



SAPIENZA
UNIVERSITÀ DI ROMA

Equità nell'analisi dei gruppi mediante tecniche di fair clustering

Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Dipartimento di Scienze Statistiche
Corso di Laurea in Statistica, Economia e Società

Elisabetta Sanasi

Matricola 1942515

Relatore

Prof. Paolo Giordani

Correlatore

Prof.ssa Donatella Vicari

Anno Accademico 2022/2023

Tesi non ancora discussa

Equità nell'analisi dei gruppi mediante tecniche di fair clustering
Tesi di Laurea Triennale. Sapienza Università di Roma

© 2023 Elisabetta Sanasi. Tutti i diritti riservati

Questa tesi è stata composta con L^AT_EX e la classe Sapthesis.

Email dell'autore: sanasi.1942515@studenti.uniroma1.it

Alla mia famiglia

Sommario

Il presente lavoro si propone di analizzare una nuova tecnica di fair clustering, denominata *Attraction Repulsion clustering*.

Nel primo capitolo si delineano i concetti, gli strumenti e la metodologia tipici dell'analisi dei gruppi. La cluster analysis viene applicata in numerosissimi campi e, in generale, ha l'obiettivo di identificare dei gruppi omogenei all'interno di una popolazione di riferimento.

Nel secondo capitolo, si esamina il caso in cui, tra le variabili considerate, se ne abbia una o più definite sensibili: ossia, variabili rispetto alle quali l'omogeneità non è un aspetto desiderabile. Ne sono esempi il sesso, l'etnia o la nazionalità. Proprio in questo contesto, la tecnica di Attraction Repulsion clustering, ispirandosi al principio di attrazione e repulsione (da cui il nome) dell'elettromagnetismo, cerca di favorire la diversità, tradotta in termini di parità demografica, con rispetto alla variabile sensibile. Il capitolo si focalizza sulla definizione dei nuovi aspetti e dei nuovi strumenti, in particolare sulla proposta di nuove dissimilarità, che caratterizzano questa tecnica.

Dopo aver delineato tale cornice teorica, nel capitolo 3 viene trattato un caso di applicazione: la tecnica è applicata ad un dataset con informazioni relative alle scuole dello stato americano del Massachusetts. L'obiettivo è quello di determinare dei gruppi di scuole che siano compatti dal punto di vista geografico ma che, al tempo stesso, cerchino di favorire la diversità dal punto di vista della composizione etnica degli studenti. Il capitolo si conclude con delle riflessioni sui risultati ottenuti e con un confronto rispetto all'applicazione di una tecnica di cluster analysis tradizionale.

Indice

1 Cluster Analysis	1
1.1 Introduzione	1
1.2 Indici di distanza o dissimilarità	1
1.2.1 La ponderazione di variabili	4
1.2.2 La distanza di Mahalanobis	5
1.2.3 Indici di similarità	5
1.3 Analisi dei gruppi	7
1.4 Metodi gerarchici	7
1.4.1 Gerarchia di partizioni e distanze tra gruppi	7
1.4.2 Il dendrogramma	10
1.4.3 Caratteristiche delle partizioni	11
1.4.4 Scomposizione della devianza	11
1.5 Metodi non gerarchici	12
1.5.1 Il metodo delle k -medie	13
1.6 Confronto di partizioni	16
2 Attraction-repulsion clustering	19
2.1 Introduzione	19
2.2 Fair clustering	20
2.3 Nuove dissimilarità e interpretazione dei parametri	21
2.4 Il metodo di Ward nel fair clustering	23
2.5 Scaling multidimensionale e metodi Kernel	24
2.6 Misure di diversità e qualità	25
3 Caso studio	29
3.1 Introduzione e descrizione del dataset	29
3.2 Procedura classica	31
3.3 Attraction Repulsion	32
3.3.1 Applicazione delle nuove dissimilarità	33
3.4 Confronto e conclusioni	36
Bibliografia	39
Appendice A	41
Appendice B	45

Capitolo 1

Cluster Analysis

1.1 Introduzione

La cluster analysis è una tecnica di statistica multivariata che si pone l'obiettivo di classificare un insieme di n unità in un numero g di gruppi, definiti clusters. I criteri con cui vengono determinati i suddetti gruppi si basano su degli indici di distanza o indici di dissimilarità che permettono di individuare le unità più simili tra di loro: l'obiettivo è l'individuazione di gruppi che presentino la massima omogeneità al loro interno e la massima eterogeneità tra di essi. I contesti di applicazione sono innumerevoli: in ambito di marketing si potrebbe essere interessati a individuare dei segmenti omogenei nel mercato di un prodotto, per poi indirizzare una campagna pubblicitaria agli individui che non lo hanno ancora acquistato; in ambito medico, si potrebbe pensare di suddividere i pazienti in gruppi per poi, a scopo preventivo, prescrivere delle analisi in base al rischio di esposizione a una particolare patologia. Per la precisione, ciò che si vuole costruire è una partizione delle n unità di partenza in g gruppi omogenei. Per costituire una partizione i gruppi devono risultare esaustivi (1.1), disgiunti (1.2) e con una cardinalità pari almeno ad una unità (1.3). Ciò vuole dire che ogni gruppo dovrà contenere almeno 1 unità, che l'unione di tutti i gruppi dovrà restituire il collettivo di partenza e che una stessa unità non potrà appartenere contemporaneamente a più gruppi. In formule, usando la lettera C per indicare i cluster, si ha:

$$\bigcup_{c=1}^g C_c = \{1, 2, \dots, n\} \quad (1.1)$$

$$C_c \bigcap C_{c'} = \emptyset \quad c \neq c'; c, c' = 1, \dots, g \quad (1.2)$$

$$|C_c| \geq 1 \quad \forall c = 1, \dots, g \quad (1.3)$$

1.2 Indici di distanza o dissimilarità

A questo punto, risulta necessaria l'introduzione e l'illustrazione di una matrice dei dati $X_{n,k}$. Una qualsiasi indagine statistica, comporta l'individuazione di un collettivo di unità statistiche sulle quali si vanno a rilevare le modalità assunte da un insieme di variabili. Si indica il totale delle unità con n e ogni unità con

il generico indice i , perciò i può assumere valori interi tra 1 e n . Si hanno in tutto k variabili, indicate con la notazione generica X_j , con $j = 1, \dots, k$. Infine, il generico elemento x_{ij} della matrice dei dati rappresenta il valore che la variabile X_j assume in corrispondenza dell'unità i -esima. In riferimento quindi alla matrice $X_{n,k}$, il numero delle righe rappresenta il numero delle unità statistiche e il numero delle colonne il numero di variabili considerate. Come già accennato, alla base del meccanismo di indentificazione dei gruppi, è posta la definizione di un indice che misuri la prossimità o, analogamente, la diversità tra due unità, ossia tra due righe della matrice dei dati. In generale, tale indice sarà una funzione $f(i, i')$ applicata a due unità che indichiamo con i e i' . Tramite f è possibile effettuare il passaggio da n unità a un numero ristretto g di gruppi; ovviamente se f è un indice di diversità, si ritroveranno nello stesso gruppo unità per cui f assume valori bassi; al contrario, ma equivalentemente, se f è un indice di prossimità, si ritroveranno nello stesso cluster unità che presentavano valori elevati dell'indice utilizzato.

Le variabili X_j possono essere di varia natura: qualitative, quantitative o miste. In base a questa caratteristica, è possibile definire diversi indici f . Nel caso in cui le variabili siano esclusivamente quantitative, si possono introdurre i concetti di distanza, indici di distanza e indici di dissimilarità.

Definizione: una distanza è una particolare funzione $f(i, i') = d(i, i')$ che gode di 4 proprietà fondamentali: la proprietà di non-negatività (1.4), di identità (1.5), di simmetria (1.6) e rispetta inoltre la disuguaglianza triangolare (1.7).

$$d(i, i') \geq 0 \quad \forall i, i' \quad (1.4)$$

$$d(i, i) = 0 \quad (1.5)$$

$$d(i, i') = d(i', i) \quad (1.6)$$

$$d(i, i') \leq d(i, i'') + d(i'', i') \quad \forall i, i', i'' \quad (1.7)$$

Tra gli esempi di distanze, si ha la *distanza euclidea* (1.8) e la *distanza a blocchi* (1.9), detta anche di Manhattan. In formule si ha:

$$d_{E,ii'} = \left[\sum_{j=1}^k (x_{ij} - x_{i'j})^2 \right]^{1/2} \quad (1.8)$$

$$d_{B,ii'} = \sum_{j=1}^k |x_{ij} - x_{i'j}| \quad (1.9)$$

Immaginando per semplicità un caso in cui $k = 2$, è interessante notare come, graficamente, la distanza euclidea rappresenti il segmento che unisce due punti del piano cartesiano che rappresentano le due unità in considerazione; la distanza a blocchi coincide, invece, con la somma dei due cateti del triangolo rettangolo, la cui ipotenusa è proprio il segmento sopra descritto. La distanza a blocchi è definita anche di Manhattan proprio per questa caratteristica di potersi muovere solamente in orizzontale e verticale proprio come tra i grattacieli di una città americana. Si nota immediatamente come, di conseguenza, fissate due unità, la distanza a blocchi risulterà sempre maggiore o al limite uguale della distanza euclidea.

Per la precisione, entrambe tali distanze sono dei casi particolari generalizzabili nella *distanza di Minkowski*:

$$d_{M,l} = \left[\sum_{j=1}^k |(x_{ij} - x_{i'j})|^l \right]^{1/l} \quad (1.10)$$

L'indice l è detto *ordine* della distanza di Minkowski ed è immediato vedere come per $l = 2$ si ottenga il caso particolare della distanza euclidea e per $l = 1$ il caso particolare della distanza a blocchi.

La (1.10) gode della proprietà di invarianza per traslazione delle variabili. In formulæ:

$$d_{M,l}(x_i^T, x_{i'}^T) = d_{M,l}(x_i^T + c, x_{i'}^T + c) \quad (1.11)$$

Dove x_i^T e $x_{i'}^T$ indicano, rispettivamente, la riga i e la riga i' della matrice $X_{n,k}$ e per questo sono dei vettori di dimensione $1 \times k$; c rappresenta invece un vettore di costanti di dimensioni $k \times 1$. Infatti, dalla (1.11), si può vedere come

$$\forall j = 1, \dots, k \quad |(x_{ij} + c_j) - (x_{i'j} + c_j)|^l = |x_{ij} - x_{i'j}|^l$$

ossia come, per ogni variabile X_j di fatto, le costanti c_j si vadano ad eliminare a coppie: ciò implica che centrare le variabili, sottraendo il loro valore medio scegliendo $c_j = \mu_j \quad \forall j = 1, \dots, k$, non altera il valore della distanza tra le unità.

Tuttavia, la stessa proprietà di invarianza non è valida in caso di cambiamenti di unità di misura, che portano invece a delle alterazioni nei valori delle distanze. Come conseguenza, la cluster analysis conduce a risultati differenti se applicata a dati standardizzati o meno e da ciò deriva la necessità di standardizzare preventivamente i dati se le variabili presentano unità di misura diverse oppure ordini di grandezza molto distanti.

Rinunciando alla proprietà della disuguaglianza triangolare, non si ha più una distanza ma un *indice di distanza*.

Definizione: si definisce indice di distanza una particolare $f(i, i')$ che gode delle proprietà di non-negatività (1.4), di identità (1.5) e di simmetria (1.6).

Un esempio di indice di distanza è la *distanza euclidea al quadrato* (1.12):

$$d_{E,ii'}^2 = \sum_{j=1}^k (x_{ij} - x_{i'j})^2 \quad (1.12)$$

Già dalla loro formulazione, si può vedere come la (1.8) e la (1.12) accentuino le differenze più grandi tra i valori che le variabili assumono in corrispondenza delle unità considerate, poichè le elevano al quadrato. La (1.9) invece, si costruisce come somma dei moduli di tali differenze; può succedere, quindi, che le differenze maggiori siano compensate da quelle minori. Perciò sicuramente, se l'obiettivo è accentuare le grandi differenze, si deciderà di utilizzare la distanza euclidea o la distanza euclidea al quadrato.

Un'ultima categoria di indici è costituita dagli indici di dissimilarità.

Definizione: si definisce indice di dissimilarità una funzione $f(i, i')$ che soddisfa le proprietà di non negatività (1.4), di simmetria (1.6) e in più la seguente:

$$x_i = x_{i'} \implies d(x_i^T, x_{i'}^T) = 0 \quad (1.13)$$

La (1.13) è una condizione chiaramente più debole della proprietà di identità: afferma infatti che se due unità coincidono allora la loro distanza è 0, ma che si potrebbe ottenere un valore nullo della distanza anche se le unità in partenza fossero distinte.

1.2.1 La ponderazione di variabili

Si possono verificare dei casi in cui il ricercatore ritenga necessario assegnare un peso w_j ad ogni variabile X_j , per indicarne il diverso grado di importanza. È subito evidente come tale operazione si configuri come estremamente complessa essendo completamente soggettiva. Tra le motivazioni che potrebbero indurre a effettuare una ponderazione delle variabili, si può immaginare una situazione in cui si voglia dare maggiore peso a variabili che si ritengono diretta espressione del fenomeno sottostante in base al quale si effettua poi la classificazione. Altre volte si ha a disposizione una pluralità di variabili di cui solo una minima parte riferite a un particolare aspetto: può avere senso allora attribuire ad esse un peso maggiore per riequilibrare gli aspetti sottorappresentati. Un ultimo caso facilmente immaginabile è quello in cui si decida di attribuire un peso minore alle variabili che si ritengono rilevate con un grado di precisione non elevato: in questo caso la ponderazione potrebbe aiutare a "limitare i danni" insiti negli errori di misurazione.

Un esempio di distanza ponderata è la distanza di Minkowski ponderata:

$$d_{M,l}^W = \left[\sum_{j=1}^k |(x_{ij} - x_{i'j})|^l w_j \right]^{1/l} \quad (1.14)$$

Considerando l'ordine $l = 2$, si ottiene la distanza euclidea ponderata:

$$d_{E,ii'}^W = \left[\sum_{j=1}^k |(x_{ij} - x_{i'j})|^2 w_j \right]^{1/2} = (x_i^T - x_{i'}^T)^T W (x_i^T - x_{i'}^T) \quad (1.15)$$

Nella (1.15), la seconda espressione è l'alternativa in forma matriciale, in cui la matrice W è una matrice diagonale di dimensione $k \times k$ che raccoglie i pesi w_j .

$$W = \begin{bmatrix} w_1 & 0 & 0 & \dots & \dots & 0 \\ 0 & w_2 & 0 & \dots & \dots & 0 \\ 0 & 0 & \ddots & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & w_j & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & w_k \end{bmatrix} \quad (1.16)$$

1.2.2 La distanza di Mahalanobis

Fin qui gli indici di distanza e le distanze considerate non hanno mai preso in considerazione l'eventuale correlazione tra le variabili X_j . Tale informazione potrebbe però alterare la valutazione della diversità tra coppie di unità. Se si considerassero due variabili altamente correlate tra loro, l'informazione che fornisce una di esse risulterà ridondante rispetto all'altra e non aggiungerà informazioni addizionali circa il grado di diversità delle unità considerate. Al contrario, le informazioni fornite da due variabili poco o per nulla correlate tra loro non risulteranno ripetitive e permetteranno di rifinire la valutazione del grado di vicinanza o lontananza tra due unità. Una distanza che tiene proprio conto di tale aspetto è la *distanza di Mahalanobis*.

Definizione. Si definisce distanza di Mahalanobis la seguente espressione:

$$d_{M,ii'} = (x_i^T - x_{i'}^T)^T S^{-1} (x_i^T - x_{i'}^T) \quad (1.17)$$

Nella (1.17), S è una matrice di dimensione $k \times k$ in cui sono raccolte le varianze e le covarianze delle variabili X_j : in particolare, sulla diagonale si trovano le varianze e al di fuori di essa, in modo simmetrico, le covarianze tra coppie di variabili. Se si lavorasse con variabili standardizzate, la matrice di varianze e covarianze S verrebbe sostituita dalla matrice di correlazione R . In quest'ultima, la diagonale è composta da soli 1 (ossia la varianza di una variabile standardizzata) e le covarianze sono sostituite dalle correlazioni disposte sempre in modo simmetrico rispetto alla diagonale principale. L'aspetto forse più interessante della distanza di Mahalanobis è che la distanza tra due unità i e i' non dipenderà solamente dai valori che le variabili assumono in corrispondenza di esse, ma sarà influenzata anche dalle altre unità che concorrono a determinare la struttura della matrice S . Inoltre essa può essere interpretata come una distanza euclidea ponderata in cui la matrice dei pesi $W = S^{-1}$.

1.2.3 Indici di similarità

Tutte le distanze e gli indici di distanza fin qui trattati, sono applicabili nel caso in cui le variabili considerate sono di natura quantitativa. Esiste anche una classe di indici da poter utilizzare quando si lavora con variabili di tipo qualitativo.

Definizione: dato un insieme di unità $u_i \in U$, si definisce indice di similarità un'applicazione $S(u_i, u_j) = S_{ij}$ che soddisfi le proprietà di non negatività, di normalizzazione e di simmetria. In formule:

$$S_{ij} \geq 0 \quad \forall u_i, u_j \in U \quad (1.18)$$

$$S_{ii} = 1 \quad \forall u_i \in U \quad (1.19)$$

$$S_{ij} = S_{ji} \quad \forall u_i, u_j \in U \quad (1.20)$$

Si noti come un indice di similarità sia definito con riferimento a un insieme di unità statistiche e non ai valori che le variabili assumono su di esse, come nel caso delle distanze. In modo molto immediato, è possibile definire anche un indice di

dissimilarità, che costituisce il complemento a 1 di un indice di similarità. Un caso particolare di applicazione degli indici di similarità è quella in cui su n unità sono rilevati una serie di caratteri dicotomici. Solitamente un variabile dicotomica assume solamente i valori 1 e 0, per indicare rispettivamente la presenza o l'assenza di un particolare fenomeno. In tal caso è possibile definire delle quantità, su cui poi è basata la costruzione di alcuni indici. In particolare si indica con a il totale dei *positive matches*, ossia la frequenza dei fenomeni contemporaneamente presenti in due unità; si indica con d il totale dei *negative matches*, ossia la frequenza dei fenomeni assenti in entrambe le unità e, infine, con b e c si indicano le frequenze dei fenomeni presenti in una unità ma assenti nell'altra. Tra gli indici di similarità più utilizzati:

- Indice di Russel e Rao:

$$S_{ij} = \frac{a}{p} \quad (1.21)$$

in cui p rappresenta il totale dei fenomeni considerati.

- Indice di Jaccard:

$$S_{ij} = \frac{a}{a + b + c} \quad (1.22)$$

che si differenzia dal precedente, poiché a denominatore non vengono considerati i negative matches (d).

- Indice di Sokal e Michener:

$$S_{ij} = \frac{a + d}{p} \quad (1.23)$$

Gli indici presentati possono anche essere considerati dei casi particolari di un'espressione più generale:

$$S_{ij} = \frac{aw_1 + dw_2}{aw_3 + dw_4 + (b + c)w_5} \quad (1.24)$$

La (1.24) rappresenta una combinazione lineare delle frequenze a, b, c e d ponderate con dei pesi w_i . La scelta opportuna di tali pesi permette di ricondursi a una delle espressione degli indici di similarità sopra esposti.

Da questo momento in poi, tuttavia, ci si limiterà a considerare casi in cui le variabili X_j sono di sola natura quantitativa. A questo punto, la scelta di una distanza o di un indice di distanza tra quelli sopra esposti, permette di effettuare il passaggio dalla matrice dei dati $X_{n,k}$ alla matrice delle distanze $D_{n,n}$ in cui il generico elemento $d_{i,i'}$ indica il valore della distanza tra l'unità i e l'unità i' . Ovviamente la matrice delle distanze avrà dimensioni $n \times n$ poichè raccoglie il valore delle distanze tra ogni unità e tutte le altre; inoltre sarà una matrice simmetrica come conseguenza della proprietà di simmetria. In alcuni casi, si ha a disposizione solamente la matrice $D_{n,n}$ e non la matrice $X_{n,k}$, perciò è possibile definire diversi metodi di aggregazione delle unità basati sulla conoscenza di una o dell'altra matrice.

1.3 Analisi dei gruppi

L'analisi dei gruppi, o cluster analysis, si caratterizza come un metodo esplorativo: ha l'obiettivo di riconoscere gruppi all'interno delle n osservazioni, senza sapere a priori se tali gruppi esistano. Tale obiettivo differenzia il metodo dell'analisi dei gruppi da quello dell'analisi discriminante: quest'ultimo ha sempre l'obiettivo di classificare n unità statistiche in gruppi omogenei, sapendo però, che tali unità appartenevano a due o più popolazioni differenti. L'analisi discriminante ha perciò l'obiettivo di stabilire un criterio per assegnare in modo corretto ulteriori osservazioni alla popolazione di appartenenza. Da qui in poi ci si concentrerà sulla cluster analysis, la quale permette di effettuare una riduzione dello spazio delle unità in g gruppi omogenei con notevoli vantaggi in termini di parsimonia e interpretabilità.

Di fondamentale importanza è chiarire per quale preciso scopo si decide di effettuare una classificazione: tale scopo guiderà infatti poi tutte le scelte successive del ricercatore, dalla scelta delle variabili a quella della distanza da utilizzare, fino a quella del metodo di formazione dei gruppi. I principali metodi sono distinti in *gerarchici* e *non gerarchici*. Con i primi si ottiene una famiglia di partizioni: da quella banale in cui ogni unità costituisce un gruppo, fino all'altra, altrettanto banale, in cui tutte le unità sono raggruppate in un unico cluster. Con i metodi non gerarchici si perviene invece ad un'unica partizione in g gruppi, con g fissato a priori.

1.4 Metodi gerarchici

Nella seguente sezione si esamineranno nel dettaglio le caratteristiche dei metodi gerarchici. Questi ultimi non necessitano di fissare il numero dei gruppi a priori e ciò li rende alquanto flessibili. Tuttavia, il numero di tutte le partizioni possibili risulta altissimo anche con valori relativamente piccoli di n e g , perciò diventa impossibile pensare di controllarle tutte in modo esaustivo. Per questo motivo si fa ricorso ai metodi gerarchici *aggregativi* o *scissori*. Nei metodi gerarchici aggregativi si parte da $g = n$ gruppi per arrivare, tramite aggregazione, per l'appunto, a un unico gruppo. I metodi gerarchici scissori procedono invece in maniera esattamente simmetrica.

1.4.1 Gerarchia di partizioni e distanze tra gruppi

Il concetto di *gerarchia* risulta fondamentale per questa classe di metodi ed è legato al concetto di distanza.

Fissato un *livello di distanza* $\gamma \in [0, +\infty)$, si può stabilire se due unità i e i' sono simili tra loro oppure no: in particolare, se la distanza tra esse risulta maggiore di γ , allora le due unità sono distanti e saranno allocate in due gruppi distinti; al contrario se la stessa distanza risultasse minore di γ allora le due unità si possono considerare simili tra loro e verrano inserite nello stesso gruppo. Se si considera una gerarchia di partizioni, allora i gruppi ottenuti con un particolare livello di distanza γ_2 , conterranno i gruppi ottenuti con un livello γ_1 , con $\gamma_1 < \gamma_2$. In altre parole, ciò vuol dire che se due unità i e i' sono raggruppate nello stesso cluster con un livello γ_1 , lo saranno anche con un livello γ_2 superiore. Questo vincolo per cui se due unità si uniscono ad un certo livello della procedura di aggregazione non si possono più

separare, può risultare a volte troppo rigido e per questo si considerano anche i metodi non gerarchici.

Praticamente, la cluster analysis procede individuando nella matrice delle distanze $D_{n,n}$ le due unità più simili, che verranno unite in un unico gruppo. Successivamente si devono calcolare le nuove distanze tra gli $n - 1$ gruppi rimanenti. Risulta perciò necessario stabilire un criterio per calcolare una distanza non più tra singole unità, ma anche tra un gruppo e un'unità e, in generale, tra due gruppi.

È possibile scegliere tra vari criteri:

- Metodo del legame singolo:

$$d_{C_1, C_2} = \min_{r \in C_1, s \in C_2} d_{r,s} \quad (1.25)$$

Secondo la (1.25), la distanza tra due generici cluster C_1 e C_2 è definita come la minima distanza che intercorre tra un'unità r del primo cluster e un'unità s del secondo.

- Metodo del legame completo:

$$d_{C_1, C_2} = \max_{r \in C_1, s \in C_2} d_{r,s} \quad (1.26)$$

In questo caso la distanza tra due gruppi risulta uguale alla massima distanza che intercorre tra due unità appartenenti ai due cluster in questione.

- Metodo del legame medio:

$$d_{C_1, C_2} = \frac{1}{n_1 n_2} \sum_{r \in C_1} \sum_{s \in C_2} d_{r,s} \quad (1.27)$$

Poichè considerare solamente la distanza minima o la distanza massima potrebbe risultare riduttivo, con la (1.27) si calcola una media tra tutte le distanze tra le unità del primo e del secondo cluster. Con n_1 e n_2 si indicano rispettivamente le numerosità dei due gruppi.

- Metodo del centroide:

$$d_{C_1, C_2} = d(\bar{x}_1, \bar{x}_2) \quad (1.28)$$

Secondo la (1.28), la distanza tra due gruppi è pari alla distanza tra i loro centroidi. In particolare, si definisce centroide di un gruppo il vettore che raccoglie le medie delle k variabili calcolate solamente sulle unità assegnate al gruppo in considerazione. Per esempio, $\bar{x}_c = [\bar{x}_{c1} \ \bar{x}_{c2} \ \dots \ \bar{x}_{ck}]$ con il generico $\bar{x}_{ck} = \frac{1}{n_c} \sum_{r \in C_c} x_{rk}$.

- Metodo di Ward:

$$d_{C_1, C_2} = \frac{n_1 n_2}{n_1 + n_2} \|\bar{x}_1 - \bar{x}_2\|^2 \quad (1.29)$$

In questo caso, la distanza tra due gruppi è definita come la distanza euclidea al quadrato tra i due centroidi, moltiplicata per una quantità che dipende dalla numerosità dei cluster. Il metodo di Ward ha una relazione con la *devianza within*, ossia la devianza interna, dei gruppi: con questo metodo si vanno

ad unire due gruppi in modo che l'incremento della devianza within W sia minimo. È intuitivo il motivo per cui tale proprietà sia desirabile: infatti, una devianza within minima implica avere unità molto simili tra loro all'interno di un gruppo.

Con uno di questi metodi è possibile ora scrivere la matrice $D_{n-1,n-1}$, all'interno della quale si andranno a identificare nuovamente i gruppi con distanza minima per unirli insieme. Si itererà tale procedura fino ad arrivare a definire $D_{2,2}$, ossia quando sarà rimasta un'unica aggregazione possibile tra due gruppi per formarne uno unico. Le varie partizioni ottenute ad ogni step, costituiscono la famiglia di partizioni restituita dalla procedura. Si può notare come i metodi del legame singolo, del legame completo e del legame medio necessitino solamente della conoscenza della matrice delle distanze. È inoltre interessante come ognuno di tali metodi dia luogo a delle forme peculiari dei cluster: il metodo del legame completo tende a determinare cluster di forma sferica; il metodo del legame singolo identifica cluster di forma allungata e va incontro al cosiddetto "effetto a catena", per il quale si verifica la tendenza ad accorpare ogni nuova unità al cluster precedentemente costituito; infine, il metodo del legame medio rappresenta una via di mezzo tra i metodi precedenti. Per quanto riguarda i metodi del centroide e di Ward, il loro utilizzo è possibile solo avendo a disposizione la matrice dei dati per poter calcolare i centroidi.

Formula di Lance-Williams

Gli algoritmi appena descritti possono essere riassunti in una formula unitaria, inizialmente proposta da Lance and Williams (1967). Per la sua illustrazione, si considerino 3 cluster C_1, C_2 e C_3 e si indichi con $d[C_3, (C_1, C_2)]$ la distanza tra il cluster 3 e il cluster risultante dalla fusione degli altri due. Si ipotizzi inoltre di conoscere le distanze reciproche tra i singoli cluster d_{12}, d_{31} e d_{32} . La formula di Lance-Williams si presenta come segue:

$$d[C_3, (C_1, C_2)] = \alpha_1 d_{31} + \alpha_2 d_{32} + \beta d_{12} + \gamma |d_{31} - d_{32}| \quad (1.30)$$

in cui α_1 , α_2 , β e γ sono dei parametri fissati dal ricercatore. Si può facilmente vedere come il metodo del legame singolo sia ottenibile scegliendo per i parametri i valori: $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, $\beta = 0$ e $\gamma = -0.5$. Infatti, la (1.30) diventa:

$$d[C_3, (C_1, C_2)] = 0.5(d_{31} + d_{32}) - 0.5|d_{31} - d_{32}|$$

A questo punto se $d_{31} < d_{32}$, si ottiene:

$$d[C_3, (C_1, C_2)] = 0.5(d_{31} + d_{32}) - 0.5d_{32} + 0.5d_{31} = d_{31}$$

Con passaggi analoghi, risulta immediato verificare che se fosse $d_{31} > d_{32}$, allora si avrebbe che:

$$d[C_3, (C_1, C_2)] = 0.5(d_{31} + d_{32}) + 0.5d_{32} - 0.5d_{31} = d_{32}$$

Perciò con la scelta di questi parametri la formula di Lance-Williams aggiorna la matrice delle distanze considerando:

$$d[C_3, (C_1, C_2)] = \min(d_{31}, d_{32})$$

Effettuando una opportuna scelta dei parametri, oltre al metodo del legame singolo è possibile ottenere anche gli altri metodi esposti nella sezione 1.4.

1.4.2 Il dendrogramma

La famiglia di partizioni che si ottiene con un metodo gerarchico può essere formalizzata introducendo il *dendrogramma*.

Definizione: si definisce dendrogramma una applicazione $D(\gamma) : \mathbb{R}^+ \rightarrow \pi(p)$, in cui γ è il livello di distanza e $\pi(p)$ è l'insieme di tutte le partizioni.

Tale applicazione presenta alcune caratteristiche ben precise. In particolare: $D(0)$, ossia l'applicazione dendrogramma in corrispondenza di un livello di distanza nullo, restituisce la partizione banale in $g = n$ gruppi ognuno costituito da una singola unità; esisterà un particolare livello di distanza h per cui $D(h)$ restituisce la partizione in un solo gruppo e per cui tale caratteristica continua ad essere vera per un qualsiasi livello f con $f > h$; infine, il dendrogramma è una funzione a scalini, ossia $D(h + \epsilon) = D(h)$ per ϵ sufficientemente piccolo. Avendo introdotto il dendrogramma, è possibile fornire una nuova interpretazione del concetto di gerarchia: dati due livelli di distanza h e f (con $h < f$), la partizione in corrispondenza di h è uguale o più fine, ossia con più gruppi, rispetto a quella ottenuta in corrispondenza di f . Il dendrogramma può essere visualizzato con un grafico che ricorda un albero stilizzato, come nella figura seguente:

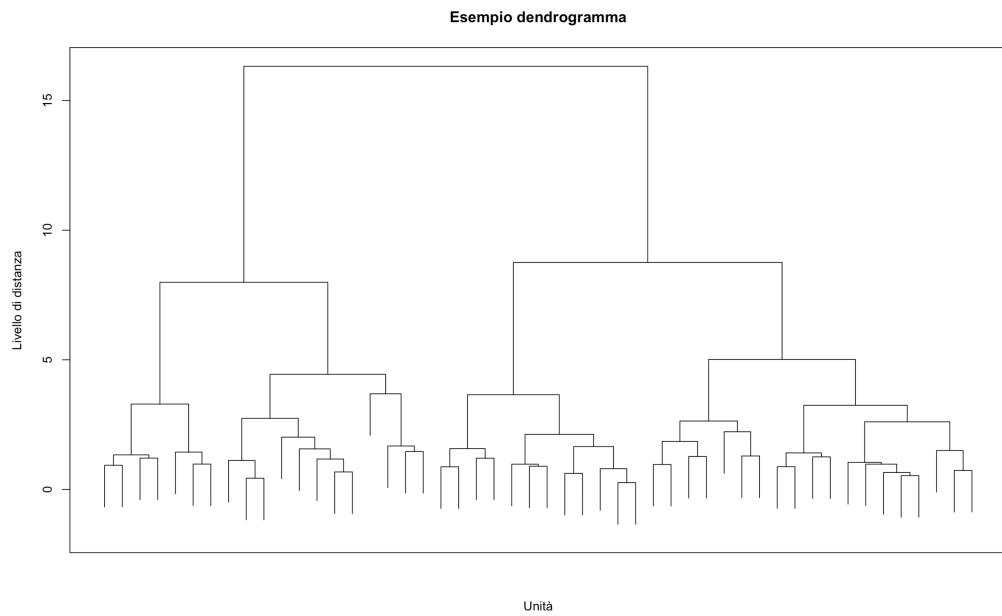


Figura 1.1. Esempio di dendrogramma

La figura 1.1 rappresenta un dendrogramma ottenuto con un metodo gerarchico aggregativo applicato al dataset **USArrest** disponibile su R. In particolare è stato utilizzato il metodo di Ward per definire le distanze tra cluster. Sull'asse delle y è riportato il livello di distanza e per visualizzare la partizione ottenuta ad un particolare livello di γ , bisogna immaginare di tagliare con una linea orizzontale

il dendrogramma in corrispondenza di tale livello. Il dendrogramma rappresenta sicuramente un metodo grafico efficace per rappresentare in modo intuitivo i risultati di una procedura aggregativa, ma fornisce anche un criterio per la scelta della partizione più opportuna: quando il "salto" tra uno step e il successivo della procedura risulta grande, vuol dire che in quel punto sono stati uniti gruppi tra loro abbastanza diversi e, per questo motivo, potrebbe risultare opportuno scegliere la partizione tagliando il grafico in corrispondenza di tale salto.

1.4.3 Caratteristiche delle partizioni

Un altro criterio per determinare il numero ottimale di gruppi è quello di scegliere la partizione per cui la massima distanza tra due unità nello stesso gruppo, risulti minore della minima distanza che intercorre tra cluster distinti. Una partizione si definisce *ben strutturata*, se si verifica la seguente condizione:

$$\max d_{p,q} \leq \min d_{r,s} \quad (1.31)$$

Nella (1.31), gli indici p e q indicano due unità appartenenti allo stesso cluster, mentre con r e s si indicano due unità appartenenti a cluster differenti. Quando la condizione di partizione ben strutturata è raggiunta in corrispondenza del minimo numero g di gruppi, allora si ha una partizione ben strutturata e *minimale*. La ricerca di una partizione con queste caratteristiche potrebbe rappresentare un criterio applicativo per la scelta del numero dei gruppi; tuttavia spesso la partizione ben strutturata e minimale si ottiene per un numero elevato di gruppi, ponendo un limite pratico a tale strada.

Si è visto come le procedure di formazione dei gruppi dipendano dalla scelta di una distanza e di un criterio di definizione di distanza tra gruppi. È interessante perciò chiedersi se la partizione individuata resti invariata a seguito di trasformazioni monotone crescenti delle distanze. Tale proprietà di invarianza si realizza con i metodi del legame singolo e del legame completo, poiché trasformazioni monotone delle distanze non vanno ad alterare l'ordinamento identificato dagli operatori di massimo e minimo. Al contrario il metodo del legame medio non risulta invariante poiché l'operatore media aritmetica può subire delle alterazioni in seguito a trasformazioni monotone delle distanze.

1.4.4 Scomposizione della devianza

Un algoritmo di tipo gerarchico conduce alla identificazione di una famiglia di partizioni. Tuttavia non tutte le partizioni considerate costituiscono una buona rappresentazione della realtà. Al fine di introdurre un criterio per valutare la bontà di una partizione, è necessario introdurre il concetto di devianza e vedere come essa possa essere vista come la somma di due componenti. La devianza totale è un indice di variabilità che considera la somma degli scarti al quadrato delle singole osservazioni dalla media generale: valori bassi per la devianza indicano una concentrazione delle osservazioni attorno alla media, mentre valori alti indicano una maggiore dispersione.

Si definisce la devianza totale come:

$$T = \sum_{j=1}^k \sum_{i=1}^n (x_{ij} - \bar{x}_j)^2 \quad (1.32)$$

Si può dimostrare che la devianza totale T può essere scomposta nella somma $T = W + B$, in cui W indica la *devianza within*, ossia all'interno dei gruppi, e B indica la *devianza between*, ossia tra i gruppi. Si definisce la devianza within come:

$$W = \sum_{c=1}^g W_c = \sum_{c=1}^g \sum_{j=1}^k \sum_{i \in C_c} (x_{ij} - \bar{x}_{cj})^2 \quad (1.33)$$

in cui c è un indice che identifica i cluster e \bar{x}_{cj} indica il centroide del gruppo c . Infine, si definisce la devianza between come:

$$B = \sum_{j=1}^k \sum_{c=1}^g (\bar{x}_{cj} - \bar{x}_j)^2 n_c \quad (1.34)$$

in cui n_c indica la numerosità del cluster c con $\sum_{c=1}^g n_c = n$.

L'obiettivo di identificare cluster omogenei al loro interno si traduce nella desiderabilità di valori contenuti della devianza W . Sotto il vincolo della decomposizione della devianza, minimizzare W equivale a massimizzare la devianza between. Si può quindi pensare di basare la scelta del numero di gruppi g su una valutazione della devianza W delle varie partizioni possibili. È intuitivo che si avrà devianza W nulla in corrispondenza della partizione banale in $g = n$ gruppi ognuno costituito da una sola unità e che la stessa tenderà ad aumentare riducendo il numero di gruppi. Esiste, in altre parole, un *trade-off* tra il numero di gruppi e devianza within, ossia tra parsimonia e omogeneità. È possibile anche considerare un indice basato sulle quantità appena definite:

$$R^2 = \frac{B}{T} = 1 - \frac{W}{T} \quad (1.35)$$

in cui la seconda espressione è, ancora una volta, una conseguenza della decomposizione della devianza. Dalla (1.35) si può vedere come R^2 assuma valori tra $[0, 1]$ e come tenda ad assumere valori più alti al diminuire del valore della devianza W . In altre parole, si avranno valori più alti di R^2 all'aumentare del numero di gruppi g . Il fatto che R^2 vari nell'intervallo $[0, 1]$ permette di utilizzare tale indice per confrontare partizioni con numero diverso di gruppi o magari ottenute con metodi differenti.

1.5 Metodi non gerarchici

Nella seguente sezione si passerà ad esaminare le caratteristiche dei metodi non gerarchici di classificazione. Come già sottolineato, una prima differenza con i metodi gerarchici, consiste nella necessità di fissare il numero di gruppi desiderato a priori. Per tale motivo, questa classe di metodi restituisce come risultato un'unica partizione e non più una famiglia. Per valutare la bontà delle partizioni ottenute, risulta utile

eseguire più volte la procedura non gerarchica con diversi valori di g , oppure, se n non è troppo elevato, far precedere l'analisi da una procedura gerarchica per avere un'idea preliminare sul numero dei gruppi. Anche in questo caso, esaminare esaustivamente tutte le partizioni possibili risulta non sostenibile e si farà nuovamente ricorso a degli algoritmi.

1.5.1 Il metodo delle k -medie

Il metodo non gerarchico più utilizzato è il metodo delle k -medie. L'obiettivo nella ricerca della partizione ottimale consiste nella minimizzazione della devianza W . L'espressione definita nella (1.33) può essere riscritta in modo alternativo introducendo due nuove matrici:

- La matrice dei centrodi \bar{X}

In questa matrice ogni riga è rappresentata dal centroide, che si ricorda essere il vettore delle medie, di un cluster. Perciò la matrice avrà dimensioni $g \times k$.

In forma matriciale:

$$\bar{X}_{gxk} = \begin{bmatrix} \bar{x}_1^T \\ \vdots \\ \bar{x}_c^T \\ \vdots \\ \bar{x}_g^T \end{bmatrix} \quad (1.36)$$

- La matrice di allocazione U

La matrice si configura come segue:

$$U_{nxg} = \begin{bmatrix} u_{11} & \dots & u_{1c} & \dots & u_{1g} \\ \vdots & & \vdots & & \vdots \\ u_{i1} & \dots & u_{ic} & \dots & u_{ig} \\ \vdots & & \vdots & & \vdots \\ u_{n1} & \dots & u_{nc} & \dots & u_{ng} \end{bmatrix} \quad (1.37)$$

Come si può vedere la matrice ha tante righe quante sono le unità e tante colonne quante sono i gruppi: perciò il generico u_{ic} farà riferimento all'unità i e al gruppo c . L'obiettivo di tale matrice è fornire informazioni sull'appartenenza delle unità ai vari gruppi; di conseguenza u_{ic} assumerà valori:

$$u_{ic} = \begin{cases} 1 & i \in C_c \\ 0 & i \notin C_c \end{cases} \quad (1.38)$$

Ricordando inoltre la proprietà delle partizioni (1.2), si avrà:

$$\sum_{c=1}^g u_{ic} = 1 \quad (1.39)$$

ossia che la somma per riga della matrice U è sempre pari a 1, poiché una stessa unità non può appartenere a più cluster.

A questo punto, con dei semplici passaggi, è possibile riscrivere l'espressione della devianza within come segue:

$$W = \sum_{c=1}^g \sum_{j=1}^k \sum_{i \in C_c} (x_{ij} - \bar{x}_{cj})^2 = \quad (1.40)$$

$$= \sum_{c=1}^g \sum_{i \in C_c} \underbrace{\sum_{j=1}^k (x_{ij} - \bar{x}_{cj})^2}_{\text{ }} = \quad (1.41)$$

$$= \sum_{c=1}^g \sum_{i \in C_c} d_E^2(x_i^T, \bar{x}_c^T) = \quad (1.42)$$

$$= \sum_{c=1}^g \sum_{i=1}^n u_{ic} d_E^2(x_i^T, \bar{x}_c^T) \quad (1.43)$$

In particolare, la seconda espressione si ottiene effettuando uno scambio tra le sommatorie di indice i e j ; la terza espressione si scrive riconoscendo nella precedente l'espressione di una distanza euclidea al quadrato e, infine, l'ultima espressione si ottiene estendendo la sommatoria con indice i e introducendo gli elementi u_{ic} precedentemente descritti.

L'obiettivo dell'algoritmo delle k -medie può quindi essere formalizzato come un problema di minimo vincolato. In particolare:

$$\begin{cases} \min_{U, \bar{X}} \sum_{c=1}^g \sum_{i=1}^n u_{ic} d_E^2(x_i^T, \bar{x}_c^T) \\ u_{ic} \in \{0, 1\} \quad \forall i = 1, \dots, n \quad c = 1, \dots, g \\ \sum_{c=1}^g u_{ic} = 1 \quad \forall i = 1, \dots, n \end{cases} \quad (1.44)$$

in cui la prima espressione rappresenta la funzione obiettivo da minimizzare rispetto alle matrici U e \bar{X} e le altre due i vincoli posti sulla matrice di allocazione.

Soluzione iterativa

Il problema di minimo vincolato (1.44) può essere risolto in maniera iterativa seguendo i seguenti step:

- Step 0 (*inizializzazione*)

In questa fase si costruisce la matrice $\bar{X}^{(t)}$ con $t = 0$, scegliendo in modo casuale i centroidi dei gruppi. Tale scelta può essere fatta estraendo numeri casuali da una normale oppure scegliendo delle unità come rappresentanti dei centroidi.

- Step 1 (*minimizzazione rispetto a U*)

Tenendo fissa la matrice $\bar{X}^{(t)}$ dello step precedente, si minimizza la funzione obiettivo solamente rispetto alla matrice U . Tale problema di minimo vincolato risulta di semplice soluzione una volta notato che la funzione obiettivo, per la generica unità i , può essere riscritta come segue:

$$u_{i1} d_E^2(x_i^T, \bar{x}_1^T) + \dots + u_{ic} d_E^2(x_i^T, \bar{x}_c^T) + \dots + u_{ig} d_E^2(x_i^T, \bar{x}_g^T) \quad (1.45)$$

In altre parole è possibile considerare g addendi, uno per ogni gruppo, per ogni unità i . Tenendo a mente i vincoli, si ha che, per ogni serie di g addendi, solamente un elemento u_{ic} può assumere il valore 1. Per minimizzare la funzione obiettivo sarà allora sufficiente assegnare il valore 1 all'elemento u_{ic} associato alla distanza euclidea al quadrato tra la riga della matrice dei dati e la riga della matrice dei centroidi più piccola. Procedendo in questo modo si costruisce la matrice di allocazione $U^{(t+1)}$.

- Step 2 (*minimizzazione rispetto a \bar{X}*)

Simmetricamente rispetto allo step 1, si tiene ora fissa la matrice $U^{(t+1)}$ e si minimizza la funzione obiettivo solamente rispetto alla matrice dei centroidi \bar{X} .

$$\min_{\bar{X}} \sum_{c=1}^g \sum_{i=1}^n u_{ic}^{(t+1)} d_E^2(x_i^T, \bar{x}_c^T) \quad (1.46)$$

Ponendo l'attenzione sul generico cluster c , la (1.46) diventa:

$$\min_{\bar{X}_c} \sum_{i=1}^n u_{ic}^{(t+1)} d_E^2(x_i^T, \bar{x}_c^T) = \sum_{i \in C_c^{(t+1)}} d_E^2(x_i^T, \bar{x}_c^T) \quad (1.47)$$

Minimizzare la (1.47) significa minimizzare la somma degli scarti dai centroidi al quadrato e tale obiettivo si raggiunge considerando la media campionaria calcolata per ogni variabile sulle unità assegnate al cluster. In formule:

$$\bar{x}_{cj} = \frac{\sum_{i \in C_c^{(t+1)}} x_{ij}}{n_c} \quad \forall j = 1, \dots, k \quad (1.48)$$

Operativamente, si devono ricalcolare i centroidi, ossia le medie di gruppo per ogni variabile X_j , considerando i gruppi determinati dalla matrice $U^{(t+1)}$ dello step 1.

- Step 3 (*convergenza*)

In questo step si effettua un confronto tra la matrice dei centroidi inizializzata allo step 0 e quella aggiornata nello step 2. Tale operazione viene attuata calcolando la norma al quadrato della differenza delle due matrici e confrontandola con un ϵ scelto piccolo:

$$\|\bar{X}^{(t)} - \bar{X}^{(t+1)}\| < \epsilon \quad (1.49)$$

In particolare se la (1.49) è soddisfatta significa che le modifiche effettuate sono trascurabili, perciò l'algoritmo giunge a termine restituendo la partizione ottimale individuata dalla matrice $U^{(t+1)}$; in caso contrario si ritorna allo step 0 e si itera l'algoritmo fino a convergenza.

L'algoritmo delle k -medie risulta particolarmente veloce in quanto lavora con un solo valore di g prefissato e non necessita del calcolo delle matrici delle distanze. Per tale motivo, è ampiamente utilizzato in situazioni in cui il numero di unità è particolarmente elevato, ma anche quando esso risulta dell'ordine di qualche centinaio, poiché il dendrogramma associato ad un metodo gerarchico risulterebbe di difficile interpretazione.

1.6 Confronto di partizioni

Quando si effettua una cluster analysis, un obiettivo pratico è quello di verificare se esista o meno consenso tra le partizioni ottenute con metodi differenti o considerando un numero diverso di gruppi. Se si considera una partizione P_1 in g_1 gruppi e una partizione P_2 in g_2 gruppi (senza che sia necessariamente $g_1 = g_2$), si può dare una risposta alla domanda su quanto tali partizioni siano simili calcolando il *Rand Index* (RI).

$$RI = \frac{a + b}{a + b + c + d} \quad (1.50)$$

Nella (1.50), le quantità a, b, c e d rappresentano:

- a : numero di coppie di unità assegnate allo stesso cluster sia in P_1 che in P_2
- b : numero di coppie di unità assegnate a cluster differenti sia in P_1 che in P_2
- c : numero di coppie di unità assegnate allo stesso cluster in P_1 ma non in P_2
- d : numero di coppie di unità assegnate allo stesso cluster in P_2 ma non in P_1

Di fatto, a e b vedono ciò che le due partizioni hanno in comune, mentre c e d colgono le differenze. Per come è costruito, $RI \in [0, 1]$ e più si avvicina a 1 più le due partizioni sono simili tra loro. Ovviamente sarà possibile avere $RI = 1$ solamente se le partizioni hanno lo stesso numero di gruppi. Nell'applicazione pratica non si utilizza quasi mai il Rand Index poiché non raggiunge mai il suo minimo: per questo tende a restituire valori relativamente elevati anche per partizioni molto diverse. Si introduce l'*Adjusted Rand Index* (ARI), così costruito:

$$ARI = \frac{RI - E(RI)}{1 - ERI} \quad (1.51)$$

La (1.51) potrebbe assumere anche valori negativi, ma mantiene il suo massimo in 1. Come per la (1.50), valori elevati dell'indice continuano ad indicare concordanza tra le due partizioni considerate

Altri metodi di classificazione

I metodi di classificazione gerarchici e non gerarchici conducono alla determinazione di una partizione o di una gerarchia di partizioni, nelle quali ogni unità può appartenere ad uno e a un solo gruppo. Nei casi in cui è sensato supporre che una stessa unità possa appartenere a più gruppi, è possibile far ricorso a tecniche di classificazione non basate su partizioni.

- I *metodi di clumping* generano delle classificazioni in gruppi non disgiunti. Tuttavia l'appartenenza di un'unità a più gruppi potrebbe complicare notevolmente l'interpretazione dei risultati ottenuti, perciò potrebbe risultare necessario introdurre dei vincoli per limitare le sovrapposizioni possibili.
- Un *insieme sfocato* (fuzzy set) definisce l'appartenenza di un'unità a un gruppo attraverso un funzione che assume valori nell'intervallo $[0, 1]$. Si ha un valore

vicino a 1 quando l'unità appartiene al gruppo, vicino a 0 in caso contrario, mentre i valori intermedi indicano un'appartenenza parziale. Spostando l'attenzione su coppie di unità, esse si considerano più simili quanto più la funzione di appartenenza a uno stesso gruppo si avvicina a 1.

- Le *classificazioni vincolate* riducono l'insieme delle partizioni possibili, considerando ulteriori vincoli scelti dal ricercatore. Molto spesso nell'analisi di dati territoriali, tale vincolo si configura come un vincolo di contiguità spaziale.

Capitolo 2

Attraction-repulsion clustering

2.1 Introduzione

L'Intelligenza Artificiale è un argomento di cui si discute sempre più. Con questa espressione, spesso abbreviata nella sigla "IA", si indica una disciplina che studia se e in che modo sia possibile realizzare degli algoritmi in grado di riprodurre il comportamento e i processi decisionali umani. Con il tempo tale disciplina è diventata sempre più invasiva e viene utilizzata in numerosissimi campi: basti pensare alle auto con guida autonoma, al monitoraggio della finanza o agli algoritmi dei social media. Nonostante la rivoluzione dell'Intelligenza Artificiale abbia portato con sé innumerevoli vantaggi, è innegabile che il tempo stia evidenziando anche i limiti e i rischi a cui si va incontro affidandosi ad algoritmi e dati, senza affiancare ad essi una supervisione umana. Esemplificativo per il rischio appena esposto, è l'utilizzo dei cosiddetti *risk assessment tools*, ossia degli algoritmi utilizzati nei processi giudiziari negli Stati Uniti che dovrebbero essere in grado di valutare la probabilità che un imputato abbia di compiere un nuovo delitto ([4]).

Gli algoritmi spesso mancano di equità e trasparenza, tanto che Cristianini, nell'articolo *Shortcuts to artificial intelligence* [2], sostiene che gli effetti negativi dell'Intelligenza Artificiale siano dovuti a un *debito etico*. Un campo di ricerca noto come *fair learning* sta cercando di ripagare parte di questo debito, studiandone le cause e cercando di mitigare gli effetti di distorsioni e mancanza di equità che caratterizzano le decisioni automatiche. In particolare si vuole essere certi che le regole di decisione degli algoritmi, basate sulle variabili esplicative X , non siano influenzate da una sottocategoria di variabili definite *sensibili* o *protette*. Esempi di tali variabili potrebbero essere il sesso, l'etnia o l'età. Poiché tali variabili sono spesso connesse ad altre, non possono considerarsi indipendenti e semplicemente ignorarle non può costituire una soluzione accettabile.

Le procedure classiche di cluster analysis esposte nel capitolo [1] non tengono conto del concetto di equità: l'obiettivo è solo quello di determinare gruppi con massima omogeneità interna. Tuttavia, con riferimento alle variabili definite sensibili, l'omogeneità spesso non è una caratteristica desiderabile. Si potrebbe immaginare una situazione in cui l'obiettivo è la formazione delle classi prime di un liceo, avendo a disposizione informazioni circa l'età, il comune di provenienza, il livello di competenza raggiunto e la necessità di Bisogni Educativi Speciali (BES). In uno scenario del genere, mentre

l'omogeneità per quanto riguarda il comune di provenienza può essere desiderabile per avere vantaggi nella stesura dell'orario delle lezioni e nell'organizzazione logistica dei mezzi di trasporto, la stessa omogeneità per il livello di competenza e la presenza di BES non solo non è desiderabile ma potrebbe avere effetti negativi sia per il corpo docente, sia per gli studenti stessi. L'idea è allora quella di stabilire a priori una definizione matematica di equità, per poi introdurre delle perturbazioni nelle misure di dissimilarità sulle quale si basa la procedura.

2.2 Fair clustering

Il contesto appena delineato fa capire come ogni procedura di *fair clustering* debba iniziare con la scelta di una definizione di equità che venga poi tradotta in termini matematici. Esistono più definizioni possibili di equità e relative traduzioni: non esiste un metodo preferibile in assoluto, ma la scelta dipenderà di volta in volta dal particolare problema in analisi. Poichè le classiche procedure di cluster analysis non tengono in considerazione il concetto di equità, di conseguenza, se le variabili considerate dovessero essere influenzate dalle variabili sensibili, anche i gruppi risultanti saranno distorti in tal senso. Una proxy del concetto di equità, molto spesso utilizzata, è la parità demografica. Nell'ambito della cluster analysis, tale idea si traduce in termini matematici nel concetto di *cluster equilibrati*, nel senso che ogni cluster dovrebbe contenere sottogruppi nella stessa proporzione con cui essi si trovano nell'intera popolazione: ciò implica che ogni cluster singolarmente considerato dovrebbe essere rappresentativo della popolazione sotto analisi. Con riguardo all'insieme degli attributi protetti, tale idea si concretizza nella situazione ideale in cui la proporzione di tali attributi è la stessa in ogni cluster e pari a quella dell'intero dataset. Tornando all'esempio della formazione delle classi, se si avesse per esempio il 20% di ragazzi appartenenti alla fascia più alta del livello di competenza, l'ideale sarebbe determinare classi che presentino tutte il 20% di studenti con tale caratteristica. Potrebbe sorgere il dubbio se sia sempre appropriato identificare il concetto di equità con quello di equilibrio; tuttavia la diversità all'interno della composizione dei cluster è sicuramente una caratteristica desiderabile: per questo motivo da ora in poi si concentrerà l'attenzione su procedure di clustering che favoriscono la diversità (*diversity enhancing clustering*). Queste ultime possono essere viste come una classe più ampia delle procedure di *diversity preserving clustering*, ossia delle procedure di clustering che conservano la diversità.

Supponiamo che nella matrice dei dati $X_{n,k}$, le k variabili di interesse siano distinguibili in p variabili sensibili o protette S e $d = k - p$ variabili non protette X , per cui $(X, S) \in \mathbb{R}^d \times \mathbb{R}^p$. La coppia (X_i, S_i) indica l'insieme delle k variabili in riferimento alla generica unità i e con DS si fa riferimento all'intero dataset $DS = \{(X_1, S_1), \dots, (X_n, S_n)\}$. Allora, una partizione che conserva la diversità rispetta la seguente condizione: la proporzione di individui in ogni cluster per ogni modalità assumibile dagli attributi protetti è la stessa della corrispondente proporzione sul totale dei dati. In formule:

$$\frac{|\{i \in C_c : S_i = m\}|}{|C_c|} = \frac{|\{i \in DS : S_i = m\}|}{n} \quad \forall c = 1, \dots, g \quad \forall m \in S \quad (2.1)$$

in cui C_c indica un generico cluster e m una delle modalità assumibili dagli attributi S .

Una diversity enhancing clustering, ha l'obiettivo di incrementare la diversità rispetto a una procedura classica, senza però imporre la condizione (2.1). Per raggiungere tale obiettivo si considera una nuova classe di indici di dissimilarità, che favoriscono l'eterogeneità rispetto agli attributi sensibili. Tali indici, proposti dagli autori dell'articolo *"Attraction-repulsion clustering: a way of promoting diversity linked to demographic parity in fair clustering"* [3], sono ispirati al principio di attrazione e repulsione dell'elettromagnetismo: unità che presentano lo stesso valore per gli attributi S tenderanno a respingersi, mentre unità che presentano valori diversi tenderanno ad attrarsi.

2.3 Nuove dissimilarità e interpretazione dei parametri

Si è già mostrato come la scelta degli indici di dissimilarità sia il primo passo fondamentale nell'ambito dell'analisi dei gruppi: essi si basano sui dati contenuti nella matrice $X_{n,k}$ e stabiliscono quanto due unità siano vicine tra loro o meno, per poi costituire in modo iterativo i cluster. Per favorire la diversità si considera una serie di dissimilarità nello spazio $(X, S) \in \mathbb{R}^d \times \mathbb{R}^p$ che tendono ad allontanare le unità che presentano lo stesso valore degli attributi protetti. È proprio come se l'etichetta S giocasse il ruolo di una carica elettrica: unità con la stessa carica si respingono tra loro, mentre unità con cariche diverse si attraggono. Le dissimilarità scelte sono tali per cui inducono diversità in procedure di clustering successive, rispettano la geometria dei dati originali e risultano di facile interpretazione.

Definizione Si definiscono dissimilarità di attrazione e repulsione tra coppie di unità le seguenti:

$$\delta_1((X_i, S_i), (X_{i'}, S_{i'})) = 1^T U 1 + S_i^T V S_{i'} + \|X_i - X_{i'}\|^2 \quad (2.2)$$

in cui U e V sono matrici simmetriche di dimensioni $p \times p$.

$$\delta_2((X_i, S_i), (X_{i'}, S_{i'})) = \left(1 + ue^{-v\|S_i - S_{i'}\|^2}\right) \|X_i - X_{i'}\|^2 \quad (2.3)$$

in cui $u, v \geq 0$.

$$\delta_3((X_i, S_i), (X_{i'}, S_{i'})) = \|X_i - X_{i'}\|^2 - u\|S_i - S_{i'}\|^2 \quad (2.4)$$

in cui $u \geq 0$.

$$\delta_4((X_i, S_i), (X_{i'}, S_{i'})) = \left(1 + \text{sgn}(S_i^T V S_{i'})u\left(1 - e^{-v(S_i^T V S_{i'})^2}\right)e^{-w\|X_i - X_{i'}\|}\right) \|X_i - X_{i'}\| \quad (2.5)$$

in cui $0 \leq u \leq 1$ e $v, w \geq 0$.

Analizzando queste nuove dissimilarità nel dettaglio si può vedere come la (2.2) sia una perturbazione additiva della distanza euclidea al quadrato: l'intensità della perturbazione è controllata dalle matrici U e V (in particolare con V si controllano le interazioni tra gli stessi e tra diversi valori degli attributi S). Anche la (2.4) è una perturbazione additiva, in cui però la penalizzazione è proporzionale alla

differenza tra gli attributi protetti, con intensità controllata dal parametro u . La dissimilarità (2.3) è, invece, una perturbazione moltiplicativa della distanza euclidea al quadrato: in particolare con il parametro u si controlla la massima perturbazione raggiungibile, mentre con il parametro v si controlla la velocità con cui si diverge da tale massimo quando i valori degli attributi S sono diversi. Infine, anche la (2.5) è una perturbazione moltiplicativa, questa volta della distanza euclidea. La particolarità di tale dissimilarità consiste nell'essere locale: interessa in misura minore punti che sono lontani tra loro e in misura maggiore punti vicini. Il parametro w è il parametro di località e più assume valori alti più la perturbazione risulta significativa per punti vicini tra loro. Come nelle precedenti, la matrice V controlla l'interazione tra gli attributi protetti, u stabilisce la massima perturbazione raggiungibile e v determina la velocità con cui si diverge da tale massimo se le due unità presentano valori diversi per gli attributi protetti.

Per meglio capire il funzionamento delle dissimilarità appena esposte, si considera un caso semplificato in cui si ha una popolazione con una minoranza identificata con $S = -1$ e una maggioranza indicata con $S = 1$: ossia, si ha un solo attributo protetto e binario. Poiché $p = 1$, le matrici U e V avranno entrambe dimensioni 1×1 , ossia saranno degli scalari. Fissando per comodità $U = V = c \geq 0$, la (2.2) diventa allora:

$$\delta_1((X_i, S_i), (X_{i'}, S_{i'})) = c(1 + S_i S_{i'}) + \|X_i - X_{i'}\|^2 \quad (2.6)$$

È immediato verificare che se $S_i \neq S_{i'}$ si ottiene la familiare distanza euclidea al quadrato $\|X_i - X_{i'}\|^2$; mentre se $S_i = S_{i'}$, il primo termine della 2.6 è positivo e quindi si è di fatto introdotto un effetto di repulsione tra unità che presentano lo stesso valore dell'unico attributo protetto S .

Scegliendo invece $u = 0.1$ e $v = 100$, la (2.3) diventa:

$$\delta_2((X_i, S_i), (X_{i'}, S_{i'})) = \left(1 + 0.1e^{-100\|S_i - S_{i'}\|^2}\right) \|X_i - X_{i'}\|^2 \quad (2.7)$$

Dalla 2.7 si può facilmente vedere che se $S_i \neq S_{i'}$, il primo fattore tende a 1 e, perciò, la dissimilarità δ_2 è all'incirca uguale alla distanza euclidea al quadrato; al contrario se $S_i = S_{i'}$, allora la 2.7 si riduce a $1.1\|X_i - X_{i'}\|^2$ e si è nuovamente introdotto un fattore di repulsione tra le unità i e i' . Un esempio in cui viene invece introdotto un effetto di attrazione, è facilmente fornito dalla dissimilarità δ_3 . Infatti, se $S_i = S_{i'}$, la (2.4) coincide con la distanza euclidea al quadrato valutata sull'insieme degli attributi non protetti; al contrario, se $S_i \neq S_{i'}$, si ottiene $\|X_i - X_{i'}\|^2 - 4u$: in altre parole la distanza euclidea al quadrato viene diminuita in modo proporzionale rispetto al parametro u , introducendo un effetto di attrazione tra le unità considerate. Infine si può vedere cosa succede alla dissimilarità δ_4 fissando $V = c \geq 0$, $u = 0.1$, $v = 100$ e $w = 1$. Si ottiene:

$$\delta_4((X_i, S_i), (X_{i'}, S_{i'})) = \left(1 + 0.1sgn(cS_i^T S_{i'})\left(1 - e^{-100(cS_i^T S_{i'})^2}\right)e^{-\|X_i - X_{i'}\|}\right) \|X_i - X_{i'}\| \quad (2.8)$$

Nel caso in cui $S_i = S_{i'}$, la 2.8 diventa $\left(1 + 0.1e^{-\|X_i - X_{i'}\|}\right) \|X_i - X_{i'}\|$ in cui il primo fattore è una quantità maggiore di uno: si ha in altre parole un effetto di repulsione. Nel caso in cui $S_i \neq S_{i'}$, si ottiene $\left(1 - 0.1e^{-\|X_i - X_{i'}\|}\right) \|X_i - X_{i'}\|$, che può essere

vista come un effetto di attrazione poichè il primo fattore è una quantità minore di uno.

Ovviamente i parametri usati negli esempi sono stati scelti per uno scopo esplicativo in modo da semplificare i calcoli e rendere immediatamente visibili le perturbazioni introdotte: è possibile attuare scelte diverse in base al problema in analisi e agli obiettivi stabiliti.

È necessario sottolineare che non è stato introdotto alcun vincolo di positività sulle nuove dissimilarità: è possibile dunque che, nel calcolo della matrice di dissimilarità, che si indicherà con $\Delta_{n,n}$ per distinguerla da $D_{n,n}$, alcuni elementi risultino negativi. Per rimediare, è sufficiente trasformare tali elementi come segue al fine di renderli positivi.

$$\delta_{m,ii'} = \delta_{m,ii'} + |min\Delta_{n,n}| + \epsilon \quad (2.9)$$

Nell'espressione (2.9), $\delta_{m,ii'}$ indica un generico elemento della matrice di dissimilarità $\Delta_{n,n}$ ottenuto applicando uno dei nuovi indici introdotti δ_m (con $m = 1, 2, 3, 4$) a due unità indicate con i e i' e ϵ è un numero piccolo scelto a piacere.

2.4 Il metodo di Ward nel fair clustering

I metodi gerarchici, già introdotti nella sezione 1.4, raggruppano un vasto insieme di procedure largamente utilizzate. La principale caratteristica è quella di produrre non una singola, ma una gerarchia di partizioni. Il punto di partenza è sempre una matrice di dissimilarità calcolata su coppie di unità, ma ciò che distingue un metodo dall'altro è la definizione delle dissimilarità tra i gruppi. Si sono già trattati i metodi del legame singolo, del legame completo e del legame medio. Si è anche visto come sia fondamentale la conoscenza della matrice dei dati $X_{n,k}$ per poter applicare il metodo dei centroidi e il metodo di Ward. Concentrando l'attenzione su quest'ultimo, si ricorda brevemente come è definita la distanza tra due cluster:

$$d_W(C_1, C_2) = \frac{n_1 n_2}{n_1 + n_2} \|\bar{x}_1 - \bar{x}_2\|^2 \quad (2.10)$$

dove \bar{x}_1 e \bar{x}_2 indicano rispettivamente il centroide del primo e del secondo cluster. È possibile adattare il metodo di Ward nel contesto di una tecnica di clustering di attrazione e repulsione, applicando ai centroidi una delle dissimilarità δ_m al posto della distanza euclidea. Per chiarezza di notazione, si continueranno ad utilizzare gli indici 1 e 2 per indicare i cluster; gli attributi non protetti saranno identificati con X , mentre gli attributi sensibili con S . Si può allora definire:

$$\delta_{W,m}(C_1, C_2) = \frac{n_1 n_2}{n_1 + n_2} \delta_m((\bar{X}_1, \bar{S}_1), (\bar{X}_2, \bar{S}_2)) \quad (2.11)$$

in cui δ_m è una tra le 4 dissimilarità introdotte nella sezione 2.3 e la notazione di barrato indica sempre la media calcolata sui punti appartenenti a un cluster. Poichè esiste una distinzione tra gli attributi non protetti e quelli sensibili, è possibile anche definire:

$$d_x(C_1, C_2) = \|\bar{X}_1 - \bar{X}_2\| \quad (2.12)$$

$$d_s(C_1, C_2) = \|\bar{S}_1 - \bar{S}_2\| \quad (2.13)$$

ossia le distanze tra i centri di calcolati solamente per gli attributi non protetti e per gli attributi sensibili.

Il metodo di Ward non è ottenibile come caso particolare dalla formula di Lance e Williams, tuttavia una formula ricorsiva apposita è proposta da Wishart:

$$d_W(C_1 \cup C_2, C_3) = \frac{n_1 + n_3}{N} d_W(C_1, C_3) + \frac{n_2 + n_3}{N} d_W(C_2, C_3) - \frac{n_3}{N} d_W(C_1, C_2) \quad (2.14)$$

in cui $N = n_1 + n_2 + n_3$. Per quanto riguarda la dissimilarità δ_1 , la formula ricorsiva corrispondente non è altro che un adattamento della (2.14), in cui le distanze tra cluster sono calcolate con la (2.11) scegliendo $m = 1$. Si ottiene:

$$\delta_{W,1}(C_1 \cup C_2, C_3) = \frac{n_1 + n_3}{N} \delta_{W,1}(C_1, C_3) + \frac{n_2 + n_3}{N} \delta_{W,1}(C_2, C_3) - \frac{n_3}{N} d_{W,x}^2(C_1, C_2) \quad (2.15)$$

Per le dissimilarità δ_2 e δ_3 si hanno invece le seguenti formule ricorsive:

$$\begin{aligned} \delta_{W,2}(C_1 \cup C_2, C_3) &= \left(1 + ue^{-v \left(\frac{n_1}{n_1+n_3} d_s^2(C_1, C_3) + \frac{n_2}{n_1+n_2} d_s^2(C_2, C_3) - \frac{n_1 n_2}{(n_1+n_2)^2} d_s^2(C_1, C_2) \right)} \right) \\ &\times \left(\frac{n_1 + n_3}{N} d_{W,x}^2(C_1, C_3) + \frac{n_2 + n_3}{N} d_{W,x}^2(C_2, C_3) - \frac{n_3}{N} d_{W,x}^2(C_1, C_2) \right) \\ \delta_{W,3}(C_1 \cup C_2, C_3) &= \frac{n_1 + n_3}{N} \delta_{W,3}(C_1, C_3) + \frac{n_2 + n_3}{N} \delta_{W,3}(C_2, C_3) - \frac{n_3}{N} \delta_{W,3}(C_1, C_2) \end{aligned}$$

A questo punto gli step per eseguire la procedura sono familiari: dopo aver determinato la matrice di dissimilità $\Delta_{n,n}$, con una scelta dell'indice di dissimilarità e relativi parametri, si effettuano le correzioni degli eventuali valori negativi ottenuti; successivamente si aggregano le due unità più vicine tra loro e si ricalcola la matrice grazie alle formule ricorsive, finché non si è arrivati ad aggregare tutte le unità in un solo cluster.

2.5 Scaling multidimensionale e metodi Kernel

Nella presente sezione si procede all'illustrazione di due tecniche utili per facilitare l'interpretazione dei risultati e per adattare il più possibile le tecniche di clustering alla geometria dei dati in analisi.

Le dissimilarità δ_m ($m = 1, 2, 3, 4$) possono essere utilizzate direttamente con un metodo gerarchico aggregativo. Tuttavia, è possibile ampliare il loro utilizzo anche ad altre tecniche di clustering non gerarchiche in cui l'algoritmo si basa su una funzione da ottimizzare: tra i più utilizzati si ha il metodo delle k -medie, trattato nel primo capitolo, e il metodo PAM (Partitioning Around Medoids) [5]. Quest'ultimo consiste nell'individuazione di k oggetti rappresentativi, definiti medoidi, per i quali la dissimilarità media delle unità assegnate a un cluster rispetto al medoide risulta minima. Si potrebbe immaginare di dover scegliere dove localizzare g servizi in una città in modo che tutti gli utenti siano soddisfatti e percorrono la minima distanza per raggiungere il servizio. Tali metodi definiti di ottimizzazione da Everitt [1], sono spesso applicati a dati che sono stati precedentemente trasformati in uno spazio

euclideo adatto.

Con la tecnica dello scaling multidimensionale, che in generale consiste nella ricerca di uno spazio euclideo di dimensioni ridotte in cui la geometria dei dati trasformati è il più possibile simile a quella originale, si rappresentano i dati in uno spazio $\mathbb{R}^{d'}$ con $d' \leq d$. Quindi, combinare la tecnica dello scaling multidimensionale con le dissimilarità sopra illustrate, consente di arrivare a una rappresentazione dei dati con una geometria simile a quella originale per quanto riguarda gli attributi non protetti e che, al tempo stesso, favorisce la diversità rispetto agli attributi sensibili. È ovvio che lo scaling multidimensionale, riducendo le dimensioni dello spazio in cui si rappresentano i dati, comporta una perdita di informazione: tuttavia in questo contesto lo scaling non è utilizzato in senso stretto come una tecnica di riduzione delle dimensioni, ma ha l'obiettivo di "trasportare" i risultati dell'applicazione delle dissimilarità δ_m in uno spazio euclideo. Per questo motivo, spesso si sceglie uno spazio di dimensioni d' molto simile a quello originario.

Se lo scaling multidimensionale consente di ottenere dei risultati più facilmente interpretabili, i metodi kernel costituiscono, invece, uno strumento estremamente utile per rispettare e adattare la tecnica di clustering alla geometria dei dati.

I metodi kernel sono una classe di algoritmi utilizzati nella pattern analysis, un insieme di tecniche (tra cui per l'appunto la cluster analysis) che ricerca regolarità sistematiche all'interno dei dati. Normalmente i metodi della cluster analysis (gerarchici e non) identificano dei cluster con una particolare forma geometrica: si tratta quasi sempre di cluster convessi e con contorni lineari. Può succedere tuttavia che questa particolare forma a volte non si adatti bene alla geometria dei dati in analisi: i cluster ottenuti potrebbero di conseguenza rappresentare una sintesi non soddisfacente delle informazioni.

In questi casi, una trasformazione non lineare dei dati originali potrebbe aiutare a dare luogo a una disposizione dei punti nello spazio che sia meglio rappresentabile da cluster convessi. Tale trasformazione non lineare è realizzata con i metodi kernel e spesso si parla di *kernel trick* per indicare questa procedura di "aggiramento dell'ostacolo". In generale un kernel è una funzione simmetrica e non negativa $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$. L'idea è quella di effettuare una trasformazione dei dati tramite un kernel adatto e poi applicare ad essi la corrispondente versione kernel delle dissimilarità δ_m . Nel caso di δ_1 si ha:

$$\delta_{k,1}((X_i, S_i), (X_{i'}, S_{i'})) = 1^T U 1 + S_i^T V S_{i'} + d_k^2(X_i, X_{i'}) \quad (2.16)$$

in cui la distanza euclidea al quadrato tra gli attributi non protetti è sostituita da $d_k^2(X_i, X_{i'})$ così definita:

$$d_k^2(X_i, X_{i'}) = k(X_i, X_i) + k(X_{i'}, X_{i'}) - 2k(X_i, X_{i'}) \quad (2.17)$$

2.6 Misure di diversità e qualità

L'obiettivo principale di una procedura di fair clustering è quella di favorire la diversità all'interno dei cluster con riferimento all'insieme degli attributi sensibili. Dopo aver eseguito la procedura, può essere dunque interessante misurare il grado

di diversità che caratterizza la partizione ottenuta.

Un primo indice adatto a questo scopo è l'indice di *Balance*. In particolare, immaginando di avere un solo attributo protetto che, per semplicità, può assumere solo la modalità *A* e la modalità *B*, l'equilibrio del generico cluster C_c è definito come segue:

$$\text{balance}(C_c) = \min\left(\frac{\#A}{\#B}, \frac{\#B}{\#A}\right) \quad (2.18)$$

ossia come il minimo tra la proporzione delle unità che presentano la modalità *A* dell'attributo *S* su quelle che presentano la modalità *B* e il suo reciproco. Nel caso di un attributo sensibile dicotomico, di fatto si sta considerando a numeratore il totale delle unità appartenenti a una ideale minoranza e a denominatore il resto della popolazione e viceversa.

In riferimento a una partizione delle unità (che si indica con \mathcal{C}), l'indice di equilibrio è definito come:

$$\text{balance}(\mathcal{C}) = \min_{C_c \in \mathcal{C}} \text{balance}(C_c) \quad \forall c = 1, \dots, g \quad (2.19)$$

Trattandosi di proporzioni, ogni indice di Balance calcolato su un insieme di punti potrà assumere valori nell'intervallo $[0, 1]$: si avrà il valore 1 nel caso in cui il numero di unità di un gruppo è uguale al numero di unità dell'altro (si è nel caso di massimo equilibrio); al contrario si otterà il valore minimo se una delle due "sottopolazioni" dovesse risultare assente (caso di massimo squilibrio). È ovvio che l'indice calcolato con riferimento a una partizione dei dati assumerà valori minori, o al massimo uguali, dell'indice calcolato sul totale n . Di conseguenza, se l'intero dataset non è perfettamente bilanciato, non potrà esserlo nessuna delle partizioni possibili.

Un'altra possibile misura di diversità è l'*Imbalance Index*. Supponendo di avere una partizione dei dati in g cluster, è possibile definire due vettori di proporzioni. In particolare, si indica con p_c il vettore che raccoglie le proporzioni di ogni modalità assumibile dagli attributi protetti nel generico cluster C_c ; mentre p_t rappresenta il vettore delle stesse proporzioni valutate sul totale dei dati. Si definisce allora il seguente indice:

$$\text{imbalance}(\mathcal{C}) = \frac{1}{g} \sum_{c=1}^g \|p_c - p_t\| \quad (2.20)$$

Di fatto per ogni cluster C_c si sta valutando la distanza tra i due vettori p_c e p_t introdotti. Si può osservare che un valore di $\text{imbalance}(\mathcal{C}) = 0$ è ottenibile solo se, per ogni cluster, si verifica l'uguaglianza tra i vettori delle proporzioni: in tal caso la partizione considerata rispetta la condizione (2.1) di cluster bilanciati.

Per dei risultati ottimali, tuttavia, oltre a una misurazione della diversità, è necessario anche valutare la qualità di una partizione. Un indice molto utilizzato è il *Silhouette Index*, abbreviato in SI, che misura quanto gli elementi appartenenti a uno stesso cluster siano effettivamente simili tra loro, rispetto agli elementi assegnati a cluster differenti. Si sa infatti che una buona partizione deve essere costituita da cluster omogenei e ben separati tra loro, in accordo con la definizione di partizione ben strutturata (1.31). Se l'unità i appartiene al cluster *A* e il cluster *B* è quello più vicino ad *A*, l'indice SI è definito come segue:

$$SI(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \quad (2.21)$$

in cui $a(i)$ è la distanza media dall'unità i delle unità che appartengono allo stesso cluster A , mentre $b(i)$ rappresenta la distanza media dall'unità i delle unità che appartengono al cluster B . Risulta ovvio che per calcolare questo indice è necessario conoscere la composizione dei gruppi e tutte le distanze tra coppie di unità. Il Silhouette Index di un gruppo si ottiene calcolando una media degli SI delle unità appartenenti a tale gruppo. Per come è costruito, l'indice SI può assumere valori nell'intervallo $[-1, 1]$. In particolare, l'indice assumerà valori prossimi a 1 quando $a(i)$ è molto più piccolo di $b(i)$: ciò significa che l'unità i risulta molto più vicina alle unità che appartengono ad A rispetto alle unità assegnate al cluster più vicino. In altre parole, la partizione è ben strutturata. Al contrario, se l'indice si avvicina al suo minimo, vuol dire che l'unità i , in media, è più vicina alle unità del cluster B rispetto alle unità appartenenti allo stesso cluster: si tratta di una partizione non ottimale in cui l'unità i dovrebbe appartenere al cluster B piuttosto che al cluster A . Un ultimo caso particolare è quello in cui l'indice assume valori vicini allo 0: ciò si può verificare quando $a(i)$ è all'incirca uguale a $b(i)$. Ciò significa che l'unità i si trova in una posizione intermedia tra i due cluster: pur essendo stata assegnata al cluster A , l'unità risulta "vicina" alle unità del cluster B .

Dovrebbe essere chiaro a questo punto come l'obiettivo di una procedura di fair clustering sia quello di incrementare la diversità rispetto a una partizione ottenuta con un metodo classico. Per valutare questo aspetto, si può utilizzare l'indice ARI descritto nella sezione 1.6. Con questo indice è possibile mettere a confronto una partizione ottenuta con uno dei metodi illustrati nel capitolo 1, con una partizione ottenuta con l'obiettivo di favorire la diversità con riguardo alla classe degli attributi protetti. Le eventuali differenze potranno essere interpretate come gli effetti indotti dalle perturbazioni delle nuove dissimilarità usate.

Capitolo 3

Caso studio

3.1 Introduzione e descrizione del dataset

Dopo aver delineato la cornice teorica entro cui si colloca, la procedura di Attraction Repulsion clustering verrà applicata ad un dataset estratto dalla *Schools Civil Rights Data Collection* (abbreviata con l'acronimo CRDC) con riferimento all'anno 2017-2018. Il software utilizzato è R.

Dal 1968 la CRDC raccoglie dati e informazioni sui principali problemi legati all'istruzione e ai diritti civili nelle scuole pubbliche degli Stati Uniti. In particolare, si raccolgono informazioni circa le caratteristiche delle scuole e dei programmi e dei servizi offerti agli studenti, distinguendo questi ultimi per sesso, etnia, disabilità e livello di apprendimento della lingua inglese. I dati sono raccolti con indagini biennali e l'obbligo di risposta è previsto dalla legge.

L'obiettivo di questa raccolta è il monitoraggio del sistema scolastico pubblico per assicurarsi, in particolar modo, che gli studenti non siano vittime di discriminazioni basate su variabili come sesso, etnia o nazionalità.

I dati sono disponibili per due diversi domini: scuole e LEA (Local Educational Agency). Queste ultime sono parte essenziale del sistema scolastico statunitense e sono responsabili della distribuzione di servizi legati all'istruzione in una particolare area geografica, nonché garanti che le scuole sotto la loro giurisdizione rispettino le norme federali.

Per effettuare l'analisi si è scelto un dataset in cui le unità di riferimento sono le singole scuole. Nel dettaglio si è estratto dalla CRDC il dataset facente riferimento al numero di nuovi iscritti nelle scuole dello stato del Massachusetts. Tali informazioni erano disponibili in base all'etnia degli studenti: bianchi, di colore, ispanici, asiatici, nativi americani e Hawaiani (o residenti nelle isole del pacifico). La variabile etnia sarà la variabile che si è definita sensibile: essa giocherà un ruolo fondamentale nella applicazione della tecnica di clustering di attrazione e repulsione.

Lo stato del Massachusetts costituisce, insieme ad altri 5, la regione della Nuova Inghilterra. Tra di essi risulta essere il più popoloso e, tra tutti gli Stati Uniti d'America, occupa il sesto posto per il PIL procapite e il secondo per i livelli dell'indice di sviluppo umano (ISU), che misura e confronta i livelli di benessere tramite PIL procapite, tasso di alfabetizzazione e speranza di vita.

Grazie alla funzione `geocode()`, presente nel pacchetto `ggmap`, a partire dal nome

della scuola, è stato possibile ricavare le coordinate geografiche (ossia la coppia di latitudine e longitudine) che identificano le posizioni delle scuole. Durante la fase di geolocalizzazione si sono riscontrati dei problemi: in particolare alcune scuole non venivano localizzate poiché si trattava di istituti distinti ma con lo stesso nome, mentre altre venivano localizzate al di fuori dello stato americano. Per quanto riguarda il problema dei "doppioni", dopo averli individuati, si è deciso di mantenere nel dataset solamente la scuola che presentava il numero di iscritti maggiore; le scuole localizzate al di fuori del territorio di interesse, invece, sono state individuate grazie ai range di latitudine e longitudine che caratterizzano il Massachusetts, per poi procedere alla loro eliminazione dalla popolazione di riferimento. In questo modo, si sono mantenute 1807 delle 1870 scuole originarie.

A questo punto la funzione `geompoint`, ha permesso di rappresentare con dei punti, individuati dalle coordinate, le scuole. La figura 3.1 rappresenta la cartina dello stato del Massachusetts e la locazione delle scuole sul territorio.

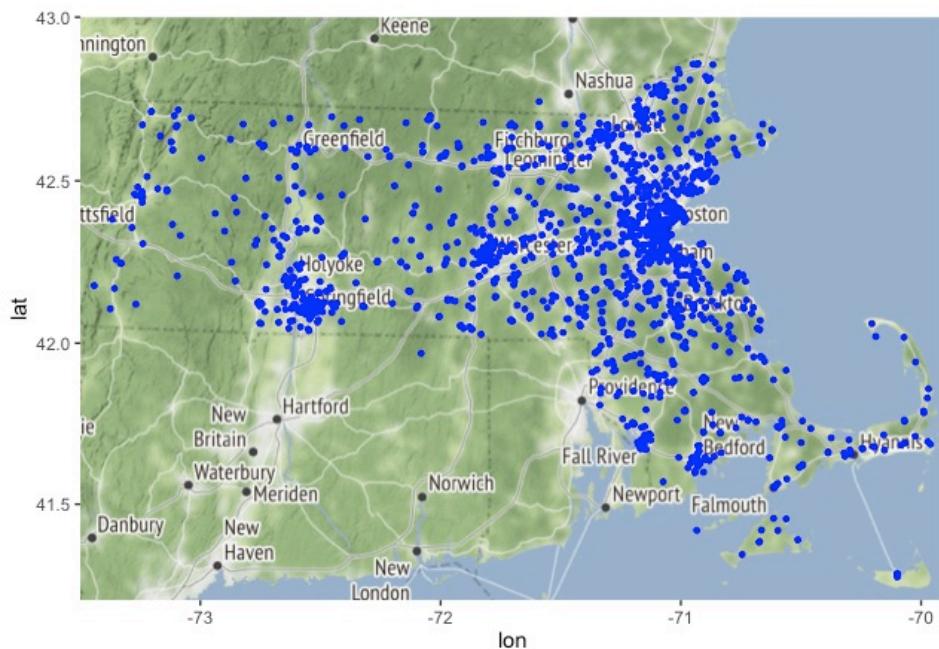


Figura 3.1. Disposizione delle scuole sul territorio del Massachusetts

L'analisi sarà articolata come segue: nella prima parte si analizzeranno i dati con le tecniche di clustering gerarchiche classiche, senza tenere conto della composizione etnica. Si utilizzeranno diversi metodi e si cercherà di individuare una partizione e un numero di gruppi ottimale sulla base del Silhouette Index e dell'Imbalance Index. Successivamente, con riferimento a tale partizione ottimale, verranno applicate le tecniche di Attraction Repulsion Clustering. In conclusione si effettuerà un

confronto e un commento dei risultati sulla base della diversità introdotta con riguardo all'attributo sensibile etnia.

L'obiettivo è identificare dei gruppi di scuole compatti e il più possibile eterogenei dal punto di vista della composizione etnica, in modo che tali gruppi possano essere identificati come distretti scolastici. Ovviamente le due richieste di compattezza e, al tempo stesso, diversità nella composizione etnica potrebbero entrare in conflitto tra di loro e, perciò, potrebbe essere necessario ricercare un compromesso.

3.2 Procedura classica

Nella presente sezione ci si concentrerà sull'applicazione dei metodi gerarchici classici. Dalle coordinate delle scuole, grazie alla funzione `geodist` presente nell'omonimo pacchetto, si è ricavata la distanza geodetica tra coppie di scuole. In matematica, e più precisamente in geometria differenziale, una geodetica è la curva più breve che congiunge due punti di uno spazio (lo spazio in questione può essere una superficie). Disponendo solamente di una matrice di distanze tra le scuole, si è scelto di utilizzare il metodo del legame completo, del legame medio e del legame singolo. Il metodo del legame singolo è stato poi scartato in quanto l'analisi del dendrogramma ha evidenziato un forte effetto catena nel processo di aggregazione delle unità.

Per gli altri due metodi (medio e completo) non si riportano i dendrogrammi poichè, a causa dell'elevato numero di unità, risulterebbero di difficile lettura. Successivamente, per entrambi i metodi, si è proceduto al calcolo del Silhouette Index (SI) e dell'Imbalance Index (II) per un numero di cluster che varia da 2 a 15. Il grafico 3.2 rappresenta l'andamento dei due indici all'aumentare del numero di gruppi.

Si può osservare come il metodo del legame medio presenti dei valori del Silhouette Index superiori rispetto a quelli del legame completo fino alla partizione in 6 gruppi; la relazione presenta, invece, più alternanze per l'Imbalance Index, tuttavia si tratta sicuramente di differenze più contenute. Sulla base dell'andamento degli indici, la partizione ottimale risulterebbe essere quella in 2 gruppi, che presenta il valore maggiore del SI e quello minore dell'II. Tuttavia, considerato il numero elevato di unità, si è ritenuto che una partizione in 2 gruppi risulterebbe riduttiva e poco adatta ai fini dell'obiettivo della determinazione di distretti scolastici. Inoltre, riducendo il numero di gruppi, ciò che accade è che il gruppo di scuole attorno alla capitale Boston tende a inglobare i gruppi confinanti. Tenendo a mente le suddette riflessioni e l'obiettivo dell'analisi, si è deciso di prendere in considerazione la partizione ottenuta con il metodo del legame medio con 6 gruppi e quella ottenuta con il metodo del legame completo con 10 gruppi. I valori del Silhouette Index e dell'Imbalance Index di tali partizioni sono riassunti di seguito:

	Medio 6	Completo 10
SI	0.41	0.42
II	0.12	0.16

Si tratta di due partizioni abbastanza compatte e con valori dell'Imbalance Index contenuti. L'obiettivo a questo punto è la ricerca di partizioni che, pur presentando le stesse caratteristiche in termini di metodo utilizzato e numero di gruppi considerato,

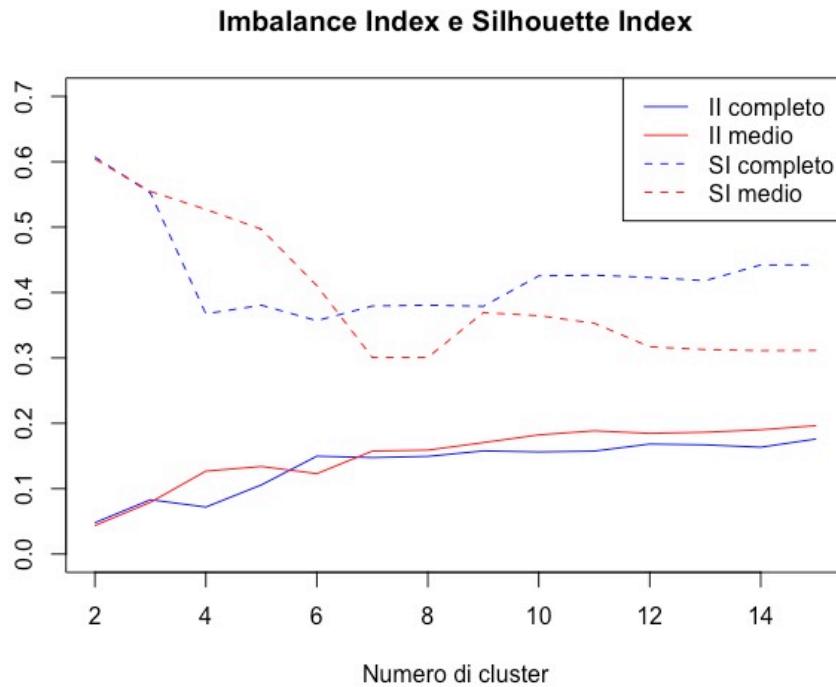


Figura 3.2. Silhouette index e Imbalance index per i metodi del legame medio e completo

siano caratterizzate da maggiore diversità nei cluster senza andare a danneggiare eccessivamente la compattezza degli stessi.

3.3 Attraction Repulsion

In questa parte dell’analisi si è presa in considerazione anche la composizione etnica degli iscritti in ogni scuola. Si ricorda che la variabile etnia può assumere 6 modalità: bianco, di colore, ispanico, nativo americano, asiatico e hawaiiano o residente nelle isole del pacifico. La codifica della variabile sensibile etnia è stata immediata, in quanto si sono considerati i numeri di iscritti per ogni etnia in ogni scuola.

La tecnica è stata eseguita grazie al pacchetto `AttractionRepulsionClustering`, di cui si riportano in appendice le funzioni principali, con relativi argomenti necessari e oggetti prodotti.

Nel secondo capitolo sono state introdotte le nuove dissimilarità (2.2|2.5), che, per essere applicate, necessitano della definizione di una serie di parametri.

In questa sezione, seguendo la scelta degli autori dell’articolo [3], si utilizzeranno le dissimilarità δ_1 , δ_2 e δ_4 per ottenere una partizione in 6 gruppi con il metodo del legame medio e una in 10 gruppi con il metodo del legame completo, per poi confrontare i risultati con le partizioni ottenute con il metodo classico nella sezione precedente.

Per procedere con ordine saranno riportati prima i risultati della partizione in 6 gruppi e poi quelli della partizione in 10 gruppi.

3.3.1 Applicazione delle nuove dissimilarità

Prima di procedere all'illustrazione dei risultati, si ritiene opportuno presentare la scelta dei parametri utilizzati per le 3 dissimilarità.

Partendo da δ_1 , essa necessita di un parametro v_0 , che determina l'intensità dell'interazione tra le classi dell'attributo sensibile, e di due matrici U e V . Le dimensioni di queste ultime dipendono dal numero di modalità assumibili dalla variabile etnia: in questo caso sono entrambe delle matrici quadrate 6×6 . La matrice U per la dissimilarità δ_1 determina una perturbazione extra e, spesso, può essere posta uguale alla matrice nulla. La matrice V controlla invece le interazioni tra le varie modalità dell'attributo sensibile. La scelta più ovvia quindi per questa matrice è la seguente:

$$V = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix} \quad (3.1)$$

In questo modo si cerca di introdurre un effetto di repulsione quando l'etnia è la stessa e un effetto di attrazione ogni volta che si hanno modalità diverse.

Per quanto riguarda, invece, la scelta del parametro v_0 , essendo l'obiettivo quello di introdurre diversità nella partizione, si è scelto di applicare la procedura con δ_1 su una griglia di 100 possibili valori del parametro compresi tra 0 e 10, per poi selezionare il valore di v in corrispondenza del quale risulta minimo il valore dell'Imbalance Index.

Per δ_2 è necessario definire i due parametri u e v . Si ricorda brevemente che con u è possibile controllare la massima perturbazione raggiungibile, mentre v stabilisce quanto velocemente ci si allontana da tale massimo. Anche in questo caso si è deciso di lavorare con due griglie di possibili valori dei due parametri. In particolare per $u = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ e $v = \{0.1, 0.3, 0.5, 0.7, 0.9, 1, 3.25, 5.5, 7.75, 10\}$.

Infine, δ_4 necessita della definizione di 3 parametri: per v si è utilizzata la griglia $v = \{0.1, 1, 10\}$; per u si sono considerati i valori $u = \{0.1, 0.3, 0.5, 0.7, 0.9\}$; infine il parametro di località w ha assunto un valore tra i seguenti $w = \{0.1, 0.5, 0.9, 1, 5.5, 10\}$. Con δ_4 si è deciso di ridurre il numero di possibili valori dei parametri a causa dell'elevata complessità computazionale: per minimizzare l'Imbalance Index rispetto alle 180 combinazioni dei parametri considerati sono state necessarie circa 26 ore.

Per tutte le dissimilarità si è eseguita la tecnica di fair clustering per ogni possibile valore e combinazione dei parametri considerati, per poi, essendo l'obiettivo quello di introdurre diversità dal punto di vista della composizione etnica, individuare la configurazione ottimale dei parametri rispetto alla quale il valore dell'Imbalance Index risulti minimo. Solo in un secondo momento si è proceduto al calcolo del SI associato a tale partizione.

È bene ricordare che le caratteristiche ricercate di diversità e, al tempo stesso, di compattezza dei cluster, potrebbero entrare in contrasto tra loro. Di conseguenza la partizione che minimizza il valore dell'Imbalance Index, potrebbe risultare non soddisfacente dal punto di vista del Silhouette Index.

A questo punto si riportano i risultati ottenuti con riferimento alla partizione in 6 gruppi determinata con il metodo del legame medio.

In particolare con δ_1 si è ottenuta la partizione ottimale con $v_0 = 7.2$, in corrispondenza della quale si ha un valore dell'indice pari a 0.13 e un Silhouette Index pari a 0.43.

Per la dissimilarità δ_2 la soluzione ottima è stata raggiunta in corrispondenza della coppia $u = 0.1$ e $v = 0.1$, con Imbalance Index e Silhouette Index pari, rispettivamente, a 0.14 e 0.44.

Infine, con δ_4 si è ottenuta la soluzione ottima per valori dei parametri $v = 0.1$, $u = 0.1$ e $w = 0.1$. In questo caso si è registrato un valore dell'Imbalance Index pari a 0.12 e uno del Silhouette Index pari a 0.41.

Riassumendo:

	classica	δ_1	δ_2	δ_4
II	0.12	0.13	0.14	0.12
SI	0.41	0.43	0.44	0.41

Si nota immediatamente come l'utilizzo delle nuove dissimilarità abbia comportato, non una diminuzione, ma un aumento dell'Imbalance Index rispetto alla partizione classica (ad eccezione di δ_4 con la quale rimane inalterato). Probabilmente ciò è dovuto al fatto di aver scelto un numero troppo piccolo di gruppi: la loro disposizione sul territorio risulta tale per cui, pur introducendo delle perturbazioni nelle misure di dissimilarità tra scuole, ciò non risulta sufficiente per determinare un incremento della diversità. Anzi, i pochi cambiamenti introdotti, determinano un incremento dell'II rispetto alla partizione classica che, in questo caso, è quella migliore rispetto alla diversità etnica.

Le partizioni ottenute, inoltre, non differiscono in modo significativo tra di loro. Tale caratteristica è visibile già graficamente, confrontando le cartine di ogni partizione riportate di seguito (3.3):

Un'ulteriore conferma all'intuizione grafica è data dai valori assunti dall'indice ARI (1.51), rispettivamente calcolato tra la partizione classica e quelle ottenute con le nuove dissimilarità.

	δ_1	δ_2	δ_4
classica	0.99	0.95	1

Dalla tabella si vede come le partizioni ottenute siano molto simili tra di loro, determinando valori estremamente alti dell'ARI, e come, addirittura, la partizione ottenuta con δ_4 coincida con quella classica.

Dopo aver illustrato i risultati con riferimento al metodo del legame medio in 6 gruppi, si passa a considerare la partizione ottenuta con il metodo del legame completo in 10 gruppi.

Le griglie dei parametri sono le stesse già utilizzate per la precedente partizione.

In questo caso, con δ_1 si è ottenuta la partizione ottimale con un valore del parametro

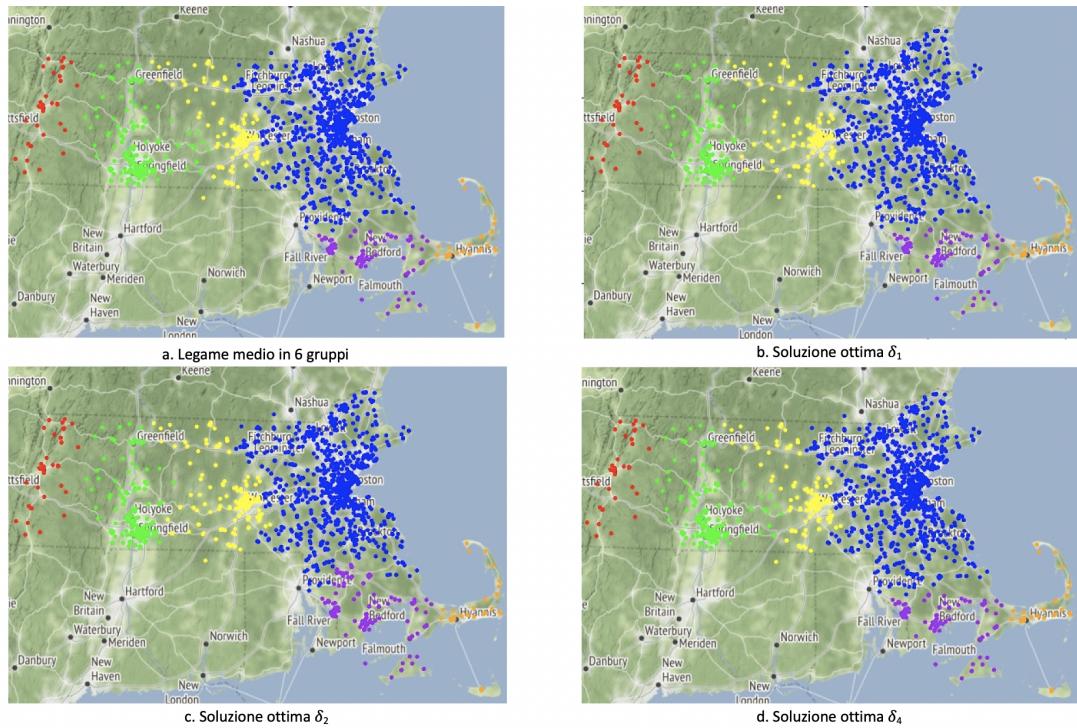


Figura 3.3

$v_0 = 8.3$, in corrispondenza del quale si è registrato un valore dell’Imbalance Index pari a 0.14 e un valore del Silhouette Index pari a 0.38.

Con δ_2 , invece, il valore minimo pari a 0.15 dell’Imbalance Index si è determinato in corrispondenza dei parametri $u = 1$ e $v = 0.1$. La partizione così ottenuta presenta un valore del Silhouette Index pari a 0.40.

Infine, con δ_4 si è determinata la soluzione ottima in corrispondenza dei valori dei parametri $u = 0.1$, $v = 0.1$ e $w = 0.1$. Il valore dell’Imbalance Index risulta pari a 0.16 e quello del Silhouette Index a 0.43.

Riassumendo:

	classica	δ_1	δ_2	δ_4
II	0.16	0.14	0.15	0.16
SI	0.43	0.38	0.40	0.43

Considerando un numero superiore di gruppi, l’applicazione delle nuove dissimilarità riesce, effettivamente, a determinare un decremento dell’Imbalance Index. Tale riduzione avviene a costo di una leggera diminuzione del Silhouette Index, che, tuttavia, si può considerare ancora soddisfacente. Ecco un esempio di come i due criteri di diversità e compattezza possano, a volte, entrare in conflitto tra loro.

Per una visualizzazione grafica, anche in questo caso, si riportano le cartine delle partizioni ottime sopra descritte:

Già visivamente, è possibile vedere come la tecnica di Attraction Repulsion clustering

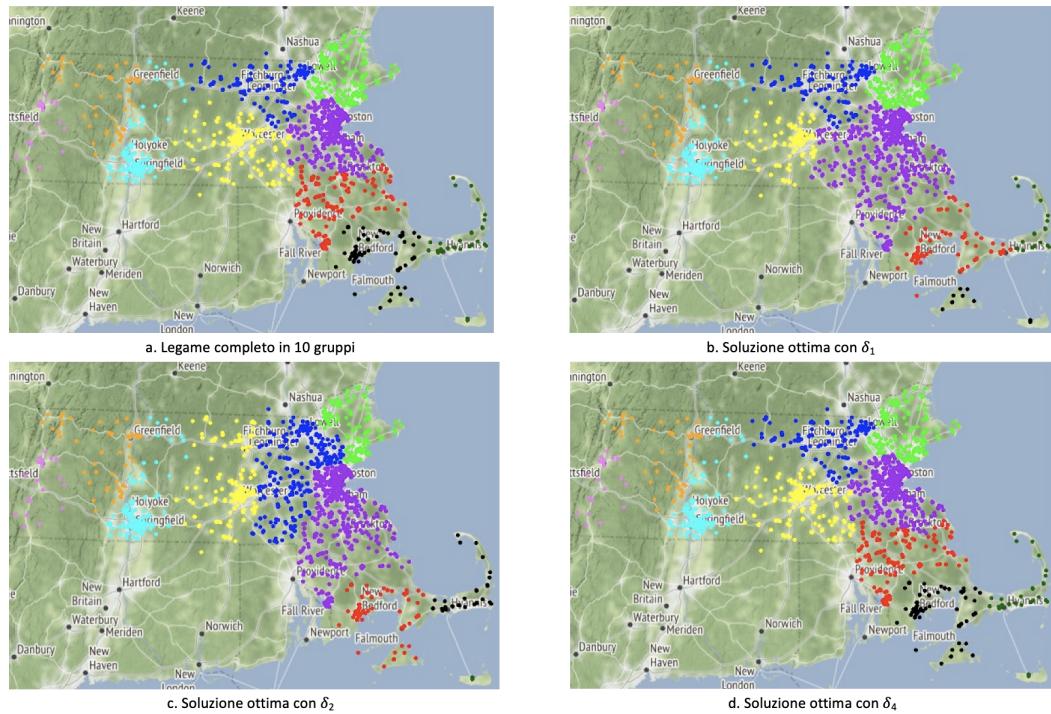


Figura 3.4

abbia, in questo caso, determinato maggiori differenze tra una partizione e l'altra. Si riporta infine la tabella riassuntiva dei valori assunti dall'indice ARI:

	δ_1	δ_2	δ_4
classica	0.79	0.61	1

Risulta immediato, anche questa volta, notare come δ_4 determini una partizione coincidente con quella iniziale. Sicuramente, inoltre, valori più bassi dell'ARI sono indice di un maggior successo della tecnica, in termini di diversità introdotta.

3.4 Confronto e conclusioni

Nella sezione precedente si sono riportati i risultati della procedura di Attraction Repulsion clustering applicata in due casi: nel primo per determinare una partizione in 6 gruppi con il metodo del legame medio; nel secondo per determinarne una in 10 gruppi con il metodo del legame completo.

Nel primo caso, la tecnica non è riuscita a introdurre degli effetti significativi in termini di diversità. Le partizioni ottenute risultano estremamente simili tra loro. Tra tutte, la dissimilarità δ_2 è quella che ha avuto un effetto maggiore, determinando un valore dell'ARI pari a 0.95 quando confrontata con la partizione iniziale. Si ricorda che, avendo usato sempre lo stesso metodo e lo stesso numero di gruppi, le eventuali differenze tra una partizione e l'altra sono attribuibili all'effetto di perturbazione introdotto con la tecnica di Attraction Repulsion clustering. Tale

aspetto è misurato, per l'appunto, dall'indice ARI: ci si aspetta un valore dell'indice tanto più basso quanto più si è riusciti a favorire la diversità nella partizione. Mentre i valori del Silhouette Index rimangono sempre soddisfacenti, al contrario delle aspettative, il valore dell'Imbalance Index aumenta nelle partizioni determinate con δ_1 , δ_2 e δ_4 . Inoltre, si vede graficamente, come gli unici cambiamenti avvengano tra due gruppi (quelli colorati in giallo e in verde), mentre gli altri rimangano inalterati. Probabilmente ciò è dovuto al fatto di aver scelto un numero troppo esiguo di gruppi: poiché la compattezza geografica rimane un aspetto fondamentale, 6 gruppi risultano insufficienti per poter registrare i cambiamenti auspicabili nella composizione etnica. In effetti, quando applicata al caso del metodo del legame completo con 10 gruppi, gli effetti introdotti sono più evidenti, in particolare per δ_1 e δ_2 . Dai valori riportati nella tabella 3.3.1, si vede come ad una diminuzione dell'Imbalance Index corrisponda una diminuzione anche del Silhouette Index: ad un incremento della diversità corrisponde una leggera diminuzione della compattezza della partizione. Ancora una volta δ_2 è la dissimilarità che restituisce il risultato migliore riuscendo, più delle altre, a introdurre una perturbazione al fine di favorire la diversità nella partizione.

Anche in questo caso, tuttavia, δ_4 non riesce a determinare degli effetti concreti, restituendo una partizione identica a quella di partenza.

In conclusione, si può affermare che i risultati migliori sono stati ottenuti quando si è considerata una partizione in 10 gruppi e il metodo del legame completo. Tra le nuove dissimilarità, δ_2 è quella che è riuscita più delle altre a introdurre diversità nella composizione etnica dei distretti scolastici. Per tale partizione viene riportata la cartina nella figura 3.5: le numerosità dei cluster ottenuti, associati ai corrispondenti colori utilizzati per la rappresentazione grafica, sono riassunti nella tabella seguente:

Colore	Numerosità
Blu	647
Verde	179
Viola	487
Arancione	55
Giallo	172
Celeste	137
Rosso	72
Nero	30
Rosa	24
Verde scuro	4

Risulta, infine, interessante anche mettere in relazione i cluster ottenuti con la conformazione geografica del territorio. Lo stato del Massachusetts è caratterizzato da due zone metropolitane: la *Greater Boston* sul versante est e la zona di Springfield a ovest. Con una popolazione di ben 7 029 917 abitanti (al primo aprile 2020), circa due terzi risiedono nella grande Boston, fulcro delle principali attività economiche di tutto lo stato. Come conseguenza si osserva, nella zona orientale dello stato, una concentrazione molto elevata di istituti scolastici. Al contrario, il versante occidentale è attraversato dalle montagne del Berkeshire: ciò determina una minore densità della popolazione e, di conseguenza, una presenza più rada di scuole che risultano anche

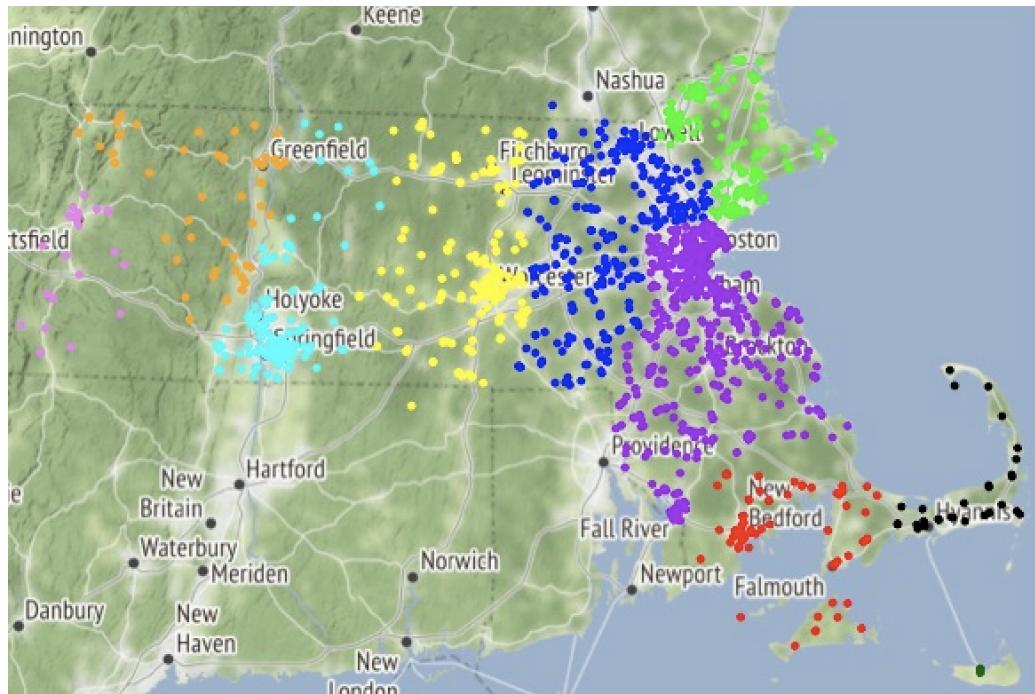


Figura 3.5. Metodo del legame completo con δ_2

più isolate tra loro. In effetti, come conseguenza del maggior grado di isolamento, i cluster collocati in quella zona, più precisamente quelli colorati in rosa, arancione e celeste, non subiscono modifiche rispetto alla partizione classica, rimanendo inalterati. I principali cambiamenti, che giustificherebbero il valore dell'indice ARI pari a 0.61, intercorrono tra i cluster localizzati attorno alle città di Boston e Worcester (seconda città per popolazione dopo la capitale). Si tratta dei gruppi colorati in viola, giallo, blu e verde che risultano essere anche i 4 più numerosi. Nella partizione considerata, risulta interessante notare, nell'area sud-est dello stato, che il gruppo colorato in verde scuro sia composto dalle uniche 4 scuole localizzate nell'isola Adams.

Bibliografia

- [1] BRIAN S. EVERITT, SABINE LANDAU, M. L. D. S. *Optimization Clustering Techniques*. John Wiley Sons, Ltd, 2011.
- [2] CRISTIANINI, N. *Shortcuts to Artificial Intelligence*. MIT Press, 2019.
- [3] DEL BARRIO, E., INOUZHE, H., AND LOUBES, J.-M. Attraction-repulsion clustering: a way of promoting diversity linked to demographic parity in fair clustering. *Advances in Data Analysis and Classification* (2022).
- [4] ECKHOUSE, L., LUM, K., CONTI-COOK, C., AND CICCOLINI, J. Layers of bias: A unified approach for understanding problems with risk assessment. *Criminal Justice and Behavior* 46, 2 (2019), 185–209.
- [5] KAUFMAN L, R. P. *Clustering by means of medoids*. Birkhäuser, 1987.
- [6] ZANI, S., AND CERIOLI, A. *Analisi dei dati e data mining per le decisioni aziendali*. Giuffrè, 2007.

Appendice A

Come installare il pacchetto di Attraction Repulsion clustering?

Il pacchetto non è presente nel CRAN di R, perciò per essere installato è necessario eseguire i seguenti comandi:

- `library(devtools)`
- `devtools::install_github("HristoInouzheAttractionRepulsionClustering", force=TRUE)`
- `library(AttractionRepulsionClustering)`

Cosa c'è nel pacchetto

Nel pacchetto sono disponibili le seguenti funzioni:

- **additive1Distance:**

Sarebbe δ_1 ed esegue una correzione additiva della distanza originale tra gli attributi non protetti grazie ad una matrice di interazione (V).

Sintassi: `additive1Distance(params, distanceMatrix, chargeMatrix)`, in cui:

- `params`: è una lista di parametri per la dissimilarità additiva. Nell'ordine vanno specificati la matrice U , il parametro v e la matrice V .
- `distanceMatrix`: è la matrice delle distanze tra gli attributi non protetti X .
- `chargeMatrix`: è una matrice in cui sono disposti gli attributi sensibili per righe.

Il risultato è una matrice con le nuove distanze calcolate con δ_1 .

- **additive2Distance:** Sarebbe δ_2 ed esegue una correzione additiva della distanze originale tra gli attributi non protetti.

Sintassi: `additive2Distance(params, distanceMatrix, chargedDistance)`, in cui:

- `params`: consiste nel parametro u per la perturbazione.

- **distranceMatrix**: è la matrice di distanze tra gli attributi non protetti.
- **chargedDistance**: è la matrice delle distanze euclidiene tra gli attributi sensibili

Di nuovo il risultato è una matrice con le distanze basate su δ_2 .

- **chargedHclust**:

Esegue la tecnica di clustering di attrazione e repulsione gerarchica.

Sintassi: `chargedHclust(X, Q, Qf, a = 0, b = 1, tipo = "single", tipoq = "additive")`, in cui:

- **X**: matrice con gli attributi non protetti per righe.
- **Q**: matrice con attributi sensibili per colonne.
- **Qf**: matrice di interazione, solo che `tipoq='additive'`.
- **a**: intensità della perturbazione, solo se `tipoq= c('additive2', 'multiplicative')`.
- **b**: velocità nella decrescita della perturbazione, solo se `tipoq='multiplicative'`.
- **tipo**: il tipo di metodo gerarchico. Accetta come valori: 'single', 'average', 'complete', 'mcquitty', 'median', 'centroid', 'ward'.
- **tipoq**: il tipo di dissimilarità da utilizzare. Accetta come valori: 'additive', 'additive2', 'multiplicative'.

Il risultato restituisce una lista di elementi:

- **niveles**: una lista con i valori del livello e del cluster che indica il livello a cui ci si trova nel dendrogramma e quali sono i gruppi da unire.
- **clusters**: una lista in cui ogni entrata è una partizione dei dati a un livello dell'albero.
- **edges**: una struttura a grafo per plottare il dendrogramma con igraph e ggraph.

- **e2DistProp**:

Calcola la distanza euclidea al quadrato tra il vettore delle proporzioni degli attributi protetti in un cluster della partizioni e il vettore delle proporzioni sul totale dei dati.

Sintassi: `e2DistProp(clusters, chargeMatrix, totalPropMatrix)`, in cui:

- **clusters**: è una partizione ottenuta con un metodo.
- **chargeMatrix**: al solito la matrice in cui le righe sono gli attributi protetti.
- **totalPropMatrix**: un vettore con le proporzioni totali degli attributi sensibili.

Il risultato è una lista di elementi:

- **distance**: la distanza euclidea media tra le proporzioni valutate nel cluster e quelle totali.

- **clusterProportions**: riporta per ogni cluster le proporzioni degli attributi sensibili.
- **clusterDistance**: la distanza euclidea media per ogni attributo protetto nei vari cluster rispetto alla proporzione originale specificata in totalPropMatrix.
- **fairDistanceCalc**:

È un wrapper (ossia una funzione che contiene un'altra funzione) per eseguire la tecnica con un particolare valore di perturbazione, per un vettore di diversi numeri di cluster desiderati e per 4 diverse procedure di cluster analysis: metodo singolo, medio, completo e k-medie.

Sintassi:

```
fairDistanceCalc(
  params,
  type,
  distances,
  chargeMatrix,
  chargeDistance,
  Ks,
  totalPropMatrix)
```

in cui:

- **params**: un vettore o una lista con i parametri specificati in accordo alla dissimilarità specificata in **type**.
- **type**: accetta valori in c("additive1", "additive2", "multiplicative", "local", "unperturbed").
- **distances**: la matrice di distanze tra gli attributi non protetti.
- **chargeMatrix**: matrice con gli attributi sensibili per righe.
- **chargeDistance**: matrice delle distanze euclideoe tra gli attributi protetti.
- **Ks**: un vettore con diversi numeri dei cluster desiderati.
- **totalPropMatrix**: un vettore che contiene le proporzioni totali degli attributi protetti.

Il risultato è una lista di elementi ciascuno dei quali, a sua volta, è una lista di elementi.

- **completeHclust**: una lista di risultati per il metodo del legame completo per ogni possibile numero di cluster:
 - * **distance**: la distanza euclidea media tra le proporzioni degli attributi protetti nel cluster rispetto alla proporzione totale.
 - * **clusterProportions**: proporzione degli attributi protetti per ogni cluster.
 - * **clusterDistance**: la distanza euclidea media per ogni attributi protetto nei diversi cluster rispetto alla proporzione totale specificata in totalPropMatrix.

E lo stesso viene prodotto per il k-medie (kmeans), per il metodo del legame singolo (singleHclust) e medio (averageHclust).

- `clusterCompleteHclust`: una lista in cui ogni elemento è la partizione per il corrispondente numero di cluster di Ks. E analogamente viene restituito `clusterKmeans`, `clusterSingleHclust` e `clusterAverageHclust`

- **localDistance:**

Sarebbe δ_4 ed esegue una correzione locale della distanza tra gli attributi non protetti usando una matrice di interazione.

Sintassi: `localDistance(params, distanceMatrix, chargeMatrix, chargedDistance)`, in cui:

- `params`: nell'ordine vettore con u, v, w e V .
- `distanceMatrix`: distanze tra gli attributi non protetti.
- `chargeMatrix`: attributi sensibili per righe.
- `chargedDistance`: distanze euclidee tra gli attributi sensibili.

Restituisce una matrice con i nuovi valori delle dissimilarità.

- **multiplicativeDistance:**

Sarebbe δ_3 ed esegue una correzione moltiplicativa della distanza tra gli attributi non protetti.

Sintassi: `multiplicativeDistance(params, distanceMatrix, chargedDistance)`, in cui:

- `params`: nell'ordine u e v .
- `distanceMatrix`: distanze tra i non protetti.
- `chargedDistance`: distanze euclidee tra i sensibili.

Il risultato è una matrice con le nuove dissimilarità.

Appendice B

Codice R

```

rm(list=ls())
dataset=Dataset_Massachusetts
dataset2=Dataset_Massachusetts
rm(list="Dataset_Massachusetts")
#1816 scuole e 16 variabili

names(dataset)
# Installo i pacchetti necessari
library(fpc)
library(cluster)
library(rgl)
library(rattle)
library(mclust)
library(corrplot)
library(factoextra)
install.packages("ggmap")
library(ggmap)
library(ggplot2)
library(httr)
library(jsonlite)
library(devtools)
devtools::install_github("HristoInouzhe/AttractionRepulsionClustering",
force=TRUE)
library(AttractionRepulsionClustering)
install.packages("geodist")
library(geodist)

# La mia API key privata
register_google(key = "***", write=TRUE)

nomi= paste(dataset$SCH_NAME , "Massachusetts" , sep="")

# con geocode ricavo latitudine e longitudine delle scuole
coordinate_scuole=geocode(nomi, source="google")
summary(coordinate_scuole)
coordinate_scuole=as.data.frame(coordinate_scuole)

# creo una copia delle coordinate per fare delle prove

```

```

prova=coordinate_scuole
prova=as.data.frame(prova)
row.names(prova)=dataset$SCH_NAME

prova[which(prova[,1]< (-73.5)),]
prova=prova[-which(prova[,1]< (-73.5)),]
prova[which(prova[,1]> (-69.9)),]
prova[which(prova[,2]> (43)),]
prova[which(prova[,2]< (41.2)),]

summary(prova)
prova = prova[!is.na(prova[,1]) ,]

#FINALE: 1807 scuole localizzate correttamente nel Massachusetts
dataset=dataset[dataset$SCH_NAME %in% row.names(prova),]
rm(list="coordinate_scuole")
coordinate_scuole=prova
rm(list="prova")
# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data = coordinate_scuole, aes(x = lon, y = lat), color = "blue",
size = 1,na.rm =F)

# Proporzione delle varie etnie nel dataset
p_tot_hispanic= sum(dataset$Hispanic)/sum(dataset$Total)
p_tot_NAmerica= sum(dataset$Namerican)/sum(dataset$Total)
p_tot_Asian= sum(dataset$Asian)/sum(dataset$Total)
p_tot_HP= sum(dataset$'Hawaiian/Pacific')/sum(dataset$Total)
p_tot_black= sum(dataset$Black)/sum(dataset$Total)
p_tot_white= sum(dataset$White)/sum(dataset$Total)

p_tot= c(p_tot_hispanic,p_tot_NAmerica,p_tot_Asian,p_tot_HP,p_tot_black,
p_tot_white)
sum(p_tot)

# Matrice delle distanze tra le scuole
distanze=geodist(coordinate_scuole,paired = T,measure="geodesic")
distanze=as.matrix(distanze)
row.names(distanze)=dataset$SCH_NAME
colnames(distanze)=dataset$SCH_NAME
dim(distanze) #matrice 1807 X 1807

# Controlli sulla matrice delle distanze
sum(distanze-t(distanze))
distanze[100,100]
min(distanze)

distanze=as.dist(distanze) #hclust vuole un oggetto dist come input

# Analisi dei gruppi classica
# LEGAME COMPLETO

```

```

distanze=as.dist(distanze)
completo=hclust(distanze)
plot(completo,main="Complete\u2014method",labels=F)

# CONSIDERO LE PARTIZIONI CON 2 FINO A 15 GRUPPI
II_comp=c() #vettore in cui raccolgo Imbalance Index
SI_comp=c() #vettore in cui raccolgo Silhouette Index
for (num_clu in 2:15){
  cluster <- cutree(completo,k=num_clu)
  SI_comp=c(SI_comp,summary(silhouette(cluster,dist = distanze))$avg.width)
  prop=e2DistProp(cluster,dataset[,4:9],p_tot)
  addendi=c()
  for(i in 1:num_clu){
    addendi=c(addendi,sqrt(sum((prop$clusterProportions[i,]-p_tot)^2)))
  }
  II_comp=c(II_comp,sum(addendi)*(1/num_clu))
}

II_comp
SI_comp

#LEGAME SINGOLO
singolo= hclust(distanze,method = "single")
plot(singolo, main='Single\u2014method',labels = F)

#LEGAME MEDIO
medio=hclust(distanze,method = "average")
plot(medio,main="average\u2014method",labels = F)

II_medio=c() #vettore in cui raccolgo Imbalance Index
SI_medio=c() #vettore in cui raccolgo Silhouette Index
for (num_clu in 2:15){
  cluster <- cutree(medio,k=num_clu)
  SI_medio=c(SI_medio,summary(silhouette(cluster,dist = distanze))$avg.width)
  prop=e2DistProp(cluster,dataset[,4:9],p_tot)
  addendi=c()
  for(i in 1:num_clu){
    addendi=c(addendi,sqrt(sum((prop$clusterProportions[i,]-p_tot)^2)))
  }
  II_medio=c(II_medio,sum(addendi)*(1/num_clu))
}
II_medio
SI_medio

#Rappresentazione degli Imbalance Index dei metodi COMPLETO E MEDIO
plot(2:15,II_comp,type = "l",col="blue",ylim = c(0,0.7),
main="Imbalance\u2014Index\u2014e\u2014Silhouette\u2014Index",
xlab = "Numero\u2014di\u2014cluster",ylab="")

```

```

lines(2:15,II_medio,col='red')
legend("topright",legend = c("II_completo","II_medio","SI_completo",
"SI_medio"), col=c("blue","red","blue","red"),
lty = c(1,1,2,2),lwd=1)
lines(2:15,SI_comp,type = "l",lty=2,col="blue")
lines(2:15,SI_medio,type = "l",lty=2,col="red")

# Scelta1: metodo del legame medio con 6 gruppi
II_medio[5] #0.12
SI_medio[5] #0.41

scelta1= cutree(medio,k=6)
coordinate_scelta1= cbind(coordinate_scuole,scelta1)
coordinate_1_1= coordinate_scelta1[which(coordinate_scelta1[3]==1),c(1,2)]
dim(coordinate_1_1) #1311 gruppo 1
coordinate_2_1= coordinate_scelta1[which(coordinate_scelta1[3]==2),c(1,2)]
dim(coordinate_2_1) #38
coordinate_3_1= coordinate_scelta1[which(coordinate_scelta1[3]==3),c(1,2)]
dim(coordinate_3_1) #143
coordinate_4_1= coordinate_scelta1[which(coordinate_scelta1[3]==4),c(1,2)]
dim(coordinate_4_1) #186
coordinate_5_1= coordinate_scelta1[which(coordinate_scelta1[3]==5),c(1,2)]
dim(coordinate_5_1) #95
coordinate_6_1= coordinate_scelta1[which(coordinate_scelta1[3]==6),c(1,2)]
dim(coordinate_6_1) #34

# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data = coordinate_1_1, aes(x = lon, y = lat), color = "blue")+
  geom_point(data = coordinate_2_1, aes(x = lon, y = lat), color = "red")+
  geom_point(data = coordinate_3_1, aes(x = lon, y = lat), color = "yellow")+
  geom_point(data = coordinate_4_1, aes(x = lon, y = lat), color = "green")+
  geom_point(data = coordinate_5_1, aes(x = lon, y = lat), color = "purple")+
  geom_point(data = coordinate_6_1, aes(x = lon, y = lat), color = "orange")

# Scelta2: metodo del legame completo con 10 gruppi
II_completo[9]
SI_completo[9]

scelta2= cutree(completo,k=10)
coordinate_scelta2=cbind(coordinate_scuole,scelta2)
coordinate_1_2=coordinate_scelta2[which(coordinate_scelta2[3]==1),c(1,2)]
dim(coordinate_1_2) #519
coordinate_2_2=coordinate_scelta2[which(coordinate_scelta2[3]==2),c(1,2)]
dim(coordinate_2_2) #221
coordinate_3_2=coordinate_scelta2[which(coordinate_scelta2[3]==3),c(1,2)]
dim(coordinate_3_2) #408
coordinate_4_2=coordinate_scelta2[which(coordinate_scelta2[3]==4),c(1,2)]
dim(coordinate_4_2) #191
coordinate_5_2=coordinate_scelta2[which(coordinate_scelta2[3]==5),c(1,2)]
dim(coordinate_5_2) #146

```

```

coordinate_6_2=coordinate_scelta2[which(coordinate_scelta2[3]==6),c(1,2)]
dim(coordinate_6_2) #55
coordinate_7_2=coordinate_scelta2[which(coordinate_scelta2[3]==7),c(1,2)]
dim(coordinate_7_2) #137
coordinate_8_2=coordinate_scelta2[which(coordinate_scelta2[3]==8),c(1,2)]
dim(coordinate_8_2) #72
coordinate_9_2=coordinate_scelta2[which(coordinate_scelta2[3]==9),c(1,2)]
dim(coordinate_9_2) #34
coordinate_10_2=coordinate_scelta2[which(coordinate_scelta2[3]==10),c(1,2)]
dim(coordinate_10_2) #24

# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_2, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_2, aes(x=lon, y=lat), color="green")+
  geom_point(data=coordinate_3_2, aes(x=lon, y=lat), color="purple")+
  geom_point(data=coordinate_4_2, aes(x=lon, y=lat), color="yellow")+
  geom_point(data=coordinate_5_2, aes(x=lon, y=lat), color="red")+
  geom_point(data=coordinate_6_2, aes(x=lon, y=lat), color="orange")+
  geom_point(data=coordinate_7_2, aes(x=lon, y=lat), color="cyan")+
  geom_point(data=coordinate_8_2, aes(x=lon, y=lat), color="black")+
  geom_point(data=coordinate_9_2, aes(x=lon, y=lat), color="darkgreen")+
  geom_point(data=coordinate_10_2, aes(x=lon, y=lat), color="violet")

# Attraction-Repulsion clustering
# SCELTA 1: medio con 6 gruppi
dataset=as.data.frame(dataset)
# Matrice delle distanze euclidee tra gli attributi sensibili
CD=daisy(dataset[,4:9])
CD_z=scale(dataset[,4:9])

# Delta 1
# Creo i parametri necessari
v= seq(from=0.1,to=10, by=0.1)
U= matrix(data=0,ncol=6,nrow=6)
U= as.matrix(U)
V= matrix(data = c(1,-1,-1,-1,-1,-1,
                  -1, 1,-1,-1,-1,-1,
                  -1,-1, 1,-1,-1,-1,
                  -1,-1,-1, 1,-1,-1,
                  -1,-1,-1,-1, 1,-1,
                  -1,-1,-1,-1,-1, 1),ncol=6,nrow=6,byrow = T)
#matrice ovvia in cui cerco di mixare tutte le etnie
V= as.matrix(V)

# Minimizzo l'Imbalance Index
imb_opt=Inf
sol_opt=NA
for(valore in v){
  params = list(U,valore,V)

```

```

funzione_D1=fairDistanceCalc(params,type = "additive1",
                             distances=as.matrix(distanze),
                             chargeMatrix = as.matrix(dataset[,4:9]),
                             Ks=6,totalPropMatrix = p_tot,
                             chargeDistance = as.matrix(CD))
clus_prop_D1= funzione_D1$averageHclust[[1]]$clusterProportions
addendi=c()
for(riga in 1:6){
  addendi=c(addendi,sqrt(sum((clus_prop_D1[riga,]-p_tot)^2)))
}
imb= sum(addendi)*(1/6)
if (imb<imb_opt){
  imb_opt=imb
  sol_opt=valore
}
}
# Con Delta1 si      ottenuta la seguente soluzione ottima
SOL_D1_1=sol_opt #v=7.2
IMB_D1_1=imb_opt #0.1344883

funzione_D1_1=fairDistanceCalc(list(U,SOL_D1-1,V),type = "additive1",
                                 distances=as.matrix(distanze),
                                 chargeMatrix = as.matrix(dataset[,4:9]),
                                 Ks=6,totalPropMatrix = p_tot,
                                 chargeDistance = as.matrix(CD))
scelta1_D1=funzione_D1_1$clusterAverageHclust
SI_D1_1= summary(silhouette(scelta1_D1[[1]],dist=distanze))$avg.width #0.43
coordinate_scelta1_D1= cbind(coordinate_scuole,scelta1_D1)

coordinate_1_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==1),
c(1,2)]
dim(coordinate_1_D1_1) #1311 gruppo 1
coordinate_2_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==2),
c(1,2)]
dim(coordinate_2_D1_1) #38
coordinate_3_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==3),
c(1,2)]
dim(coordinate_3_D1_1) #164
coordinate_4_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==4),
c(1,2)]
dim(coordinate_4_D1_1) #165
coordinate_5_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==5),
c(1,2)]
dim(coordinate_5_D1_1) #95
coordinate_6_D1_1= coordinate_scelta1_D1[which(coordinate_scelta1_D1[3]==6),
c(1,2)]
dim(coordinate_6_D1_1) #34
# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_D1_1, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_D1_1, aes(x=lon, y=lat), color="red")+

```

```

geom_point(data=coordinate_3_D1_1, aes(x=lon, y=lat), color="yellow")+
geom_point(data=coordinate_4_D1_1, aes(x=lon, y=lat), color="green")+
geom_point(data=coordinate_5_D1_1, aes(x=lon, y=lat), color="purple")+
geom_point(data=coordinate_6_D1_1, aes(x=lon, y=lat), color="orange")

#TENTATIVO CON CD STANDARDIZZATA
v= seq(from=0.1,to=10, by=0.1)
U= matrix(data=0, ncol=6, nrow=6)
U= as.matrix(U)
V= matrix(data = c(1,-1,-1,-1,-1,-1,
                  -1, 1,-1,-1,-1,-1,
                  -1,-1, 1,-1,-1,-1,
                  -1,-1,-1, 1,-1,-1,
                  -1,-1,-1,-1, 1,-1,
                  -1,-1,-1,-1,-1, 1), ncol=6, nrow=6, byrow = T)
V= as.matrix(V)

# Minimizzo Imbalance Index
imb_opt_z=Inf
sol_opt_z=NA
for(valore in v){
  params = list(U,valore,V)
  funzione_D1=fairDistanceCalc(params,type = "additive1",
                                 distances=as.matrix(distanze),
                                 chargeMatrix = as.matrix(dataset[,4:9]),
                                 Ks=6, totalPropMatrix = p_tot,
                                 chargeDistance = as.matrix(CD_z) )
  clus_prop_D1= funzione_D1$averageHclust[[1]]$clusterProportions
  addendi=c()
  for(riga in 1:6){
    addendi=c(addendi,sqrt(sum((clus_prop_D1[riga,]-p_tot)^2)))
  }
  imb= sum(addendi)*(1/6)
  if (imb<imb_opt_z){
    imb_opt_z=imb
    sol_opt_z=valore
  }
  print(valore)
}
imb_opt_z
sol_opt_z
# Ottengo gli stessi risultati
# Scelgo di lavorare con CD non standardizzata da ora in poi

distanze_bis=distanze
distanze_bis=as.matrix(distanze_bis)
for (riga in 1:1807){
  for (colonna in 1:1807){
    if (distanze_bis[riga,colonna]==0){
      if (riga!=colonna){
        distanze_bis[riga,colonna]=1
      }
    }
  }
}

```

```

        }
    }

# Delta 2
# per delta2 servono i parametri u e v
u= c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1,2,3,4,5,6,7,8,9,10)
v= c(0.1,0.3,0.5,0.7,0.9,1,3.25,5.5,7.75,10)

# Minimizzo Imbalance Index
imb_opt=Inf
sol_opt=NA
for(valore in v){
  for (valore2 in u){
    params = c(valore2,valore)
    funzione_D2=fairDistanceCalc(params,type = "multiplicative",
                                   distances=as.matrix(distanze_bis),
                                   chargeMatrix = as.matrix(dataset[,4:9]),
                                   Ks=6, totalPropMatrix = p_tot,
                                   chargeDistance = as.matrix(CD))
    clus_prop_D2= funzione_D2$averageHclust[[1]]$clusterProportions
    addendi=c()
    for(riga in 1:6){
      addendi=c(addendi,sqrt(sum((clus_prop_D2[riga,]-p_tot)^2)))
    }
    imb= sum(addendi)*(1/6)
    if (imb<imb_opt){
      imb_opt=imb
      sol_opt=c(valore2,valore)
    }
  }
}

# Con Delta2 si     ottenuta la seguente soluzione ottima
SOL_D2_1=sol_opt #u=0.1   v=0.1
IMB_D2_1=imb_opt #0.1390062

funzione_D2_1=fairDistanceCalc(c(0.1,0.1),type = "multiplicative",
                                 distances=as.matrix(distanze_bis),
                                 chargeMatrix = as.matrix(dataset[,4:9]),
                                 Ks=6, totalPropMatrix = p_tot,
                                 chargeDistance = as.matrix(CD))
scelta1_D2=funzione_D2_1$clusterAverageHclust
SI_D2_1= summary(silhouette(scelta1_D2[[1]],dist=distanze_prova))$avg.width
coordinate_scelta1_D2= cbind(coordinate_scuole,scelta1_D2)

coordinate_1_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==1),
c(1,2)]
dim(coordinate_1_D2_1) #1284 gruppo 1
coordinate_2_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==2),
c(1,2)]
dim(coordinate_2_D2_1) #38

```

```

coordinate_3_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==3),
c(1,2)]
dim(coordinate_3_D2_1) #164
coordinate_4_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==4),
c(1,2)]
dim(coordinate_4_D2_1) #165
coordinate_5_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==5),
c(1,2)]
dim(coordinate_5_D2_1) #122
coordinate_6_D2_1= coordinate_scelta1_D2[which(coordinate_scelta1_D2[3]==6),
c(1,2)]
dim(coordinate_6_D2_1) #34

# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_D2_1, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_D2_1, aes(x=lon, y=lat), color="red")+
  geom_point(data=coordinate_3_D2_1, aes(x=lon, y=lat), color="yellow")+
  geom_point(data=coordinate_4_D2_1, aes(x=lon, y=lat), color="green")+
  geom_point(data=coordinate_5_D2_1, aes(x=lon, y=lat), color="purple")+
  geom_point(data=coordinate_6_D2_1, aes(x=lon, y=lat), color="orange")

# Delta 4
# Definisco i parametri
u= seq(0.1,1,by=0.2)
v= c(0.1, 1, 10)
w= c(0.1, 0.5, 0.9, 1, 5.5, 10)
V= matrix(data = c(1,-1,-1,-1,-1,-1,
                   -1, 1,-1,-1,-1,-1,
                   -1,-1, 1,-1,-1,-1,
                   -1,-1,-1, 1,-1,-1,
                   -1,-1,-1,-1, 1,-1,
                   -1,-1,-1,-1,-1, 1), ncol=6, nrow=6, byrow = T)
V= as.matrix(V)

imb_opt=Inf
sol_opt=NA
for(valore in v){
  for (valore2 in u){
    for (valore3 in w){
      params = list(valore2,valore,valore3,V)
      funzione_D4=fairDistanceCalc(params,type = "local",
                                    distances=as.matrix(distanze_bis),
                                    chargeMatrix = as.matrix(dataset[,4:9]),
                                    Ks=6,totalPropMatrix = p_tot,
                                    chargeDistance = as.matrix(CD))
      clus_prop_D4= funzione_D4$averageHclust[[1]]$clusterProportions
      addendi=c()
      for(riga in 1:6){
        addendi=c(addendi,sqrt(sum((clus_prop_D4[riga,]-p_tot)^2)))
      }
    }
  }
}

```

```

    }
    imb= sum(addendi)*(1/6)
    if (imb<imb_opt){
      imb_opt=imb
      sol_opt=c(valore2,valore,valore3)
    }
  }
}

# Con Delta4 si     ottenuta la seguente soluzione ottima
SOL_D4_1=sol_opt # u=0.1, v=0.1, w=0.1
IMB_D4_1=imb_opt # 0.12

funzione_D4_1=fairDistanceCalc(list(0.1,0.1,0.1,V),type = "local",
                                distances=as.matrix(distanze_prova),
                                chargeMatrix = as.matrix(dataset[,4:9]),
                                Ks=6, totalPropMatrix = p_tot,
                                chargeDistance = as.matrix(CD))
scelta1_D4=funzione_D4_1$clusterAverageHclust
SI_D4_1= summary(silhouette(scelta1_D4[[1]],dist=distanze_prova))$avg.width
coordinate_scelta1_D4= cbind(coordinate_scuole,scelta1_D4)

coordinate_1_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==1),
c(1,2)]
dim(coordinate_1_D4_1) #1311 gruppo 1
coordinate_2_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==2),
c(1,2)]
dim(coordinate_2_D4_1) #38
coordinate_3_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==3),
c(1,2)]
dim(coordinate_3_D4_1) #143
coordinate_4_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==4),
c(1,2)]
dim(coordinate_4_D4_1) #186
coordinate_5_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==5),
c(1,2)]
dim(coordinate_5_D4_1) #95
coordinate_6_D4_1= coordinate_scelta1_D4[which(coordinate_scelta1_D4[3]==6),
c(1,2)]
dim(coordinate_6_D4_1) #34

#il totale torna a 1807
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_D4_1, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_D4_1, aes(x=lon, y=lat), color="red")+
  geom_point(data=coordinate_3_D4_1, aes(x=lon, y=lat), color="yellow")+
  geom_point(data=coordinate_4_D4_1, aes(x=lon, y=lat), color="green")+
  geom_point(data=coordinate_5_D4_1, aes(x=lon, y=lat), color="purple")+
  geom_point(data=coordinate_6_D4_1, aes(x=lon, y=lat), color="orange")

```

```

# Confronto tra le partizioni
adjustedRandIndex(cutree(medio,k=6),scelta1_D1[[1]]) # 0.99
adjustedRandIndex(cutree(medio,k=6),scelta1_D2[[1]]) # 0.95
adjustedRandIndex(cutree(medio,k=6),scelta1_D4[[1]]) # 1

# Ripeto per la scelta2: completo con 10 gruppi
# Delta1
v= seq(from=0.1,to=10, by=0.1)
U= matrix(data=0,ncol=6,nrow=6)
U= as.matrix(U)
V= matrix(data = c(1,-1,-1,-1,-1,-1,
                  -1, 1,-1,-1,-1,-1,
                  -1,-1, 1,-1,-1,-1,
                  -1,-1,-1, 1,-1,-1,
                  -1,-1,-1,-1, 1,-1,
                  -1,-1,-1,-1,-1, 1),ncol=6,nrow=6,byrow = T)
#matrice ovvia in cui cerco di mixare tutte le etnie
V= as.matrix(V)

imb_opt=Inf
sol_opt=NA
for(valore in v){
  params = list(U,valore,V)
  funzione_D1=fairDistanceCalc(params,type = "additive1",
                                distances=as.matrix(distanze),
                                chargeMatrix = as.matrix(dataset[,4:9]),
                                Ks=10,totalPropMatrix = p_tot,
                                chargeDistance = as.matrix(CD))
  clus_prop_D1= funzione_D1$completeHclust[[1]]$clusterProportions
  addendi=c()
  for(riga in 1:10){
    addendi=c(addendi,sqrt(sum((clus_prop_D1[riga,]-p_tot)^2)))
  }
  imb= sum(addendi)*(1/10)
  if (imb<imb_opt){
    imb_opt=imb
    sol_opt=valore
  }
}
# Con Delta1 si ottenuta la seguente soluzione ottima
SOL_D1_2=sol_opt #v=8.3
IMB_D1_2=imb_opt #0.14

funzione_D1_2=fairDistanceCalc(list(U,SOL_D1_2,V),type = "additive1",
                                distances=as.matrix(distanze),
                                chargeMatrix = as.matrix(dataset[,4:9]),
                                Ks=10,totalPropMatrix = p_tot,
                                chargeDistance = as.matrix(CD))
scelta2_D1=funzione_D1_2$clusterCompleteHclust
SI_D1_2= summary(silhouette(scelta2_D1[[1]],dist=distanze))$avg.width
coordinate_scelta2_D1= cbind(coordinate_scuole,scelta2_D1)

```

```

coordinate_1_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==1),
c(1,2)]
dim(coordinate_1_D1_2) #519 gruppo 1
coordinate_2_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==2),
c(1,2)]
dim(coordinate_2_D1_2) #239
coordinate_3_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==3),
c(1,2)]
dim(coordinate_3_D1_2) #589
coordinate_4_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==4),
c(1,2)]
dim(coordinate_4_D1_2) #55
coordinate_5_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==5),
c(1,2)]
dim(coordinate_5_D1_2) #128
coordinate_6_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==6),
c(1,2)]
dim(coordinate_6_D1_2) #137
coordinate_7_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==7),
c(1,2)]
dim(coordinate_7_D1_2) #85
coordinate_8_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==8),
c(1,2)]
dim(coordinate_8_D1_2) #24
coordinate_9_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==9),
c(1,2)]
dim(coordinate_9_D1_2) #19
coordinate_10_D1_2= coordinate_scelta2_D1[which(coordinate_scelta2_D1[3]==10),
c(1,2)]
dim(coordinate_10_D1_2) #12

# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_D1_2, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_D1_2, aes(x=lon, y=lat), color="green")+
  geom_point(data=coordinate_3_D1_2, aes(x=lon, y=lat), color="purple")+
  geom_point(data=coordinate_4_D1_2, aes(x=lon, y=lat), color="orange")+
  geom_point(data=coordinate_5_D1_2, aes(x=lon, y=lat), color="yellow")+
  geom_point(data=coordinate_6_D1_2, aes(x=lon, y=lat), color="cyan")+
  geom_point(data=coordinate_7_D1_2, aes(x=lon, y=lat), color="red")+
  geom_point(data=coordinate_8_D1_2, aes(x=lon, y=lat), color="violet")+
  geom_point(data=coordinate_9_D1_2, aes(x=lon, y=lat), color="darkgreen")+
  geom_point(data=coordinate_10_D1_2, aes(x=lon, y=lat), color="black")

# Delta2
u= c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1,2,3,4,5,6,7,8,9,10)
v= c(0.1,0.3,0.5,0.7,0.9,1,3.25,5.5,7.75,10)

imb_opt=Inf
sol_opt=NA

```

```

for(valore in v){
  for (valore2 in u){
    params = c(valore2,valore)
    funzione_D2=fairDistanceCalc(params,type = "multiplicative",
                                  distances=as.matrix(distanze_prova),
                                  chargeMatrix = as.matrix(dataset[,4:9]),
                                  Ks=10, totalPropMatrix = p_tot,
                                  chargeDistance = as.matrix(CD))
    clus_prop_D2= funzione_D2$completeHclust[[1]]$clusterProportions
    addendi=c()
    for(riga in 1:10){
      addendi=c(addendi,sqrt(sum((clus_prop_D2[riga,]-p_tot)^2)))
    }
    imb= sum(addendi)*(1/10)
    if (imb<imb_opt){
      imb_opt=imb
      sol_opt=c(valore2,valore)
    }
  }
}

# Con Delta2 si     ottenuta la seguente soluzione ottima
SOL_D2_2=sol_opt # u=1, v=0.1
IMB_D2_2=imb_opt # 0.15

funzione_D2_2=fairDistanceCalc(SOL_D2_2,type = "multiplicative",
                               distances=as.matrix(distanze_prova),
                               chargeMatrix = as.matrix(dataset[,4:9]),
                               Ks=10, totalPropMatrix = p_tot,
                               chargeDistance = as.matrix(CD))
scelta2_D2=funzione_D2_2$clusterCompleteHclust
SI_D2_2= summary(silhouette(scelta2_D2[[1]],dist=distanze_prova))$avg.width
coordinate_scelta2_D2= cbind(coordinate_scuole,scelta2_D2)

coordinate_1_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==1),
c(1,2)]
dim(coordinate_1_D2_2) #647
coordinate_2_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==2),
c(1,2)]
dim(coordinate_2_D2_2) #179
coordinate_3_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==3),
c(1,2)]
dim(coordinate_3_D2_2) #487
coordinate_4_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==4),
c(1,2)]
dim(coordinate_4_D2_2) #55
coordinate_5_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==5),
c(1,2)]
dim(coordinate_5_D2_2) #172
coordinate_6_D2_2= coordinate_scelta2_D2[which(coordinate_scelta2_D2[3]==6),
c(1,2)]

```



```

            chargeMatrix = as.matrix(dataset[,4:9]),
            Ks=10, totalPropMatrix = p_tot,
            chargeDistance = as.matrix(CD))
clus_prop_D4= funzione_D4$completeHclust[[1]]$clusterProportions
addendi=c()
for(riga in 1:10){
  addendi=c(addendi,sqrt(sum((clus_prop_D4[riga,]-p_tot)^2)))
}
imb= sum(addendi)*(1/10)
print(imb)
if (imb<imb_opt){
  imb_opt=imb
  sol_opt=c(valore2,valore,valore3)
}
print(c(valore2,valore,valore3))
}
}

# Con Delta4 si ottenuta la seguente soluzione ottima
SOL_D4_2=sol_opt #u=0.1, v=0.1, w=0.1
IMB_D4_2=imb_opt #0.16

funzione_D4_2=fairDistanceCalc(list(...,V),type = "local",
                                 distances=as.matrix(distanze_prova),
                                 chargeMatrix = as.matrix(dataset[,4:9]),
                                 Ks=10, totalPropMatrix = p_tot,
                                 chargeDistance = as.matrix(CD))
scelta2_D4=funzione_D4_2$clusterCompleteHclust
SI_D4_2= summary(silhouette(scelta2_D4[[1]],dist=distanze_prova))$avg.width
coordinate_scelta2_D4= cbind(coordinate_scuole,scelta2_D4)

coordinate_1_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==1),
c(1,2)]
dim(coordinate_1_D4_2) #519
coordinate_2_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==2),
c(1,2)]
dim(coordinate_2_D4_2) #221
coordinate_3_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==3),
c(1,2)]
dim(coordinate_3_D4_2) #408
coordinate_4_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==4),
c(1,2)]
dim(coordinate_4_D4_2) #191
coordinate_5_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==5),
c(1,2)]
dim(coordinate_5_D4_2) #146
coordinate_6_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==6),
c(1,2)]
dim(coordinate_6_D4_2) #55
coordinate_7_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==7),
c(1,2)]

```

```

dim(coordinate_7_D2_2) #137
coordinate_8_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==8),
c(1,2)]
dim(coordinate_8_D2_2) #72
coordinate_9_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==9),
c(1,2)]
dim(coordinate_9_D4_2) #34
coordinate_10_D4_2= coordinate_scelta2_D4[which(coordinate_scelta2_D4[3]==10),
c(1,2)]
dim(coordinate_10_D4_2) #24

# Rappresentazione geografica
ggmap(get_map(c(left = -73.5, bottom = 41.2, right = -69.9, top = 43),
source = "stamen"))+
  geom_point(data=coordinate_1_D4_2, aes(x=lon, y=lat), color="blue")+
  geom_point(data=coordinate_2_D4_2, aes(x=lon, y=lat), color="green")+
  geom_point(data=coordinate_3_D4_2, aes(x=lon, y=lat), color="purple")+
  geom_point(data=coordinate_4_D4_2, aes(x=lon, y=lat), color="yellow")+
  geom_point(data=coordinate_5_D4_2, aes(x=lon, y=lat), color="red")+
  geom_point(data=coordinate_6_D4_2, aes(x=lon, y=lat), color="orange")+
  geom_point(data=coordinate_7_D4_2, aes(x=lon, y=lat), color="cyan")+
  geom_point(data=coordinate_8_D4_2, aes(x=lon, y=lat), color="black")+
  geom_point(data=coordinate_9_D4_2, aes(x=lon, y=lat), color="darkgreen")+
  geom_point(data=coordinate_10_D4_2, aes(x=lon, y=lat), color="violet")

# Confronto tra le partizioni
adjustedRandIndex(cutree(completo,k=10),scelta2_D1[[1]]) # 0.79
adjustedRandIndex(cutree(completo,k=10),scelta2_D2[[1]]) # 0.61
adjustedRandIndex(cutree(completo,k=10),scelta2_D4[[1]]) # 1

```