf\text{-}idf_m)$, dove $tf\text{-}idf_i = tf_i \cdot idf_i$ avendo denominato idf_i il fattore globale dell'i-esimo termine e che vale $\log(n/df_i)$, con df_i numero di documenti contenenti l'i-esimo termine. Questo perchè termini presenti in molti documenti non sono significativi nel distinguere un documento dall'altro.

Anche le queries sono rappresentate come vettori nello stesso spazio dei documenti della raccolta.
Una collezione di n documenti descritta da m parole chiave è pertanto rappresentata da una matrice $A \in \mathbb{R}^{m \times n}$ avente sulle righe le parole chiave e sulle colonne i documenti. Tale matrice A è detta *matrice di termine-documento*. Siccome difficilmente ogni termine è contenuto in ciascun documento, la matrice A è solitamente sparsa.

[1] Con il termine "frequenza" si intende il numero di ricorrenze di una parola.

Come esempio di una tale costruzione, si riportano 1033 documenti provenienti dal database bibliografico biomedico MEDLINE (*Medical Literature Analysis and Retrieval System Online*) e organizzati nella matrice di termine-documento `A_med`$\in \mathbb{R}^{5735\times1033}$. La matrice `dict_med`$\in \mathbb{R}^{5735\times23}$ raccoglie i termini chiave utilizzati. Ciascun coefficiente $a_{i,j}$ di `A_med` è la frequenza del termine i nel documento j. Dal plot di una porzione di `A_med` in Figura 1.1 è visualizzabile la struttura della matrice.

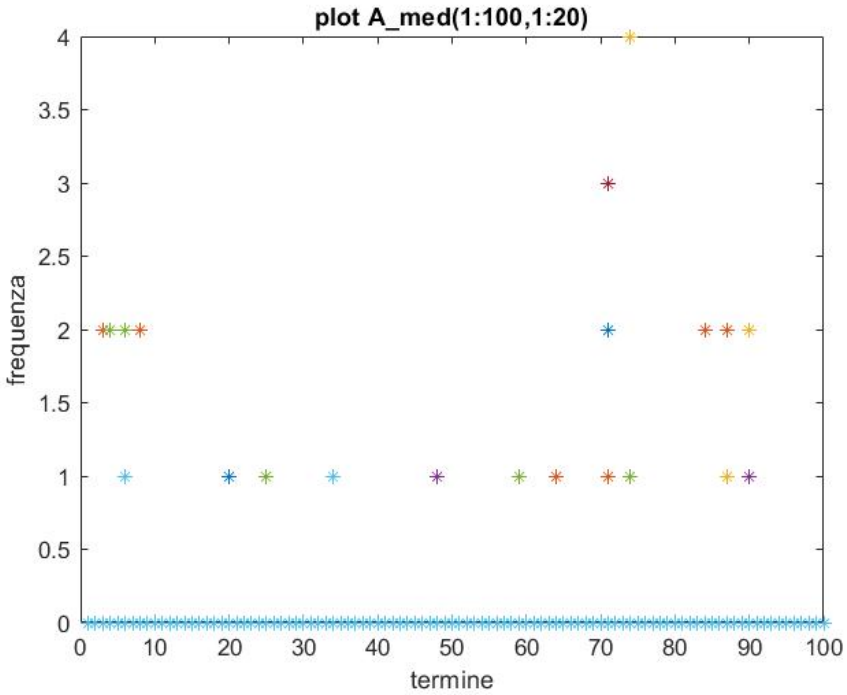

Figura 1.1: *Frequenza dei termini nei documenti di A_med*

Come già osservato, la matrice di termine-documento è sparsa. Questo fatto, riscontrabile già in Figura 1.1, è ben visbile in Figura 1.2, dove sono evidenziati gli elementi non zero di parte di `A_med`. Il numero di elementi zero di `A_med` è circa il $99,14\%$.

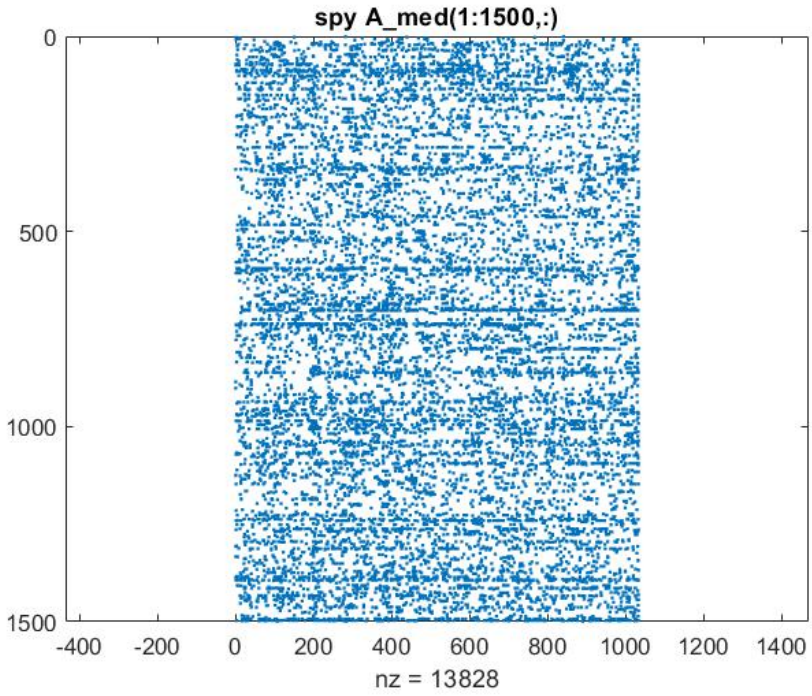

Figura 1.2: *Sparsità della matrice A_med*

Se si vogliono confrontare una query q con un documento d (o se si vogliono confrontare due documenti) la misura più adottata è la misura di similarità del coseno, definita come segue:

$$\text{sim}(q, d) = \cos\theta = \frac{q^T d}{\|q\|_2 \|d\|_2}$$

dove θ è l'angolo fra q e d.
La misura di similarità del coseno appare migliore in questo contesto rispetto a misure che quantificano la grandezza della differenza fra i vettori (ad esempio, la distanza Euclidea). Infatti, con queste ultime potrebbe accadere che due documenti di contenuto simile risultino distanti semplicemente a causa di una significativa differenza di lunghezza dei due documenti. Inoltre, la misura di similarità del coseno permette di sfruttare la sparsità dei vettori documento.

Il modello di spazio vettoriale appena descritto è soggetto ad alcune limitazioni provenienti dalla complessità del linguaggio naturale. In particolare, la sinonimia (il caso in cui parole diverse abbiano lo stesso significato) e la polisemia (il caso in cui una stessa parola abbia più significati) costituiscono un ostacolo qualora si vogliano confrontare dei documenti, questo poiché il calcolo della similarità fra due documenti contenenti sinonimi, farà sì che essi risultino meno simili rispetto a quello che giudicherebbe l'utente; analogamente, due documenti con termini polisemici, risulteranno più simili per la mi-

sura di similarità di quanto non siano in realtà. Esistono però tecniche che sfruttano la struttura semantica della collezione di documenti, ovvero che mirano a cogliere associazioni di contenuto e significato fra i documenti. Fra queste tecniche, la più comunemente nota è la *Latent Semantic Indexing* (LSI). La LSI è una variante del modello di spazio vettoriale che si basa una fattorizzazione particolare della matrice di termine-documento, ossia la *Decomposizione ai Valori Sigolari* (SVD), o Singular Value Decomposition.

1.2 Decomposizione ai Valori Singolari (SVD)

La Decomposizione ai Valori Singolari consiste nello scomporre una matrice A nel prodotto di tre matrici $A = U\Sigma V^T$, con U e V ortogonali. Più precisamente, vale il seguente teorema di esistenza di tale decomposizione.

Teorema 1.1. *Una qualsiasi matrice $A \in \mathbb{R}^{m\times n}$ può essere fattorizzata come:*

$$A = U \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T$$

con $U \in \mathbb{R}^{m\times m}$ e $V \in \mathbb{R}^{n\times n}$ ortogonali e, posto $q = \min\{m, n\}$, $\Sigma \in \mathbb{R}^{q\times q}$ diagonale,

$$\Sigma = \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_q \end{pmatrix}$$

con $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_q \geq 0$.

Le colonne $u_1, \ldots, u_m$ di U e le colonne $v_1, \ldots, v_n$ di V sono dette, rispettivamente, *vettori singolari sinistri* e *vettori singolari destri*, e $\sigma_1, \ldots, \sigma_q$ sono detti *valori singolari*. Si ha che il numero di $\sigma_i \neq 0$ coincide con il rango di A. La decomposizione ai valori singolari permette di scrivere:

$$A = \sum_{i=1}^{q} u_i \sigma_i v_i^T$$

dove $u_i\sigma_i v_i^T$ sono dette *componenti singolari*.

Dimostrazione. Senza perdere di generalità si suppone $m \geq n$, poiché nel caso in cui $m < n$ si applica il teorema ad A^T. Sia $\hat{x}$ la soluzione del problema di massimo

$$\max_{\substack{x\in\mathbb{R}^n \\ \|x\|_2=1}} \|Ax\|_2.$$

Posto $\sigma_1 = \|A\|_2$, sia $y = \frac{A\hat{x}}{\sigma_1}$, $\|y\|_2 = 1$.
Si costruiscano le matrici $X_1 = [\hat{x}, \tilde{X_2}] \in \mathbb{R}^{n\times n}$ e $Y_1 = [y, \tilde{Y_2}] \in \mathbb{R}^{m\times m}$ in modo che siano ortogonali. Sia

$$A_1 = Y_1^T A X_1 = \begin{pmatrix} y^T Ax & y^T A\tilde{X_2} \\ \tilde{Y_2}^T Ax & \tilde{Y_2}^T A\tilde{X_2} \end{pmatrix}$$

dove $y^T Ax = \sigma_1$ e $\tilde{Y_2}^T Ax = 0$ per costruzione. Dunque, si pone

$$A_1 = \begin{pmatrix} \sigma_1 & d^T \\ 0 & B \end{pmatrix} \text{ con } d \in \mathbb{R}^{n-1} \text{ e } B \in \mathbb{R}^{(m-1)\times(n-1)}.$$

Si osservi che $\|A_1\|_2 = \|A\|_2$ dal momento che A_1 si ottiene da A mediante trasformazioni ortogonali. Sia $z = \begin{pmatrix} \sigma_1 \\ d \end{pmatrix}$. Allora

$$\sigma_1^2 = \|A_1\|_2^2 \geq \frac{\|A_1 z\|_2^2}{\|z\|_2^2} = \frac{1}{\sigma_1^2 + d^T d} \left\| \begin{pmatrix} \sigma_1^2 + d^T d \\ Bd \end{pmatrix} \right\|_2^2 \geq$$
$$\geq \frac{(\sigma_1^2 + d^T d)^2 + \|Bd\|_2^2}{\sigma_1^2 + d^T d} \geq \sigma_1^2 + d^T d$$

Dunque, necessariamente $d = 0$. A questo punto, siccome $A_1 = \begin{pmatrix} \sigma_1 & 0^T \\ 0 & B \end{pmatrix}$ si prova la tesi procedendo per induzione su B, ottenendo $U = Y_1, \ldots, Y_{n-1}$ e $V = X_1, \ldots, X_{m-1}$. □

Una proprietà particolarmente importante della SVD è la possibilità di ottenere una approssimazione di rango basso di una matrice, $A \in \mathbb{R}^{m\times n}$ con $m \geq n$. In particolare, il risultato che segue mostra come la SVD troncata di A, ossia il troncamento di A alle prime k componenti singolari, sia la migliore approssimazione di A di rango k e che la distanza di A da tale troncamento è σ_{k+1}. Prima di enunciare il teorema, si osserva che

$$A = \sum_{i=1}^{n} u_i \sigma_i v_i^T = \sum_{i=1}^{k} u_i \sigma_i v_i^T + \sum_{i=k+1}^{n} u_i \sigma_i v_i^T$$

e si denota

$$A_k := \sum_{i=1}^{k} u_i \sigma_i v_i^T = U_k \Sigma_k V_k^T.$$

Teorema 1.2. *Sia $A \in \mathbb{R}^{m\times n}$ una matrice tale che esiste $k \in \{1, \ldots, n-1\}$ per cui $\sigma_{k+1} << \sigma_k$. Allora $A_k = U_k \Sigma_k V_k^T$ è la soluzione del problema di minimo*

$$\min_{\substack{B\in\mathbb{R}^{m\times n} \\ rango(B)=k}} \|A - B\|_2, \tag{1.1}$$

e

$$\|A - A_k\|_2 = \sigma_{k+1}. \tag{1.2}$$

Dimostrazione. Si procede innanzitutto dimostrando la relazione (1.2).
Posto $\Sigma_1 = \text{diag}(\sigma_1, \ldots, \sigma_k)$,

$$A = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & \\ & \Sigma_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} = U_1\Sigma_1 V_1^T + U_2\Sigma_2 V_2^T = A_k + U_2\Sigma_2 V_2^T.$$

Allora

$$\|A - A_k\|_2 = \|U_2\Sigma_2 V_2^T\|_2 = \|\Sigma_2 V_2^T\|_2 = \|V_2\Sigma_2^T\|_2 = \|\Sigma_2\|_2 = \sigma_{k+1}.$$

Siano $B \in \mathbb{R}^{m\times n}$ matrice di rango k e $\{x_1, \ldots, x_{n-k}\}$ una base per $Ker(B)$. Sia poi $Z = span(\{x_1, \ldots, x_{n-k}\} \cap \{v_1, \ldots, v_{k+1}\})$. Preso $z \in Z$, $\|z\|_2 = 1$ e osservato che $Bz = 0$ e $Az = \sum_{i=1}^{k+1} u_i\sigma_i v_i^T z$, si ottiene

$$\|A - B\|_2^2 \geq \|(A - B)z\|_2^2 = \|Az\|_2^2 = \sum_{i=1}^{k+1} \sigma_i^2 (v_i^T z)^2 \geq \sigma_{k+1}^2$$

Da (1.2) segue che il minimo è raggiunto per $B = A_k$.

□

Si vogliono ora riportare alcuni risultati riguardanti la SVD che verranno utilizzati nel Capitolo 3: è possibile impiegare la SVD per risolvere un problema ai minimi quadrati, questo anche nel caso in cui la matrice non sia a rango pieno. Sia $A \in \mathbb{R}^{m\times n}$ a rango pieno e $q = \min\{m, n\}$. Sia $b \in \mathbb{R}^m$ e si consideri il problema ai minimi quadrati

$$\min_{x\in\mathbb{R}^n} \|b - Ax\|_2. \tag{1.3}$$

Effettuando la SVD della matrice,

$$A = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 \\ 0 \end{pmatrix} V^T$$

dove $U_1 \in \mathbb{R}^{m\times q}$ e $\Sigma_1 \in \mathbb{R}^{q\times q}$, si ha

$$\begin{aligned} \|b - Ax\|_2^2 &= \|UU^T b - U\Sigma V^T x\|_2^2 = \|U(U^T b - \Sigma V^T x)\|_2^2 = \\ &= \left\| \begin{pmatrix} U_1^T b - \Sigma_1 V^T x \\ U_2^T b \end{pmatrix} \right\|_2^2 = \|U_1^T b - \Sigma_1 V^T x\|_2^2 + \|U_2^T b\|_2^2. \end{aligned}$$

Pertanto, dovendo minimizzare la quantità $\|U_1^T b - \Sigma_1 V^T x\|_2^2$, occorre risolvere il problema $U_1^T b - \Sigma_1 V^T x = 0$ che ha soluzione

$$x = V\Sigma_1^{-1} U_1^T b = \sum_{i=1}^{q} \left(\frac{u_i^T b}{\sigma_i}\right) v_i.$$

Se la matrice A non è a rango pieno, il problema ai minimi quadrati (1.3) ha infinite soluzioni, ma si può provare che la soluzione ottenuta con la SVD è quella di norma minima. Sia $A \in \mathbb{R}^{m\times n}$ con $\text{rank}(A) = r < \min\{m,n\}$, allora

$$A = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

con $U_1 \in \mathbb{R}^{m\times r}, V_1 \in \mathbb{R}^{n\times r}$ e $\Sigma_1 \in \mathbb{R}^{r\times r}$ la parte non singolare di Σ.

Teorema 1.3. *Sia $A \in \mathbb{R}^{m\times n}$ con $rank(A) = r < \min\{m,n\}$. Allora la soluzione di norma minima del problema ai minimi quadrati* (1.3) *è ottenuta tramite la SVD di A, ossia*

$$x = \sum_{i=1}^{r} (\frac{u_i^T b}{\sigma_i}) v_i.$$

Dimostrazione. La norma del residuo è

$$\|b - Ax\|_2^2 = \left\| b - \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} x \right\|_2^2.$$

Posto $z = V^T x = \begin{pmatrix} V_1^T x \\ V_2^T x \end{pmatrix} = \begin{pmatrix} y \\ d \end{pmatrix}$ si ha allora

$$\|b - Ax\|_2^2 = \left\| b - \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y \\ d \end{pmatrix} \right\|_2^2 = \left\| \begin{pmatrix} U_1^T b \\ U_2^T b \end{pmatrix} - \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y \\ d \end{pmatrix} \right\|_2^2 =$$
$$= \left\| \begin{pmatrix} U_1^T b - \Sigma_1 y \\ U_2^T \end{pmatrix} \right\|_2^2 = \|U_1^T b - \Sigma_1 y\|_2^2 + \|U_2^T b\|_2^2.$$

Si può minimizzare il residuo prendendo $y = \Sigma_1^{-1} U_1^T b$, da cui

$$x = V_1 \Sigma_1^{-1} U_1^T b + V_2 d.$$

Allora

$$\|x\|_2^2 = \left\| \begin{pmatrix} y \\ d \end{pmatrix} \right\|_2^2 = \|y\|_2^2 + \|d\|_2^2 \geq \|y\|_2^2.$$

Segue che

$$\min_d \|x\|_2 = \|y\|_2$$

per $d = 0$. □

1.3 Latent Semantic Indexing (LSI)

La Latent Semantic Indexing è una tecnica per l'information retrieval che si basa sull'ipotesi che esistano strutture semantiche latenti nei dati della matrice di termine-documento, le quali possono essere individuate tramite una approssimazione di rango basso ottenuta con la SVD della matrice.

Sia $A \in \mathbb{R}^{m \times n}$ matrice di termine-documento con $m \geq n$ e si consideri la sua decomposizione ai valori singolari $A = U\Sigma V^T$. Indicato con e_j il j-esimo vettore della base canonica, la j-esima colonna di A è

$$Ae_j = \sum_{i=1}^{n} u_i(\sigma_i v_i^T e_j),$$

dunque, ogni colonna di A (ogni documento della collezione) è combinazione lineare dei vettori singolari sinistri, ovvero le colonne di U sono una base ortonormale dello spazio dei documenti. Allora si possono pensare le colonne di U come una rappresentazione degli "argomenti": ogni vettore u_i codifica un argomento, sempre meno dominante a partire da u_1. Sia ora $k < n$ e si consideri la SVD troncata di A alle prime k componenti singolari, $A_k = U_k\Sigma_k V_k^T$: le colonne di $U_k \in \mathbb{R}^{m \times k}$ e di $\Sigma_k V_k^T \in \mathbb{R}^{k \times n}$ sono, rispettivamente, una rappresentazione degli argomenti e dei documenti nello spazio k-dimensionale. L'idea alla base della LSI è che siccome il numero k di dimensioni è minore del numero m di parole chiave iniziale, proiettando A nello spazio di dimensione ridotta tramite la SVD troncata, vengono ignorate le informazioni poco significative o di disturbo dai dati e catturate le strutture semantiche date dalle associazioni di termini e documenti. Ad esempio, quando vengono proiettati i documenti nello spazio k-dimensionale (matrice $\Sigma_k V_k^T \in \mathbb{R}^{k \times n}$), i termini con co-occorrenze simili vengono uniti (dove con co-occorrenza ci si riferisce alla frequenza con cui due parole appaiono insieme in uno stesso documento). Ci si aspetta allora che termini sinonimi vengano utilizzati nello stesso contesto (e pertanto avranno co-occorrenze simili) e che quindi la loro relazione venga colta nello spazio ridotto. Per quanto riguarda la polisemia, il troncamento potrebbe non tenere conto di usi poco frequenti di una determinata parola. Dato un vettore query $q \in \mathbb{R}^m$, è possibile ottenere la sua rappresentazione k-dimesionale $q_k = U_k^T q \in \mathbb{R}^k$, confrontarlo con gli altri documenti nello spazio ridotto e ordinare i documenti sulla base della similarità con la query.

Nonostante la Latent Semantic Indexing sia un metodo largamente diffuso e ben performante nell'ambito dell'information retrieval, esso presenta alcune difficoltà: l'elevato

costo computazionale della SVD e la scelta della dimensione k per lo spazio ridotto. Valori troppi piccoli di k causano una eccessiva perdita di informazione, mentre valori troppo grandi di k non sfruttano le proprietà su cui si basa la LSI, tornando ad avere le problematiche del modello vettoriale standard. Un criterio piuttosto immediato con il quale effettuare la scelta della dimensione k è quello di osservare l'andamento dei valori singolari della matrice di termine-documento e considerare il valore k in corrispondenza del quale il plot dei valori singolari presenta un "gomito", dopo il quale essi decrescono velocemente. Nell'esempio della matrice `A_med` il plot dei valori singolari evidenzia come sia opportuno un valore di k fra 100 e 200.

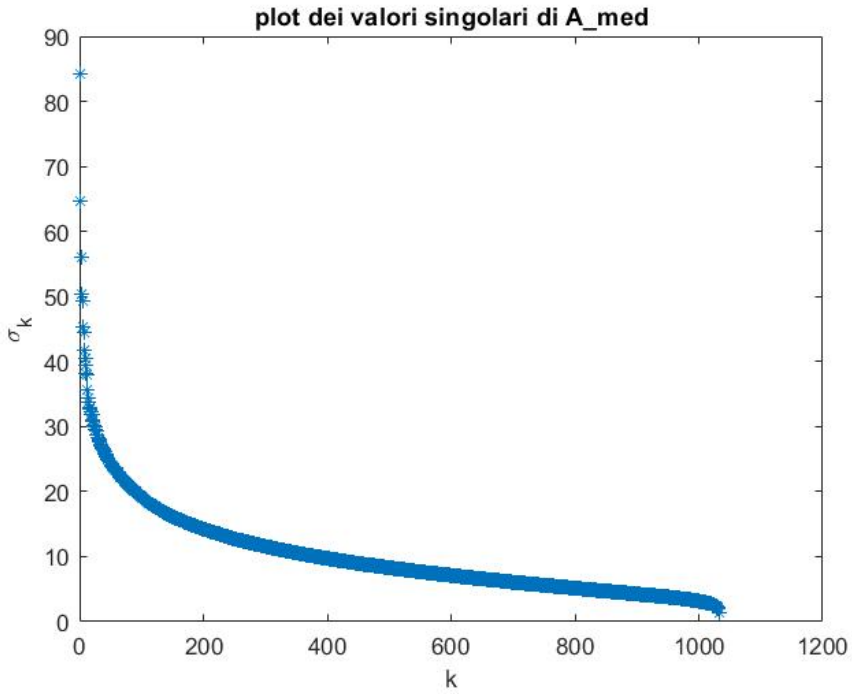

Figura 1.3: *Andamento dei valori singolari di A_med*

Capitolo 2

Document clustering

In generale, si definisce il clustering come l'insieme delle tecniche per determinare raggruppamenti omogenei (*cluster*) di un insieme di dati, in modo che oggetti appartenenti allo stesso cluster siano il più simili possibile fra loro e dissimili da quelli in altri cluster. In particolare, si può parlare di *document clustering* qualora oggetto del clustering sia una collezione di documenti che si vuole organizzare raggruppando i suoi elementi sulla base dell'argomento in essi trattato. Gli algoritmi di clustering più comuni adottano il modello vettoriale per la rappresentazione dei documenti e ne valutano la somiglianza con una misura di similarità (o di distanza), operando in modo da massimizzare la similarità fra documenti appartenenti allo stesso cluster e minimizzare la similarità fra documenti in cluster diversi.

2.1 Tecniche di clustering

È possibile distinguere gli algoritmi di clustering in algoritmi di *clustering gerarchico* e algoritmi di *clustering partizionale* (o clustering non gerarchico) sulla base della struttura del risultato finale.

2.1.1 Clustering gerarchico

Le tecniche di clustering gerarchico determinano sequenze nidificate di partizioni dell'insieme dei dati, le quali possono essere visualizzabili in una struttura ad albero (dendrogramma). Queste tecniche possono essere a loro volta suddivise in metodi gerarchici *agglomerativi* o *divisivi*, a seconda del criterio con cui vengono formati i cluster: un metodo agglomerativo fonde progressivamente i cluster più vicini a partire da una iniziale suddivisione dei dati in cluster contenenti ciascuno un oggetto e fino ad ottenere un unico cluster o fino a che non si verifica una condizione di terminazione. Un metodo divisivo,

invece, procede con la progressiva suddivisione in cluster più piccoli a partire da un unico cluster contenente tutti i dati fino a che non si ottengono cluster contenenti ciascuno un dato o non è soddisatto un criterio d'arresto.

Considerato un insieme $\{x_1, \ldots, x_n\}$ di n oggetti, la tipica procedura di un metodo gerarchico agglomerativo è la seguente:

1. Costruisce la matrice di distanza (o di similarità), ossia una matrice $D \in \mathbb{R}^{n \times n}$ simmetrica tale che $D_{i,j} = d(x_i, x_j)$ dove d è una misura di distanza (o di similarità);
2. Valuta gli elementi di D e determina i due cluster più simili π e π' ;
3. Unisce π e π';
4. Aggiorna D sostituendo alle righe corrispondenti a π e π' una riga con le distanze del nuovo cluster $\pi \cup \pi'$ da tutti gli altri;
5. Ripete i passi 2, 3 e 4 fino a che non è formato un unico cluster o fino a che non si verifica un criterio d'arresto.

Occorre specificare come viene definita la distanza fra i cluster e questo distingue i metodi di connessione (*linkage*), nei quali la distanza fra i cluster è una funzione della distanza fra le coppie di elementi degli insiemi. Dati due cluster π e π' si definiscono:

a) *Single linkage* in cui la distanza fra π e π' è data dalla minima distanza:

$$\min_{x \in \pi, y, \in \pi'} d(x, y);$$

b) *Complete linkage* in cui la distanza fra π e π' è data dalla massima distanza:

$$\max_{x \in \pi, y \in \pi'} d(x, y);$$

c) *Average linkage* in cui la distanza fra π e π' è data dalla distanza media:

$$\frac{1}{|\pi||\pi'|} \sum_{x \in \pi} \sum_{y \in \pi'} d(x, y).$$

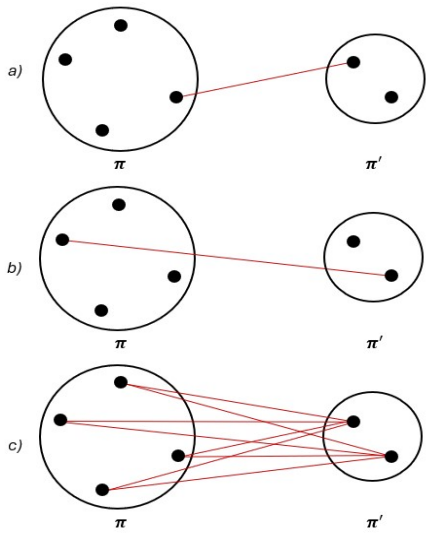

Figura 2.1: *Esempio delle distanze fra cluster per gli algoritmi di clustering gerarchico agglomerativo.*

2.1.2 Clustering partizionale

Le tecniche di clustering partizionale determinano una singola partizione dell'insieme dei dati. Il problema di clustering partizionale può essere formulato come segue: data una collezione di n oggetti $\mathcal{D} = \{x_1, \ldots, x_n\}$ e $k \leq n$ fissato, si vuole ottenere una partizione di cardinalità k di $\mathcal{D}$, ovvero una famiglia di k cluster $\pi_1, \ldots, \pi_k$ tale che

(i) $\pi_i \neq \emptyset, \forall i = 1, \ldots, k,$

(ii) $\pi_i \cup \pi_j = \emptyset, \forall i, j = 1, \ldots, k, i \neq j,$

(iii) $\bigcup_{i=1}^{k} \pi_i = \mathcal{D},$

che minimizzi (o massimizzi) una certa funzione obiettivo definita in termini di una misura di distanza (o di similarità) e di rappresentanti dei cluster. La scelta della misura e dei rappresentanti è caratterizzante dell'algoritmo.

Algoritmo delle k-medie

L'algoritmo di clustering partizionale più comune è l'algoritmo delle k-medie, la cui finalità è quella di determinare una partizione dell'insieme dei dati che minimizzi la media dei quadrati delle distanze Euclidee degli oggetti dal centroide del cluster cui appartengono, dove si definisce il *centroide* del cluster π come la media degli oggetti contenuti in π:

$$c = \frac{1}{|\pi|} \sum_{x \in \pi} x$$

e che viene utilizzato come rappresentante del cluster. La funzione obiettivo che si vuole minimizzare con l'algoritmo delle k-medie è la somma residua dei quadrati (Residual Sum of Squares)

$$RSS = \sum_{i=1}^{k} \sum_{x \in \pi_i} \|x - c_i\|_2^2.$$

La determinazione di una partizione che minimizzi RSS è un problema NP-hard e, pertanto, si costruisce una procedura iterativa che ad ogni passo raffina il clustering a partire da una partizione iniziale. Il primo passo è la selezione di k oggetti (*seed points*) che saranno i centroidi iniziali dei cluster. Dopodiché, ad ogni iterazione viene calcolata la distanza di ciascun oggetto dai centroidi, gli oggetti vengono riassegnati al cluster avente centroide più vicino e vengono aggiornati i centroidi con i nuovi elementi del cluster. L'algoritmo termina quando è soddisfatto un criterio di arresto. Essendo ogni partizione un raffinamento di quella precedente, al termine della procedura è stato esaminato solo

un numero ristretto di tutte le possibili partizioni dell'insieme dei dati ed è per questo motivo che algoritmi basati su questo principio hanno lo svantaggio di convergere a minimi (o massimi) locali della funzione obiettivo e che questa convergenza dipende fortemente dall'inizializzazione.

2.2 Scelta del metodo per il document clustering

Dal momento che esistono numerosi algoritmi di clustering, ciascuno dei quali produrrà partizioni diverse e avrà costi diversi, la scelta della tecnica da utilizzare deve essere effettuata sulla base del risultato che si vuole ottenere e delle caratteristiche dei dati. Nel caso del document clustering, una tale scelta deve tener conto della vastità della collezione di documenti e delle principali proprietà che caratterizzano la loro rappresentazione vettoriale, ovvero l'elevato numero di dimensioni, la sparsità e un numero di entrate non nulle che può differire notevolmente da un documento all'altro. Alcuni studi ([9], [12]) hanno mostrato come gli algoritmi di clustering partizionale siano migliori nel trattare i documenti rispetto a quelli di clustering gerarchico, sia dal punto di vista della complessità computazionale che della qualità del clustering. Innanzitutto, gli algoritmi di clustering gerarchico utilizzano una matrice di distanza e ad ogni iterazione è previsto il calcolo delle distanze fra tutti i cluster, il che spiega il costo computazionale almeno quadratico nel numero di documenti. Inoltre, uno svantaggio che presentano questi algoritmi è l'impossibilità di migliorare la qualità del clustering correggendo gli assegnamenti una volta che i cluster sono stati accorpati (o divisi). Questa poca flessibilità è particolarmente evidente nel document clustering: per come sono costruiti i vettori documento nel modello vettoriale, ciò che discrimina l'appartenenza di un documento ad un cluster o ad un altro è la frequenza con cui le parole chiave ricorrono nel documento, da cui segue che ogni cluster possiede un vocabolario di termini più frequenti e quindi caratterizzanti. Tuttavia, può accadere che documenti di contenuto diverso condividano parte di questi vocabolari e, pertanto, vengano riconosciuti come simili e posti erroneamente nello stesso cluster. Dunque, se i documenti non hanno contenuti ben distinti, questi errori emergono già a partire dalla prima fase di fusione dei cluster e vengono portati avanti nella procedura influenzando negativamente gli accorpamenti successivi. Per questi motivi, nel document clustering sono preferibili algoritmi di clustering partizionale e fra questi l'algoritmo delle k-medie presenta alcuni vantaggi che lo hanno reso uno degli algoritmi di clustering più diffusi, ovvero la semplicità di implementazione e la facile interpretazione del risultato, il costo computazionale lineare nel numero di documenti e

la qualità del clustering. In particolare, sono proprio la visione globale data dal valutare la distanza di tutti i documenti dalle medie dei cluster e il progressivo aggiornamento e raffinamento che rendono l'algoritmo delle k-medie migliore rispetto ad altri per la qualità del clustering.

Come già osservato nel capitolo precedente, la misura di similarità del coseno risulta essere più accurata della norma Euclidea per il confronto dei vettori documento, poiché la prima ne sfrutta la sparsità e si basa sull'orientamento piuttosto che sulla magnitudine. Nella sezione successiva viene allora presentata una versione dell'algoritmo delle k-medie che utilizza questa misura. Si noti che se i vettori hanno norma unitaria, la similarità del coseno equivale al calcolo del prodotto scalare fra essi: dati due vettori x e y tali che $\|x\|_2 = \|y\|_2 = 1$, si ha

$$\text{sim}(x, y) = \cos\theta = \frac{x^T y}{\|x\|_2 \|y\|_2} = x^T y.$$

In particolare, l'algoritmo descritto in seguito considera i vettori documento e i centroidi normalizzati e quindi, dal momento che questi giacciono su una ipersfera (unitaria), tale variante delle k-medie prende il nome di *algoritmo delle k-medie sferiche*.

2.3 Algoritmo delle k-medie sferiche

Data una collezione di n vettori documento $\{d_1, \ldots, d_n\}$ di dimensione m, si supponga che essi abbiano norma unitaria. Sia $\pi_1, \ldots, \pi_k$ una partizione di cardinalità k dell'insieme dei documenti, con $k \leq n$ fissato, e

$$c_j = \frac{\sum_{d \in \pi_j} d}{\|\sum_{d \in \pi_j} d\|_2}$$

il centroide di norma unitaria del cluster π_j.

Proposizione 2.1. *Per ogni $z \in \mathbb{R}^m$ tale che $\|z\|_2 = 1$ e $j \in \{1, \ldots, k\}$ si ha*

$$\sum_{d \in \pi_j} d^T z \leq \sum_{d \in \pi_j} d^T c_j$$

Dimostrazione. Utilizzando la disuguaglianza di Cauchy-Schwarz e l'ipotesi $\|z\|_2 = 1$ si ottiene

$$\sum_{d \in \pi_j} d^T z = \left(\sum_{d \in \pi_j} d \right)^T z \leq \left\| \sum_{d \in \pi_j} d \right\|_2 .$$

D'altra parte, per definizione di c_j, si ha che

$$\sum_{d\in\pi_j} d^T c_j = \left(\sum_{d\in\pi_j} d\right)^T \frac{\sum_{d\in\pi_j} d}{\left\|\sum_{d\in\pi_j} d\right\|_2} = \frac{\left\|\sum_{d\in\pi_j} d\right\|_2^2}{\left\|\sum_{d\in\pi_j} d\right\|_2} = \left\|\sum_{d\in\pi_j} d\right\|_2.$$

□

Dunque, fra tutti i possibili vettori, il centroide di norma unitaria c_j è il vettore "globalmente" più simile ai vettori documento contenuti nel j-esimo cluster. Pertanto, lo si può pensare come il vettore che rappresenta l'argomento comune ai vettori documento contenuti nel cluster π_j e per tale motivo viene chiamato *concept vector*.

Dalla Proposizione 2.1 si può definire una misura della coerenza del cluster π_j, $j \in \{1,\dots,k\}$, basata sulla similarità del coseno:

$$\sum_{d\in\pi_j} d^T c_j.$$

Dopodiché, si può definire una funzione obiettivo che sia una misura della coerenza dell'intero partizionamento $\{\pi_j\}_{j=1}^k$ della collezione di documenti:

$$\mathcal{Q}\Big(\{\pi_j\}_{j=1}^k\Big) = \sum_{j=1}^{k}\sum_{d\in\pi_j} d^T c_j.$$

Il problema consiste allora nel determinare una partizione di cardinalità k della collezione di documenti $\{d_1,\dots,d_n\}$ che massimizzi la funzione obiettivo $\mathcal{Q}$, ossia si vuole determinare una partizione $\pi_1^*,\dots,\pi_k^*$ tale che

$$\{\pi_j^*\}_{j=1}^k = \arg\max_{\{\pi_j\}_{j=1}^k} \mathcal{Q}\Big(\{\pi_j\}_{j=1}^k\Big). \tag{2.1}$$

Siccome la determinazione della soluzione al problema di massimo (2.1) è NP-hard, in Algoritmo 1 è descritta una procedura che iterativamente determina un massimo (locale).

Algoritmo 1: k-medie sferiche

Inizializzazione: Sia $\{\pi_j^{(0)}\}_{j=1}^k$ una partizione arbitraria dell'insieme dei vettori documento di norma unitaria e $\{c_j^{(0)}\}_{j=1}^k$ i concept vector associati.

for *t=0,1,2,...* **do**

  for *i=1,...,n* **do**

    for *l=1,...,k* **do**

      $\alpha(l) = d_i^T c_l^{(t)}$;

    end

    Determina $j = \arg\max_l \alpha(l)$;

    Considera la nuova partizione

$$\pi_j^{(t+1)} = \{d \in \{d_i\}_{i=1}^n : j = \arg\max_l d^T c_l^{(t)}\}; \tag{2.2}$$

  end

  for *j=1,...,k* **do**

    Aggiorna i concept vector:

$$\hat{c_j}^{(t+1)} = \sum_{d \in \pi_j^{(t+1)}} d;$$

$$c_j^{(t+1)} = \frac{\hat{c_j}^{(t+1)}}{\|\hat{c_j}^{(t+1)}\|_2};$$

  end

  if *è soddisfatto il criterio d'arresto* **then**

    Termina

  end

end

Come condizione di terminazione per Algoritmo 1, si può scegliere una delle seguenti:

- È raggiunto un numero massimo di iterazioni;
- L'assegnazione di documenti ai cluster non cambia da una iterazione all'altra;
- I centroidi non cambiano da una iterazione all'altra;
- La variazione della funzione obiettivo è al di sotto di una certa soglia $\epsilon \geq 0$:

$$\left| \mathcal{Q}\left(\{\pi_j^{(t)}\}_{j=1}^k \right) - \mathcal{Q}\left(\{\pi_j^{(t+1)}\}_{j=1}^k \right) \right| \leq \epsilon.$$

È necessario combinare le ultime tre condizioni con un limite massimo di iterazioni per garantire la terminazione ed evitare lunghi tempi di esecuzione.

2.3.1 Complessità computazionale

Siano λ_i il numero di elementi non zero dell'i-esimo vettore documento e $\lambda := \lambda_1 + \cdots + \lambda_n$ il numero totale di elementi non zero della matrice di termine-documento. Sia τ il numero di iterazioni. Ad ogni iterazione:

- vengono calcolati kn prodotti scalari $d_i^T c_l^{(t)}$ con $i = 1, \ldots, n$, e $l = 1, \ldots, k$, ciascuno dei quali ha costo $2\lambda_i - 1$. Segue che complessivamente il costo dei prodotti scalari è $O(\lambda k)$;
- per ogni documento si determina il centroide più vicino con la ricerca di $j = \arg\max_l \alpha(l)$ e questo ha costo $O(kn)$;
- vengono aggiornati i concept vector $c_j^{(t+1)}$ per $j = 1, \ldots, k$ ricalcolando e normalizzando il centroide di ciascun cluster, per un costo complessivo di $O(\lambda + km)$.

Allora la complessità computazionale totale dopo τ iterazioni è $O((\lambda k + kn + km)\tau)$, ma solitamente $\lambda \gg \max\{m, n\}$ e quindi la complessità computazionale dell'algoritmo è $O(\lambda k \tau)$. In Figura 2.2 è visualizzabile la dipendenza lineare del tempo di calcolo da k.

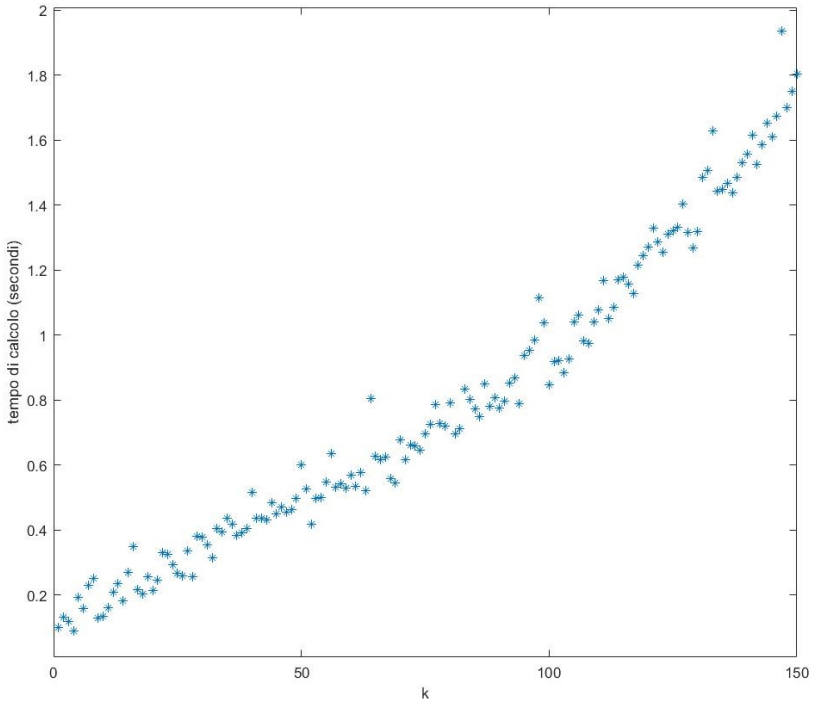

Figura 2.2: *Tempo impiegato per il clustering di A_med al crescere di k.*

2.3.2 Risultati di convergenza

Si dimostra ora che l'algoritmo delle k-medie sferiche converge dimostrando innanzitutto che esso è tale per cui il valore della funzione obiettivo $\mathcal{Q}$ non decresce da una iterazione all'altra.

Lemma 2.2. *Per ogni $t \geq 0$, si ha che*

$$\mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big) \leq \mathcal{Q}\Big(\{\pi_j^{(t+1)}\}_{j=1}^k\Big).$$

Dimostrazione.

$$\begin{aligned}
\mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big) &= \sum_{j=1}^k \left(\sum_{d \in \pi_j^{(t)}} d^T c_j^{(t)} \right) = \sum_{j=1}^k \left(\sum_{l=1}^k \sum_{d \in \pi_j^{(t)} \cap \pi_l^{(t+1)}} d^T c_j^{(t)} \right) \leq \\
&\leq \sum_{j=1}^k \left(\sum_{l=1}^k \sum_{d \in \pi_j^{(t)} \cap \pi_l^{(t+1)}} d^T c_l^{(t)} \right) = \sum_{l=1}^k \left(\sum_{j=1}^k \sum_{d \in \pi_j^{(t)} \cap \pi_l^{(t+1)}} d^T c_l^{(t)} \right) = \\
&= \sum_{l=1}^k \sum_{d \in \pi_l^{(t+1)}} d^T c_l^{(t)} \leq \sum_{l=1}^k \sum_{d \in \pi_l^{(t+1)}} d^T c_l^{(t+1)} = \mathcal{Q}\Big(\{\pi_j^{(t+1)}\}_{j=1}^k\Big)
\end{aligned}$$

dove la prima disuguaglianza segue dalla costruzione (2.2) dei cluster nell'algoritmo e la seconda disuguaglianza segue dalla Proposizione 2.1. □

Corollario 2.3. *Il limite seguente esiste:*

$$\lim_{t\to+\infty} \mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big).$$

Dimostrazione. Dal Lemma 2.2 si ha che $\Big\{\mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big)\Big\}_{t\geq 0}$ è una successione monotona crescente. Si osservi poi che, posto $\hat{c_j}^{(t)}$ il centroide non normalizzato del cluster $\pi_j^{(t)}$ e $n_j^{(t)} = |\pi_j^{(t)}|$, siccome $\sum_{d\in\pi_j^{(t)}} d = n_j^{(t)}\hat{c_j}^{(t)}$ e $\|c_j^{(t)}\|_2 = 1$ si ha che

$$\sum_{d\in\pi_j^{(t)}} d^T c_j^{(t)} = n_j^{(t)}\hat{c_j}^{(t)^T} c_j^{(t)} = n_j^{(t)}\|\hat{c_j}^{(t)}\|_2 c_j^{(t)^T} c_j^{(t)} = n_j^{(t)}\|\hat{c_j}^{(t)}\|_2.$$

Allora, per quanto appena provato e per la disuguaglianza triangolare (ricordando che i vettori documento hanno norma unitaria)

$$\mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big) = \sum_{j=1}^k \sum_{d\in\pi_j^{(t)}} d^T c_j^{(t)} = \sum_{j=1}^k n_j^{(t)}\|\hat{c_j}^{(t)}\|_2 \leq \sum_{j=1}^k n_j^{(t)} = n.$$

Dunque, siccome $\Big\{\mathcal{Q}\Big(\{\pi_j^{(t)}\}_{j=1}^k\Big)\Big\}_{t\geq 0}$ è una successione monotona crescente e superiormente limitata da una costante, si può concludere che la successione converge e il limite esiste. □

Dunque, per quanto appena dimostrato, si ha che iterando l'algoritmo delle k-medie sferiche la funzione obiettivo converge. Come già osservato per l'algoritmo delle k-medie, può accadere che l'algoritmo raggiunga un massimo che non è garantito essere il massimo globale ed una convergenza a massimi locali poco significativi è un problema legato ad una sensibilità dell'algoritmo all'inizializzazione. Ad esempio, gli outliers (ovvero documenti dissimili da tutti gli altri documenti della collezione) costituiscono un ostacolo soprattutto se scelti come rappresentanti iniziali dei cluster, poiché in tal caso si potrebbe ottenere un cluster contenente un singolo elemento. Un modo per ovviare il problema dei massimi locali poco rilevanti è quello di far girare l'algoritmo più volte e tenere in considerazione la partizione migliore, ovvero la partizione avente valore più alto della funzione obiettivo.

2.3.3 Stima del numero di cluster

Uno degli svantaggi dell'algoritmo delle k-medie (e quindi anche delle k-medie sferiche) è la necessità di una conoscenza a priori sul numero k di cluster in cui suddividere l'insieme dei dati. Qualora non si abbia questa informazione, esistono metodi che consentono

di stimare k. Il più semplice e intuitivo fra questi consiste nell'eseguire l'algoritmo per diversi valori di k, costruire il grafico con i valori della funzione obiettivo al variare di k ed infine considerare il k in corrispondenza del quale il grafico presenta un gomito. Più precisamente, nel caso delle k-medie sferiche: per ogni k si calcolano i partizioni di cardinalità k, ciascuna con una diversa inizializzazione e si calcola $\mathcal{Q}(k)$ per ognuna di queste. Sia $\mathcal{Q}_{\max}(k)$ il massimo fra gli i valori di $\mathcal{Q}(k)$. Al crescere di k si riportano su un grafico i valori di $\mathcal{Q}_{\max}(k)$ e si stima il numero di cluster opportuno osservando il gomito nella curva.

Di seguito è riportata una applicazione di tale metodo al dataset di documenti `A_med` del quale non è nota la cluster structure. Si tenga presente che la funzione `kmeans` di Matlab con la quale è stato effettuato il clustering non utilizza la misura di similarità del coseno analizzata fino ad ora, bensì la misura di distanza

$$d(x, c) = 1 - \frac{x^T c}{\|x\|_2 \|c\|_2} = 1 - \cos\theta,$$

dove θ è l'angolo fra x e c. Si ha allora una riformulazione del problema delle k-medie sferiche in termini di minimo. Infatti:

$$\begin{aligned} \sum_{j=1}^{k} \sum_{d \in \pi_j} \Big(1 - \frac{d^T c_j}{\|d\|_2 \|c_j\|_2}\Big) &= \sum_{j=1}^{k} n_j - \sum_{j=1}^{k} \sum_{d \in \pi_j} \frac{d^T c_j}{\|d\|_2 \|c_j\|_2} \quad = \\ &= n - \sum_{j=1}^{k} \sum_{d \in \pi_j} \frac{d^T c_j}{\|d\|_2 \|c_j\|_2} \end{aligned} \tag{2.3}$$

e, dunque, il problema di massimo (2.1) equivale al problema di minimo

$$\{\pi_j^*\}_{j=1}^k = \arg \min_{\{\pi_j\}_{j=1}^k} \sum_{j=1}^{k} \sum_{d \in \pi_j} \Big(1 - \frac{d^T c_j}{\|d\|_2 \|c_j\|_2}\Big).$$

Pertanto, la procedura per la valutazione di k descritta sopra è stata eseguita considerando di volta in volta il minimo. In Figura 2.3 è riportato il grafico dei valori della funzione obiettivo (2.3) al variare di k, ottenuto con il metodo spiegato precedentemente. Si può allora stimare il numero di cluster di `A_med` intorno a 25.

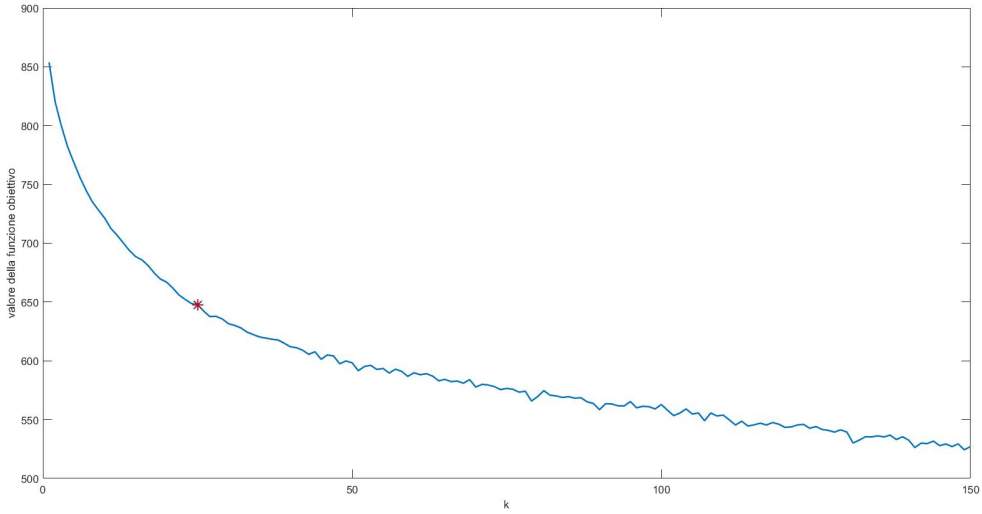

Figura 2.3: *Plot dei valori della funzione obiettivo* (2.3) *al variare di* k *per il dataset A_med. Il simbolo* $*$ *evidenzia il valore della funzione in corrispondenza di* $k = 25$.

2.3.4 Applicazione del metodo al dataset A_med e interpretazione dei risultati

Se il document clustering, raggruppando per argomento i documenti di una collezione, permette di organizzare tale collezione, facilitando così la visualizzazione, la ricerca e la classificazione, è utile che ciascun cluster sia descritto da una etichetta (*label*) che ne colga il contenuto in modo che la suddivisione sia comprensibile all'utente. Il metodo presentato non genera automaticamente tali etichette, tuttavia è possibile utilizzare i concept vector per accedere al contenuto dei cluster. Infatti, per come sono rappresentati i documenti con il modello vettoriale e per definizione di centroide, la i-esima coordinata del concept vector c relativo al cluster π è la frequenza media della i-esima parola chiave nei documenti appartenenti a π. Allora, se si considerano le coordinate dei concept vector di valore maggiore, si ottengono i termini mediamente più frequenti (e quindi caratterizzanti) all'interno del cluster, andando così a costruire un vocabolario che descrive il contenuto del cluster π. Occorre però notare come tale metodo non sia sempre efficace poiché può accadere che alcuni termini siano altamente frequenti all'interno di un cluster ma poco significativi e descrittivi dell'argomento.

È stato eseguito il clustering del dataset `A_med` (già descritto nel Capitolo 1) con il metodo delle k-medie sferiche per $k = 25$ sulla base della stima precedentemente

effettuata. Dopodiché, per ciascuno dei 25 concept vector sono state ordinate le parole chiave in ordine decrescente di frequenza. Nella Tabella 2.1 sono visualizzabili le prime 10 parole chiave di ciascun concept vector. Si possono notare cluster più coerenti e con un contenuto più evidente rispetto ad altri; ad esempio, è chiaro che il cluster 5 tratti di ipotermia, il cluster 9 di problematiche pediatriche, il cluster 13 di apparato respiratorio e il cluster 14 di tumori, al contrario di altri (ad esempio i cluster 8, 12, 15 e 17) che non appaiono particolarmente coesi e l'argomento è piuttosto vago. Questo può essere riconducibile al fatto che il numero di cluster scelto non sia quello più appropriato (si tratta, infatti, di una stima), oppure che emergano termini particolarmente frequenti ma poco descrittivi del tema (questo problema è eventualmente migliorabile con una più attenta fase di preprocessing e schema di pesi per il modello vettoriale, ricordando, infatti, che `A_med` ha come pesi le sole frequenze delle parole nei documenti), oppure che la partizione determinata dall'algoritmo sia solo un massimo locale.

cluster		cluster	
1	hydrocephalus, congenital, cases, case, malformations, shunt, surgical, clinical, infants, syndrome	14	cancer, patients, breast, treatment, chemotherapy, therapy, excretion, carcinoma, urinary, advanced
2	diabetes, insipidus, patients, vasopressin, antidiuretic, sodium, treatment, water, urine, volume	15	tumor, specific, antigen, cells, tumour, mice, virus, antigens, found, growth
3	selenium, infants, embryos, toxic, level, treatment, levels, months, selenate, selenite	16	microfilariae, normal, method, patients, filariasis, group, antigen, methods, positive , test
4	fluid, present, cell, exchange, observations, cerebrospinal, bone, concentrations, normal, lesion	17	cases, renal, clinical, present, patients, plasma, results, reported, case, sle
5	hypothermia, blood, perfusion, rate, cardiac, temperature, brain, cooling, circulation, volume	18	dna, subtilis, phage, bacillus, synthesis, rna, transforming, cells, acid, genetic
6	cells, marrow, cell, bone, lymphocytes, normal, type, tissue, irradiation, irradiated	19	fatty, acids, glucose, acid, plasma, free, increased, serum, maternal, pregnancy
7	case, patient, visual, presented, patients, report, bilateral, bitemporal, hemianopia, surgical	20	jaundice, biliary, atresia, neonatal, bilirubin, hepatitis, liver, obstructive, extrahepatic, infants
8	patients, therapy, amyloidosis, treatment, disease, cases, patient, clinical, results, renal	21	lens, crystallin, protein, activity, fractions, proteins, soluble, acid, fraction, lenses
9	children, child, autistic, language, speech, treatment, early, group, development, autism	22	human, mycoplasma, cell, strains, strain, cultures, tissue, mycoplasmas, hominis, bone
10	kidney, renal, rats, nephrectomy, hypertrophy, compensatory, increased, unilateral, animals, increase	23	growth, hormone, hgh, parathyroid, human, effect, insulin, treatment, plasma, serum
11	nickel, found, contact, eczema, carbonyl, sensitization, skin, action, nicl2, demonstrated	24	ventricular, aortic, septal, defect, left, pulmonary, patients, pressure, lesions, cases
12	disease, hemophilia, factor, mice, normal, viii, persons, organisms, plasma, found	25	oxygen, pressure, cerebral, blood, flow, tension, response, rats, arterial, po2
13	alveolar, lung, pulmonary, surface, epithelial, cells, electron, lungs, bodies, membrane		

Tabella 2.1: I 10 termini più rilevanti dei concept vector di A_med.

Capitolo 3

Riduzione delle dimensioni mediante centroidi

Se uno dei problemi dell'information retrieval è la gestione di grandi moli di dati, ridurne le dimensioni risulta essere di fondamentale importanza per migliorare l'efficienza computazionale, soprattutto per fini quali il document retrieval e la classificazione, nei quali il confronto fra vettori documento è causa di un elevato costo computazionale. Nel seguito sono mostrati dei metodi per la riduzione delle dimensioni della matrice di termine-documento che sfruttano una informazione a priori sulla collezione di documenti, ovvero la sua cluster structure. In particolare, tali metodi risultano essere più efficienti ed efficaci della SVD ai fini della classificazione.

3.1 Rappresentazione di rango basso della matrice di termine-documento

Siano $A \in \mathbb{R}^{m\times n}$ la matrice di termine-documento e $k \ll \min\{m,n\}$ il numero di dimensioni desiderato. Si vogliono determinare due matrici B e Y tali che

$$A \approx BY \tag{3.1}$$

dove $B \in \mathbb{R}^{m\times k}$ con $\text{rank}(B) = k$ e $Y \in \mathbb{R}^{k\times n}$ con $\text{rank}(Y) = k$. La matrice Y è la rappresentazione di A nello spazio ridotto k-dimensionale.

Osservazione 3.1. L'approssimazione (3.1) non è unica. Infatti, presa una qualsiasi matrice non singolare $Z \in \mathbb{R}^{k\times k}$ si ha

$$A \approx BY = (BZ)(Z^{-1}Y),$$

dove $BZ \in \mathbb{R}^{m\times k}$ con $\text{rank}(BZ) = k$ e $Z^{-1}Y \in \mathbb{R}^{k\times n}$ con $\text{rank}(Z^{-1}Y) = k$.

Il problema di determinare le due matrici B ed Y come in (3.1) può essere formulato in termini di un problema di minimo: si vogliono calcolare B e Y che miniminzzino l'errore dell'approssimazione $A \approx BY$ in una certa norma p, ovvero si vuole risolvere

$$\min_{B,Y} \|A - BY\|_p \tag{3.2}$$

dove $B \in \mathbb{R}^{m\times k}$ con $\text{rank}(B) = k$ e $Y \in \mathbb{R}^{k\times n}$ con $\text{rank}(Y) = k$. Nel seguito si assumerà $p = 2$.

È noto dal Teorema 1.2 che la migliore approssimazione di A di rango k in norma 2 è data dalla SVD di A troncata alle prime k componenti singolari. Dunque, una soluzione per il problema di minimo (3.2) può essere ottenuta dalla SVD troncata di A prendendo $B = U_k$ e $Y = \Sigma_k V_k^T$ e questa è proprio la soluzione ottimale per $p = 2$. Dopodiché, dato un qualsiasi vettore documento $q \in \mathbb{R}^m$, se si vuole calcolare la sua rappresentazione $\hat{q}$ nello spazio ridotto k-dimensionale, occorre risolvere il problema ai minimi quadrati

$$\min_{\hat{q}\in\mathbb{R}^k} \|q - B\hat{q}\|_2 = \min_{\hat{q}\in\mathbb{R}^k} \|q - U_k\hat{q}\|_2 \tag{3.3}$$

da cui si ottiene

$$\hat{q} = U_k^T q.$$

Tuttavia, occore fare una distinzione fra migliore *approssimazione* e migliore *rappresentazione* di rango basso della matrice di termine-documento. Infatti, per ottenere buoni risultati di efficienza computazionale è spesso necessario ridurre le dimensioni drasticamente e questo può portare ad una notevole perdita dell'informazione contenuta nei dati originari. Pertanto, è preferibile cercare di ottenere una migliore rappresentazione dei dati in funzione di uno specifico scopo piuttosto che ridurre le dimensioni per approssimare al meglio l'intera matrice di termine-documento. Per fare ciò si utilizza una opportuna informazione a priori sull'insieme dei dati, la quale può essere tradotta nella scelta delle matrici B e Y. Nella prossima sezione verrà mostrato come incorporare una informazione a priori sulla "cluster structure" della matrice di termine-documento e come tale scelta si riveli particolarmente efficace per la classificazione.

3.2 Riduzione di dimensioni per dati cluster structured e classificazione

Siano $\pi_1, \ldots, \pi_k$ i k cluster in cui è suddivisa la collezione di documenti. Nel caso la partizione non fosse già nota, si raggruppano i documenti utilizzando un algoritmo di clustering come, ad esempio, l'algoritmo delle k-medie sferiche presentato nel Capitolo 2. Ricordando la definizione di centroide del cluster π_i

$$c_i = \sum_{d \in \pi_i} d,$$

si definisce la *matrice dei centroidi*

$$C = [c_1, \ldots, c_k] \in \mathbb{R}^{m \times k},$$

ovvero la matrice avente come i-esima colonna il centroide dell'i-esimo cluster. L'idea è di prendere $B = C$ in (3.1), cioè di approssimare ciascun vettore documento della matrice A con una combinazione lineare dei centroidi dei suoi cluster:

$$a_i \approx Cy_i = c_1 y_{i,1} + c_1 y_{i,2} + \cdots + c_k y_{i,k}.$$

Dunque, la matrice dei centroidi C può essere utilizzata come una base approssimata per lo "spazio dei documenti", dopodiché, per il document retrieval e per la classificazione è necessario determinare le coordinate di tutti i documenti di A rispetto a tale base, ovvero determinare la matrice Y dell'approssimazione $A \approx CY$. Per fare ciò si risolve il problema ai minimi quadrati

$$\min_{Y \in \mathbb{R}^{k \times n}} \|A - CY\|_2. \tag{3.4}$$

La soluzione $Y \in \mathbb{R}^{k \times n}$ fornisce la rappresentazione di A nello spazio k-dimensionale. Un qualsiasi nuovo documento (o query) $q \in \mathbb{R}^m$ può essere trasformato nello spazio ridotto risolvendo il problema ai minimi quadrati

$$\min_{\hat{q} \in \mathbb{R}^k} \|q - C\hat{q}\|_2.$$

Quanto detto può essere riassunto nel seguente algoritmo:

Algoritmo 2: Algoritmo dei Centroidi

Input: $A \in \mathbb{R}^{m \times n}$ matrice dei dati con k cluster e $q \in \mathbb{R}^m$

Output: Rappresentazione k-dimensionale $\hat{q} \in \mathbb{R}^k$ di q

1 Per $i = 1, \ldots, k$: calcola il centroide c_i dell'i-esimo cluster;

2 Costruisci $C = [c_1, \ldots, c_k]$ matrice dei centroidi;

3 Risolvi $\min_{\hat{q}} \|q - C\hat{q}\|_2$.

Osservazione 3.2. È stato fino ad ora supposto che la matrice dei centroidi C abbia rango pieno k. Qualora dovesse accadere che C abbia rango $r < k$, siccome l'obiettivo è risolvere un problema ai minimi quadrati, si può procedere come spiegato nel Capitolo 1: si calcola la SVD di C

$$C = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

con $\Sigma_1 \in \mathbb{R}^{r\times r}$ non singolare e $\hat{q} = V_1\Sigma_1^{-1}U_1^T q$ è soluzione del problema di minimo

$$\min_{\hat{q}} \|q - C\hat{q}\|_2.$$

Inoltre, per il Teorema 1.3, tale $\hat{q}$ è la soluzione di norma minima. Nel seguito si continuerà a suppore $\text{rank}(C) = k$.

Come già precisato, la riduzione delle dimensioni della matrice di termine-documento può essere utile per rendere più efficiente la classificazione. Quest'ultima può essere effettuata utilizzando l'algoritmo che segue, il quale prevede di assegnare un nuovo documento al cluster avente centroide più vicino.

Algoritmo 3: Classificazione mediante centroidi

Input: $A \in \mathbb{R}^{m\times n}$ matrice dei dati con k cluster, c_i, $i = 1, \ldots, k$ i corrispondenti centroidi e $q \in \mathbb{R}^m$ un nuovo vettore documento

Output: Indice j del cluster cui q appartiene

1 Determina l'indice j del cluster tale che

$$j = \arg \max_{i=1,\ldots,k} \frac{q^T c_i}{\|q\|_2\|c_i\|_2}.$$

Equivalentemente si può scegliere una misura di distanza (ad esempio la distanza Euclidea) e determinare l'indice j corrispondente al valore minimo. Si vuole ora sfruttare la riduzione delle dimensioni, ottenuta mediante l'algoritmo dei Centroidi, per la classificazione. Si noti che la soluzione del problema ai minimi quadrati

$$\min_y \|c_i - Cy\|_2,\ i = 1, \ldots, k$$

è $e_i \in \mathbb{R}^k$, ovvero la i-esima colonna della matrice identità e ciò significa che i centroidi sono rappresentati dai vettori della base canonica nello spazio ridotto. Quindi, se $\hat{q}$ è la

rappresentazione k-dimensionale di un vettore documento q, ottenuta con Algoritmo 2, nell'algoritmo di classificazione mediante centroidi si ricerca

$$\arg \max_{i=1,\ldots,k} \frac{\hat{q}^T e_i}{\|q\|_2 \|e_i\|_2}$$

e il massimo verrà raggiunto quando j è l'indice della più grande componente in valore assoluto di $\hat{q}$. Anche utilizzando la distanza Euclidea, l'indice del cluster è dato dalla più grande componente di $\hat{q}$. Ciò significa che, effettuando la riduzione di dimensioni descritta precedentemente, si può sostituire il calcolo di k misure di similarità del coseno con la sola ricerca della più grande fra le k componenti di $\hat{q}$ e questo si traduce in una notevole riduzione del costo computazionale ($O(k)$ invece di $O(mk)$).

Se la matrice B dell'approssimazione $A \approx BY$ ha colonne ortonormali, allora per ogni colonna a_i, $i = 1, \ldots, n$ di A e vettore documento q si ha

$$cos(a_i, q) \approx cos(By_i, B\hat{q}) = cos(y_i, \hat{q})$$

e dunque il confronto di due vettori documento può essere semplicemtente ricondotto al confronto delle loro rappresentazioni nello spazio ridotto, senza quindi il coinvolgimento della matrice B. A prescindere da come sono scelte B e Y, si può ottenere tale risultato con la fattorizzazione QR di B:

$$B = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

dove $Q = \big(Q_k, Q_r\big) \in \mathbb{R}^{m \times m}$ ortogonale, $Q_k \in \mathbb{R}^{m \times k}$ e $R \in \mathbb{R}^{k \times k}$ triangolare superiore. La forma $B = Q_k R$ è detta *fattorizzazione QR ridotta*. Si consideri ancora una volta $B = C$ e si effettui la fattorizzazione QR ridotta: $C = Q_k R$. Allora

$$A \approx CY = Q_k(RY).$$

Pertanto, se si vuole ottenere la rappresentazione k-dimensionale $\hat{q}$ di $q \in \mathbb{R}^m$, si risolve il problema ai minimi quadrati

$$\min_{\hat{q}} \|q - Q_k \hat{q}\|_2.$$

Quanto detto può essere riassunto nel seguente algoritmo:

Algoritmo 4: Algoritmo dei Centroidi Ortogonali

Input: $A \in \mathbb{R}^{m \times n}$ matrice dei dati con k cluster e $q \in \mathbb{R}^m$

Output: Rappresentazione k-dimensionale $\hat{q} \in \mathbb{R}^k$ di q

1 Per $i = 1, \ldots, k$: calcola il centroide c_i dell'i-esimo cluster;

2 Costruisci $C = [c_1, \ldots, c_k]$ matrice dei centroidi;

3 Calcola la fattorizzazione QR ridotta di C: $C = Q_k R$;

4 $\hat{q} = Q_k^T q$.

Osservazione 3.3. Indicate con $\hat{q}_c$ la rappresentazione ottenuta tramite l'algoritmo dei Centroidi e con $\hat{q}_{co}$ la rappresentazione ottenuta con l'algoritmo dei Centroidi Ortogonali, queste sono legate dalla relazione $\hat{q}_{co} = R\hat{q}_c$. Infatti, $\hat{q}_c$ è la soluzione del problema di minimo

$$\min_{\hat{q}} \|q - C\hat{q}\|_2$$

che, risolto mediante la fattorizzazione QR di C, ha soluzione

$$\hat{q}_c = R^{-1} Q_k^T q = R^{-1} \hat{q}_{co}.$$

Si può notare come il clustering, congiuntamente alla classificazione, possa essere utilizzato ai fini del document retrieval poiché permette di ottimizzare la ricerca dei documenti più rilevanti: una volta organizzata la collezione di documenti in cluster, si determina quello più pertinente alla query (ad esempio tramite l'algoritmo di classificazione mediante centroidi) e si conduce la ricerca sui soli documenti appartenenti a quel cluster, riducendo così il numero di confronti ed escludendo automaticamente i documenti del tutto irrilevanti. Inoltre, se si combina tale procedimento con il metodo di riduzione delle dimensioni proposto, si può ottenere una ulteriore diminuzione del costo computazionale per la ricerca dei documenti rilevanti.

3.2.1 Invarianza della classificazione per riduzione di dimensioni mediante Centroidi Ortogonali

Si vuole provare che la classificazione effettuata in termini dei centroidi (Algoritmo 3) fornisce lo stesso risultato sia che essa venga applicata ai full data, sia che essa venga applicata ai dati ridotti mediante l'Algortimo 4 dei Centroidi Ortogonali.

Definizione 3.1 (Ordine di similarità)**.** *Dati un qualsiasi vettore $q \in \mathbb{R}^m$ e una matrice $B \in \mathbb{R}^{m\times k}$, si definisce l'ordine di similarità $S(q, B)$ come la lista di indici delle colonne di B, ordinati in ordine crescente di similarità fra q e le k colonne di B.*

Il seguente teorema mostra che l'ordine di similarità fra un qualsiasi vettore e la matrice dei centroidi nello spazio pieno è conservato nello spazio ridotto ottenuto dall'algoritmo dei Centroidi Ortogonali.

Teorema 3.1. *Dati $q \in \mathbb{R}^m$ un vettore qualsiasi e $C \in \mathbb{R}^{m\times k}$ la matrice dei centroidi, siano $\hat{q} = Q_k^T q$ e $\hat{C} = Q_k^T C$ le loro rappresentazioni k dimensionali ottenute mediante l'algoritmo dei Centroidi Ortogonali. Allora*

$$S(q, C) = S(\hat{q}, \hat{C}).$$

Dimostrazione. Si dimostra la tesi per la misura di similarità del coseno, ma un'argomentazione analoga vale anche per la norma Euclidea.
Si consideri la fattorizzazione QR di C:

$$C = \begin{pmatrix} Q_k & Q_r \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix}.$$

Preso $i \in \{1, \dots, k\}$, per l'ortogonalità di Q e siccome $Q_r^T c_i = 0$ si ha

$$\begin{aligned} cos(q, c_i) = cos(Q^T q, Q^T c_i) &= \frac{(Q^T q)^T Q^T c_i}{\|Q^T q\|_2 \|Q^T c_i\|_2} = \\ &= \frac{1}{\|Q^T q\|_2 \|Q^T c_i\|_2} \begin{pmatrix} q^T Q_k & q^T Q_r \end{pmatrix} \begin{pmatrix} Q_k^T c_j \\ 0 \end{pmatrix} = \frac{q^T Q_k Q_k^T c_i}{\|Q^T q\|_2 \|Q_k^T c_i\|_2} \end{aligned}$$

e

$$cos(\hat{q}, \hat{c}_i) = cos(Q_k^T q, Q_k^T c_i) = \frac{q^T Q_k Q_k^T c_i}{\|Q_k^T q\|_2 \|Q_k^T c_i\|_2}.$$

Se $cos(q, c_i) \leq cos(q, c_j)$ allora

$$\frac{q^T Q_k Q_k^T c_i}{\|Q^T q\|_2 \|Q_k^T c_i\|_2} \leq \frac{q^T Q_k Q_k^T c_j}{\|Q^T q\|_2 \|Q_k^T c_j\|_2}.$$

Di conseguenza,

$$\frac{q^T Q_k Q_k^T c_i}{\|Q_k^T q\|_2 \|Q_k^T c_i\|_2} \leq \frac{q^T Q_k Q_k^T c_j}{\|Q_k^T q\|_2 \|Q_k^T c_j\|_2}.$$

ovvero

$$cos(\hat{q}, \hat{c}_i) \leq cos(\hat{q}, \hat{c}_j)$$

□

Osservazione 3.4. La conservazione dell'ordine di similarità è un discorso di carattere generale, è vera a prescindere dalla qualità del clustering. Infatti, nella dimostrazione del Teorema 3.1 non viene utilizzato il fatto che le colonne della matrice C sono i centroidi dei cluster. Il teorema è vero per una qualsiasi matrice B, basta che la riduzione delle dimensioni venga effettuata tramite una base ortogonale Q_B per il range di B.

3.3 Risultati sperimentali

In questa sezione vengono presentati alcuni risultati che mostrano come il modello di riduzione delle dimensioni basato sull'utilizzo di una informazione a priori sulla cluster structure della matrice di termine-documento sia particolarmente efficace per la classificazione e migliore della SVD sia dal punto di vista dell'accuratezza della classificazione che dell'efficienza computazionale. Nei test condotti verrà utilizzata la matrice di termine-documento `A_med`, nonché una matrice di 30 queries `Q_med`$\in \mathbb{R}^{5735\times 30}$.

In questo primo test si vuole mostrare che l'ordine di similarità fra un qualsiasi vettore documento e la matrice dei centroidi è lo stesso sia nello spazio pieno che in quello ridotto con algoritmo dei Centroidi Ortogonali (da cui segue che i risultati di classificazione sono uguali nei due spazi). È stato effettuato il clustering della matrice `A_med` tramite algoritmo delle k-medie sferiche con $k = 5$ e sono stati presi quattro vettori query provenienti dalla matrice `Q_med`, $q_i =$`Q_med(:,i)`, $i = 1, \ldots, 4$. Dopodiché sono state calcolate le distanze di ciascun vettore q_i dai centroidi dei cluster sia nello spazio pieno che in quello ridotto mediante Centroidi Ortogonali (Algoritmo 4). Tali distanze sono riportate in Tabella 3.1 e in Tabella 3.2, rispettivamente con la similarità del coseno e con la distanza Euclidea sia nello spazio pieno che in quello ridotto; inoltre, in corsivo è evidenziata la misura che indica il centroide più vicino, ovvero la più alta per la similarità del coseno e la più bassa per la distanza Euclidea. Nell'ultima colonna di entrambe le tabelle sono mostrati gli ordini di similarità nello spazio pieno e in quello ridotto: come ci si aspettava dal Teorema 3.1, essi non cambiano in seguito a riduzione di dimensione tramite Centroidi Ortogonali.

	$cos(q_i, c_j)$					$S(q_i, C)$
i	c_1	c_2	c_3	c_4	c_5	
1	0.0111	*0.0727*	0.0055	0.0356	0.0049	$(2, 4, 1, 3, 5)$
2	*0.1420*	0.0472	0.0659	0.0641	0.0425	$(1, 3, 4, 2, 5)$
3	0.0260	0.0163	0	*0.1056*	0.0252	$(4, 1, 5, 2, 3)$
4	0.0331	*0.1274*	0.0310	0.0060	0.0439	$(2, 5, 1, 3, 4)$
	$cos(Q_k^T q_i, Q_k^T c_j)$					$S(\hat{q}_i, \hat{C})$
1	0.1444	*0.9481*	0.0712	0.4642	0.0636	$(2, 4, 1, 3, 5)$
2	*0.9782*	0.3254	0.4537	0.4414	0.2926	$(1, 3, 4, 2, 5)$
3	0.2367	0.1487	3.32e-17	*0.9627*	0.2302	$(4, 1, 5, 2, 3)$
4	0.2535	*0.9768*	0.2373	0.0462	0.3367	$(2, 5, 1, 3, 4)$

Tabella 3.1

	$\|q_i - c_j\|_2$					$S(q_i, C)$
i	c_1	c_2	c_3	c_4	c_5	
1	2.0085	*1.9965*	2.0215	2.0097	2.0197	$(2, 1, 4, 5, 3)$
2	*3.1380*	3.1607	3.1567	3.1559	3.1622	$(1, 4, 3, 2, 5)$
3	1.7399	1.7423	1.7538	*1.7264*	1.7522	$(4, 1, 2, 5, 3)$
4	2.2404	*2.2180*	2.2487	2.2496	2.2376	$(2, 5, 1, 3, 4)$
	$\|Q_k^T(q_i - c_j)\|_2$					$S(\hat{q}_i, \hat{C})$
1	0.2402	0.0971	0.3314	0.2502	0.3204	$(2, 1, 4, 5, 3)$
2	0.2403	0.4478	0.4191	0.4128	0.4583	$(1, 4, 3, 2, 5)$
3	0.2514	0.2678	0.3345	0.1285	0.3260	$(4, 1, 2, 5, 3)$
4	0.3233	*0.0680*	0.3761	0.3820	0.3031	$(2, 5, 1, 3, 4)$

Tabella 3.2

Nel secondo test si vuole evidenziare l'efficacia del metodo di riduzione delle dimensioni per la classificazione. È stato eseguito il clustering della matrice `A_med` tramite l'algoritmo delle k-medie sferiche con $k = 25$. Dopodiché è stato suddiviso il dataset `A_med` in training set e test set ponendo nel training set l'80% dei documenti di `A_med` scelti in modo random tramite la funzione `datasample` di Matlab; il restante 20% dei documenti sono stati utilizzati come test set. Dunque, le matrici del training set e del test set sulle quali è stato condotto l'esperimento hanno dimensioni, rispettivamente, 5735×826 e 5735×207. Sono stati ricalcolati i centroidi dei cluster utilizzando i soli documenti appartenenti al training set; si osservi che la matrice dei centroidi ha rango pieno 25. In seguito è stata eseguita la classificazione per 10 scelte random di training set e test set con le dimensioni di cui sopra. La classificazione è stata effettuata mediante centroidi (Algoritmo 3) con la similarità del coseno e con la distanza Euclidea nello spazio pieno e nello spazio ridotto con l'algoritmo dei Centroidi (Algoritmo 2) e dei Centroidi Ortogonali (Algoritmo 4). In Tabella 3.3 sono riportate le percentuali medie di buona allocazione per entrambe le misure[1]. Ancora una volta si può notare come la classificazione nello spazio pieno e in quello ridotto con i Centroidi Ortogonali fornisca lo stesso risultato.

Accuratezza della classificazione (in %)			
	Full	Centroidi	Centroidi Ortogonali
Similarità del coseno	95.36	94.44	95.36
Distanza Euclidea	94.40	93.38	94.40

Tabella 3.3: Numero di cluster $k = 25$. Classificazione effettuata mediante centroidi sia nello spazio pieno che nello spazio k-dimensionale ottenuto tramite algoritmo dei Centroidi e algoritmo dei Centroidi Ortogonali.

Si vogliono ora confrontare i tre metodi per la riduzione di dimensioni presentati in questi capitoli: Centroidi, Centroidi Ortogonali e SVD. Ancora una volta è stato effettuato il clustering di `A_med` con l'algoritmo delle k-medie sferiche per $k = 25$ ed è stato suddiviso il dataset in training set (di dimensioni 5735×826) e test set (di dimensioni 5735×207) come descritto precedentemente; si osservi che la matrice dei centroidi ha rango pieno 25. È stata poi calcolata la SVD di `A_med` troncata a diversi valori $l = 25, 50, 100, 150, 200$ ed è stata calcolata la rappresentazione l-dimensionale

[1]Lo stesso esperimento condotto con diverse dimensioni di training set e test set fornisce risultati analoghi e, pertanto, sono mostrati esclusivamente quelli ottenuti con la suddivisione descritta.

dei centroidi risolvendo il problema ai minimi quadrati (3.3) con U_l. Infine, è stata effettuata la classificazione mediante centroidi con la similarità del coseno nello spazio ridotto l-dimensionale tramite SVD per 10 scelte random di training set e test set con le dimensioni di cui sopra. In Tabella 3.4 sono riportate le percentuali medie di buona allocazione al variare di l e si può notare che, mentre la classificazione in seguito alla riduzione di dimensioni con algoritmo dei Centroidi e dei Centroidi Ortogonali fornisca ottimi risultati anche per riduzioni drastiche delle dimensioni (si veda la Tabella 3.3), l'efficacia della classificazione in seguito a riduzione con SVD aumenta con l'aumentare della dimensione. In particolare, essa diventa più accurata per dimensioni superiori a 100 e questo può essere riconducibile al fatto che, come già osservato nel Capitolo 1 (Figura 1.3) una stima per un valore al quale troncare la SVD per mantenere l'informazione è fra 100 e 200. Per quanto riguarda le dimensioni del training set e del test set vale ancora quanto specificato nella nota 1 a fronte.

Accuratezza della classificazione (in %) con similarità del coseno	
Dimensione ridotta (l)	SVD
25	76.52
50	84.59
100	92.27
150	93.96
200	95.60

Tabella 3.4: Numero di cluster $k = 25$. Classificazione effettuata mediante centroidi nello spazio ridotto l-dimensionale in seguito ad SVD troncata, per diversi valori di l.

I due metodi di riduzione delle dimensioni presentati, oltre ad essere più efficaci per la classificazione rispetto alla SVD, hanno anche il vantaggio di essere significativamente meno costosi. A tal proposito è stato effettuato il clustering di `A_med` con l'algoritmo delle k-medie sferiche per diversi valori di k e si osserva che il rango della matrice dei centroidi è pieno per ciascuno di tali valori. Nella Tabella 3.5 sono confrontati i tempi di classificazione al variare di k (la dimensione ridotta) di un vettore query, sia esso $q =$`Q_med(:,15)`. La classificazione è stata effettuata mediante i centroidi. Nel calcolo dei tempi di calcolo sono stati inclusi anche i costi per il calcolo della fattorizzazione QR della matrice dei centroidi C e della SVD di `A_med`.

Tempi di riduzione e classificazione (in secondi)			
k	Centroidi	Centroidi Ortogonali	SVD
5	0.0002763	0.00063569	0.0404584
50	0.00502828	0.00608778	0.25966
100	0.0123939	0.0130706	0.73132
150	0.0326628	0.0261413	1.88362
200	0.0584359	0.0370388	3.14162

Tabella 3.5

Bibliografia

[1] Anil K. Jain, Richard C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

[2] Barbara Rosario, *Latent Semantic Indexing: An overview*, Infosys 240 Spring 2000.

[3] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, *Introduction to information retrieval*, Cambridge University Press, 2009.

[4] David C. Anastasiu, Andrea Tagarelli, George Karypis, *Document Clustering: The Next Frontier*, in *Data Clustering. Algorithms and Applications*, edito da Charu C. Aggarwal, e Chandan K. Reddy, CRC Press LLC, 2013.

[5] Haesun Park, Moongu Jeon, J. Ben Rosen, *Lower dimensional representation of text data based on centroids and least squares*, BIT Numerical Mathematics, 2003.

[6] Inderjit S. Dhillon, Dharmendra S. Modha, *Concept Decomposition for Large Sparse Text Data using Clustering*, in Machine Learning 42, 2001.

[7] Inderjit S. Dhillon, James Fan, Yuqiang Guan, *Efficient Clustering of Very Large Document Collections*, in *Data Mining for Scientific and Engineering Applications. Massive computing*, edito da R. L. Grossman, C. Kamath, P. Kegelmeyer, V. Kumar, R. R. Namburu, Springer, 2001.

[8] Lars Elden, *Matrix Methods in Data Mining and Pattern Recognition*, SIAM, April, 2007.

[9] M. Steinbach, G. Karypis, V. Kumar, *A comparison of document clustering techniques*, Department of Computer Science and Egineering, University of Minnesota, 2000.

[10] M. W. Berry, S. T. Dumais, G. W. O'Brien, *Using linear algebra for intelligent information retrieval*, SIAM Review, Vol. 37, 1995.

[11] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, *Modern Information Retrieval*, ACM Press, 1999.

[12] Ying Zhao, George Karypis, *Evaluation of hierarchical clustering algorithms for document datasets*, in *Proceedings of the eleventh international conference on Information and knowledge management*, 2002.