Invididuazione di community di utenti per il turismo utilizzando il clustering congiuntamente al collaborative filtering

**ABSTRACT**

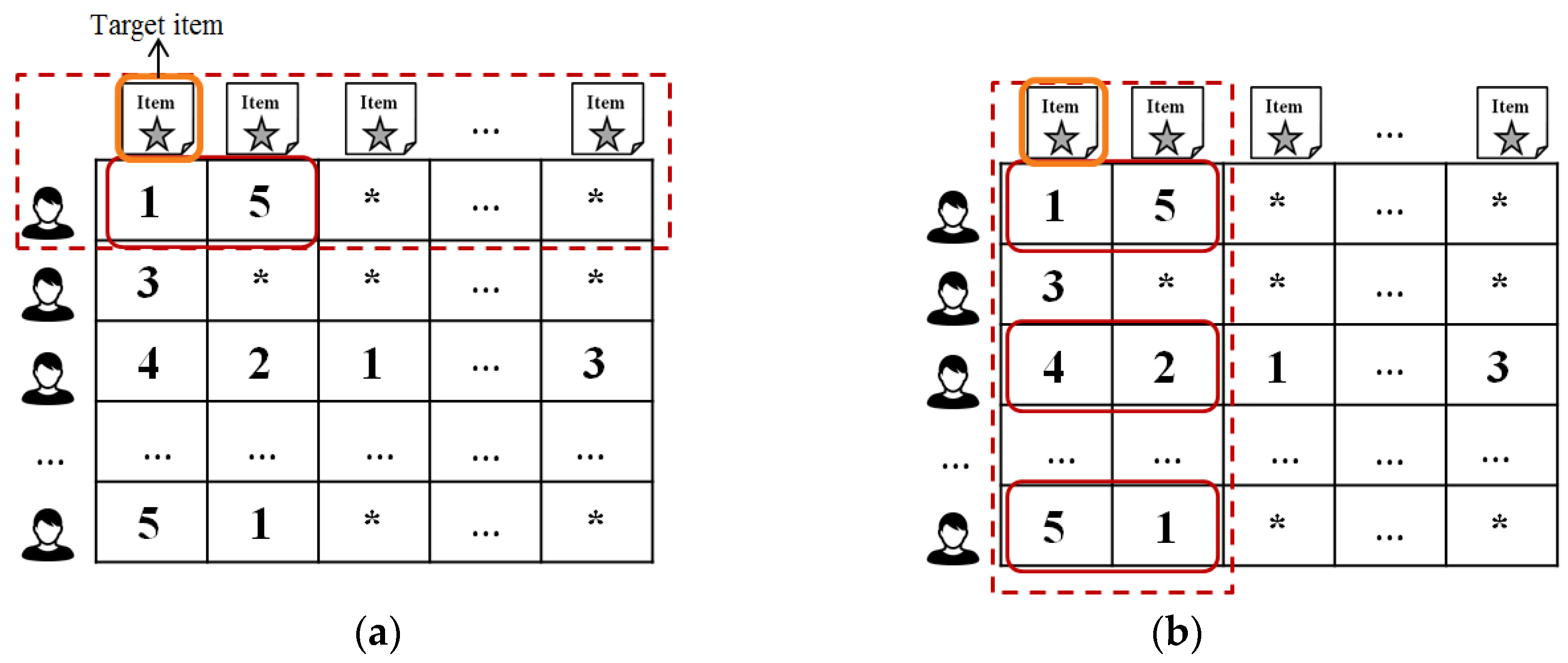
Il turismo è uno dei settori più sviluppati in Italia e uno tra quelli con maggior tasso di crescita, tuttavia molte persone decidono di non partire perché non riescono ad organizzarsi con persone nella propria cerchia di conoscenze, il sistema che proponiamo in questo articolo ha quindi lo scopo di cercare di risolvere il problema per queste persone individuando all’interno del web community di persone con un account Google con gusti simili con l’intenzione di partire cercando di sfruttare il potenziale inespresso di questa cerchia di persone, tuttavia molti utenti non esprimono valutazioni per ogni tipo di attrazione, il sistema tenta quindi di individuare delle community anche dopo aver predetto le valutazioni inespresse e confrontando i risultati con quelli ottenuti non facendo alcuna predizione. Il sistema fa uso di un dataset composto da 5455 utenti con una media di valutazione per ogni tipologia di attrazione e si compone di varie fasi: preprocessamento dei dati, analisi dei dati, individuazione e addestramento del modello predittivo per i voti inespressi, individuazione di community con e senza predizione, visualizzazione mediante grafici dei risultati con e senza predizione dei voti inespressi di ciascun utente, confronto e analisi dei risultati con entrambe le metodologie.

1. Introduzione

Molte persone scelgono di partire con degli sconosciuti sia per mancanza di organizzazione con le persone conosciute, sia per fare nuove conoscenze, tuttavia molti sistemi già presenti non sfruttano a pieno i gusti delle persone basandosi sulle votazioni già effettuate per ciascun luogo e soprattutto non sfruttano a pieno il potenziale che può avere un sistema di raccomandazione per trovare community in modo più preciso. Il Sistema si compone di varie fasi: preprocessamento dei dati eliminando utenti e tipologie di attrazioni per cui non esiste un numero sufficiente di votazioni, analisi dei dati graficandoli, normalizzandoli in quanto non tutti gli utenti esprimono votazioni allo stesso modo, riducendo la dimensionalità del problema individuando delle correlazioni tra le dimensioni (le categorie di attrazioni in questo caso) ed eliminandole, individuazione e addestramento del modello predittivo per i voti inespressi cercando una metrica che misuri bene il grado di similarità tra utenti e items, individuazione di community con e senza predizione, visualizzazione mediante grafici dei risultati con e senza predizione dei voti inespressi di ciascun utente, confronto e analisi dei risultati con entrambe le metodologie utilizzando diversi metodi per l’individuazione di community (clusters).

1. Metodo di ricerca

Un sistema di raccomandazione basato su collaborative filtering è un sistema che tenta di fare delle previsioni sulle votazioni inespresse sulla base dei k utenti o items più simili, con alla base l’idea che utenti che hanno avuto fino ad ora gusti simili su rating già avvenuti, avranno probabilmente gusti simili anche su rating futuri, ciò è alla base dei principali sistemi di raccomandazione di film, musica, etc come Amazon, Netflix etc. Ci sono due tipi di collaborative filtering, uno trovando i K utenti più simili e facendo una media pesata in base alla similarità delle loro valutazioni per l’item non valutato dall’utente per cui si sta facendo una predizione, questo però ha una serie di problemi, come la complessità computazionale, il fatto che la matrice item-user sia una matrice sparsa in quanto ogni utente vota soltanto pochi item etc, oppure trovando gli item più simili e facendo una media pesata sempre in base alla similarità delle votazioni già effettuate dall’utente per i k item più simili (fig a).



Prima di tutto è necessario rappresentare la relazione tra utenti e item sotto forma di una matrice dove le righe rappresentano gli utenti e le colonne gli item, dopo di che si sceglie una metrica per misurare la similarità tra user o tra item, è stato visto empiricamente che per la similarità tra user spesso la metrica migliore è la correlazione di Pearson, mentre tra gli item è la coseno similarità:



Dove :

* zi è l’user/item iesimo
* zj e l’user/item jesimo
* ru,a è il rating dato dall’utente u all’item a
* ru,b è il rating dato dall’utente u all’item b

Mentre la predizione pesata del rating Rp dell’utente u viene effettuata come segue:



Per l’individuazione di community invece vengono provate diverse tecniche di clustering, per clustering si intende invece una tecnica di apprendimento non supervisionato in cui i dati non sono etichettati con lo scopo di trovare raggruppamenti tra elementi simili tra di loro, in questo articolo in particolare utilizziamo le seguenti tecniche: K-means con inizializzazione K-means ++, spectral clustering, clustering gerarchico (agglomerativo/divisivo).

* 1. TIPOLOGIA DI DATI:

Il dataset utilizzato si compone di 5455 utenti e 22 categorie, dove gli utenti rappresentano le righe, le categorie le colonne e una entry rappresenta su una scala continua da 1 a 5 la media delle valutazioni effettuate da quell’utente per quella specifica categoria di attrazione.

* 1. PREPROCESSAMENTO DATI:

La prima operazione da fare è la lettura del dataset dal file CSV in un DataFrame Pandas, successivamente eliminiamo gli utenti e le categorie con un numero di valutazioni inespresse (entry con 0.0). (fig1)

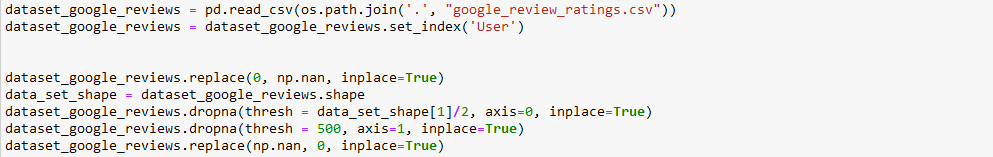


Fig 1

* 1. ANALISI DATI:

Come si evince dall’ istogramma Fig 2, la media delle votazioni per ogni categoria è molto diversa, pertanto utilizzeremo un min max upscaler per uniformare le votazioni di ogni scategoria su una scala comune nell’intevallo [0,1].

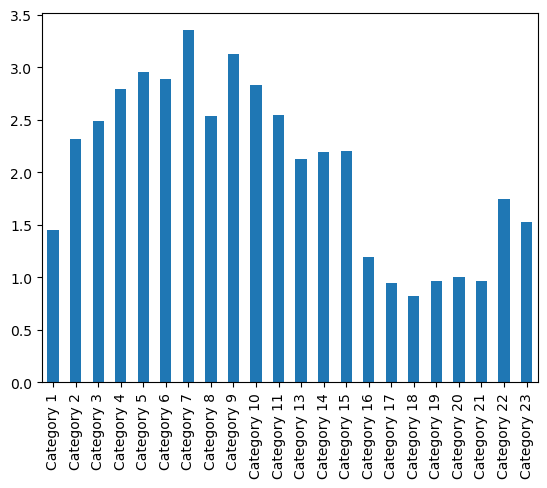
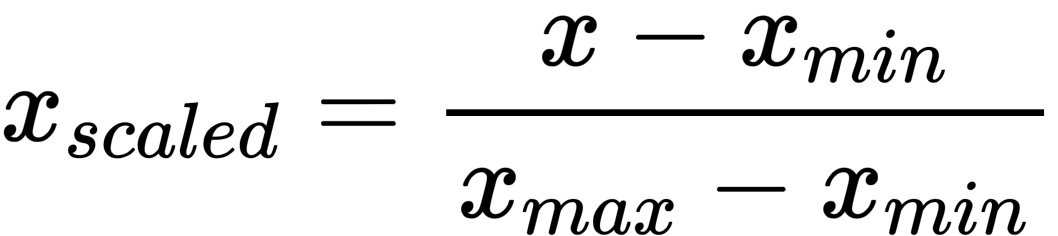


Fig2



Il prossimo step è quello di ridurre la dimensionalità del problema essendo il numero di dimensioni pari a 22, quindi un numero di dimensioni abbastanza grande, individuando delle correlazioni tra dimensioni ed eliminandole, lasciando soltanto le dimensioni più significative e a contenuto informativo più alto, a tale scopo grafichiamo la varianza per numero di componenti e scegliamo un numero di componenti per cui la varianza è almeno il 90% (Fig3).

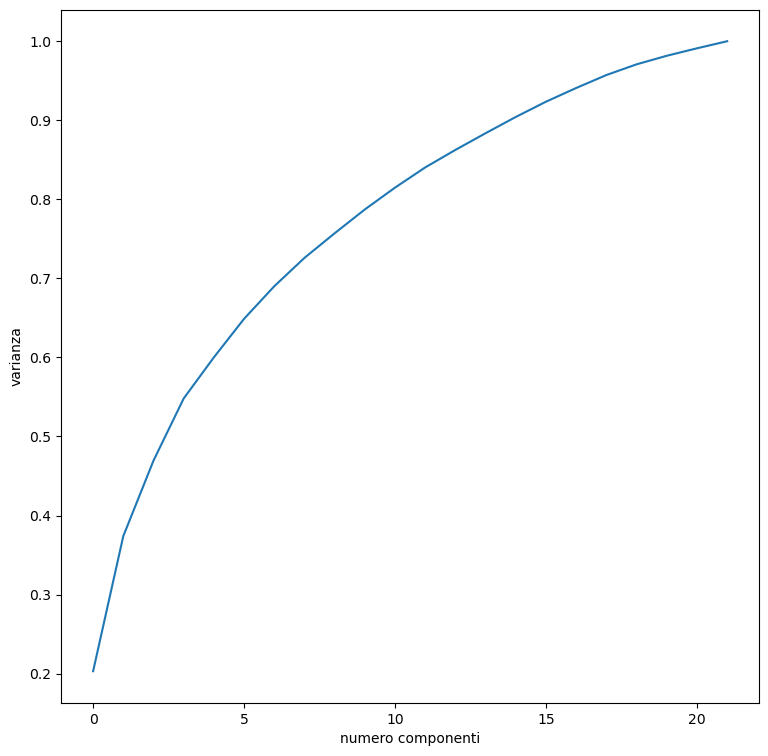
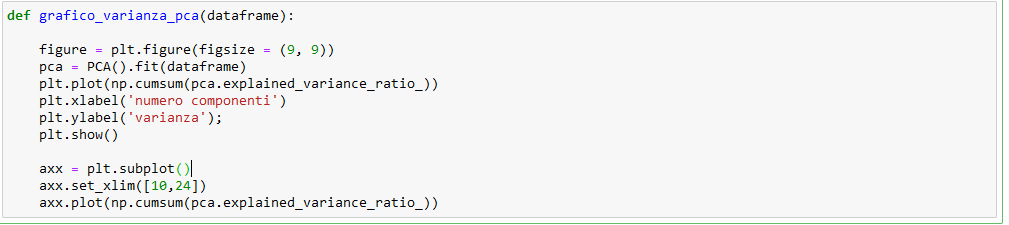


Fig3

Possiamo notare dal grafico che con un numero di componenti pari a 15, conserviamo il 90% della varianza, effettuiamo quindi una riduzione della dimensionalità basata sul metodo non supervisionato **PCA**, che nell’operazione di riduzione, effettua anche feature selection eliminando eventuali correlazioni.

Il prossimo step è quello di individuare il numero ottimale di cluster per i metodi che minimizzano la distanza dal cluster center come il k-means, a tal fine possiamo utilizzare il metodo/euristica elbow che ci dice di graficare sull’asse Y la distanza dal cluster center detta inerzia/inertia e sull’asse X il numero di cluster e prendere il valore delle ascisse per cui la curva passa repentinamente da una pendenza alta a una pendenza più bassa (Fig4)

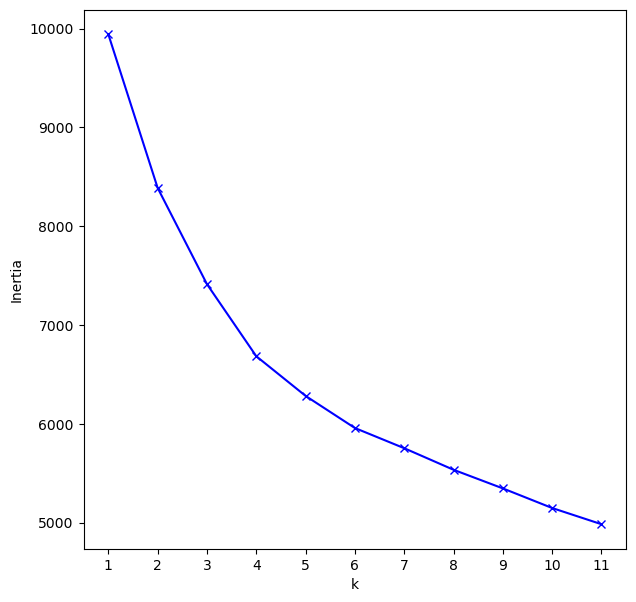
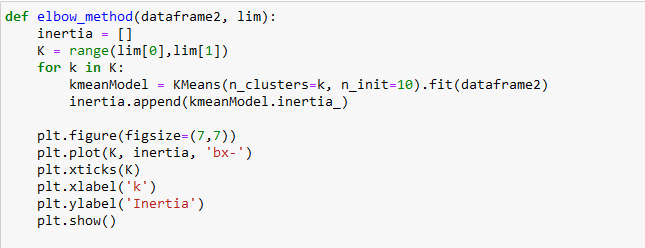


Fig4

Come si evince dal grafico, non c’è un elbow netto ed evidente, tuttavia comunque per un numero di cluster pari a 4, c’è comunque una netta riduzione della pendenza, pertanto decidiamo di utilizzare un numero di cluster pari a **4**.

* 1. INDIVIDUAZIONE E ADDESTRAMENTO MODELLO:

Il primo step per la creazione del modello è quello di scegliere tra l’approccio user o item based e la metrica per la misurazione della similarità, solitamente l’approccio user based ha diverse problematiche, come il fatto che la matrice user-item è una matrice sparsa e quindi le predizioni sono solitamente effettuate su un numero ridotto di item co-rated e possono quindi risultare imprecise e sensibili a successive votazioni, il problema della complessità computazionale in quanto il numero di utenti è solitamente molto grande, pertanto solitamente per fare predizioni si preferisce un approccio item-based ed è stato appurato empiricamente che la metrica migliore per valutare la distanza nel caso di approccio item-based è la coseno similarità, pre calcoliamo quindi prima di tutto la matrice di similarità con il metodo NearestNeighbors della libreria scikit-sklearn usando come metrica la coseno similarità e per ogni utente per cui esiste almeno 1 entry con un voto inespresso (Fig5), calcoliamo la predizione sulla base delle formule illustrate precedentemente (Fig6) e forniamo in input a diversi tipi di algoritmi di clustering, anche qui usando le implementazioni già fornite dalla libreria scikit-sklearn, sia il dataset originale che quello con le predizioni. Usiamo infine sempre il metodo PCA a 3 componenti per graficare i punti col rispettivo cluster, usando le librerie seaborn e matplotlib per i grafici.

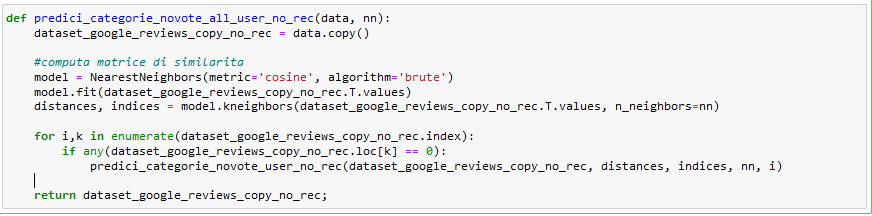


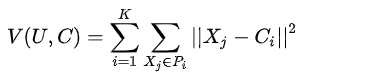
Fig5



Fig6

Per quanto riguarda il clustering invece utilizziamo vari tipi di clustering, il k-means con inizializzazione k-means ++,

il quale ha come obiettivo la minimizzazione del quadrato della norma della differenza di ciascun punto dal suo cluster center, ossia la funzione obiettivo:



Dove Ci è il cluster center del cluster iesimo e Xj è un punto qualsiasi appartenente al cluster i-esimo, l’inizializzazione k means ++ rende il k means un pò meno sensibile all’inizializzazione random dei centroidi scegliendoli con probabilità pesata dalla distanza di ciascun punto al cluster center più vicino.

Un altro tipo di tecnica di clustering usata è il clustering di tipo gerarchico, i quali hanno dietro l’idea o di partire da un unico cluster e poi suddividere (clustering divisivo) oppure partire da tanti cluster e tentare di accorparne quanti più possibile (clustering agglomerativo).

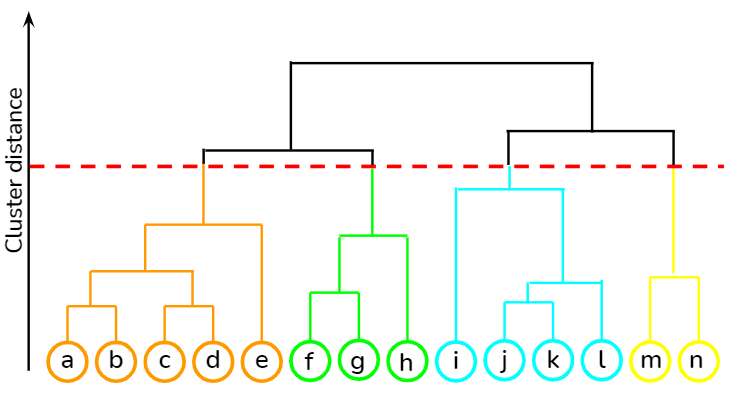
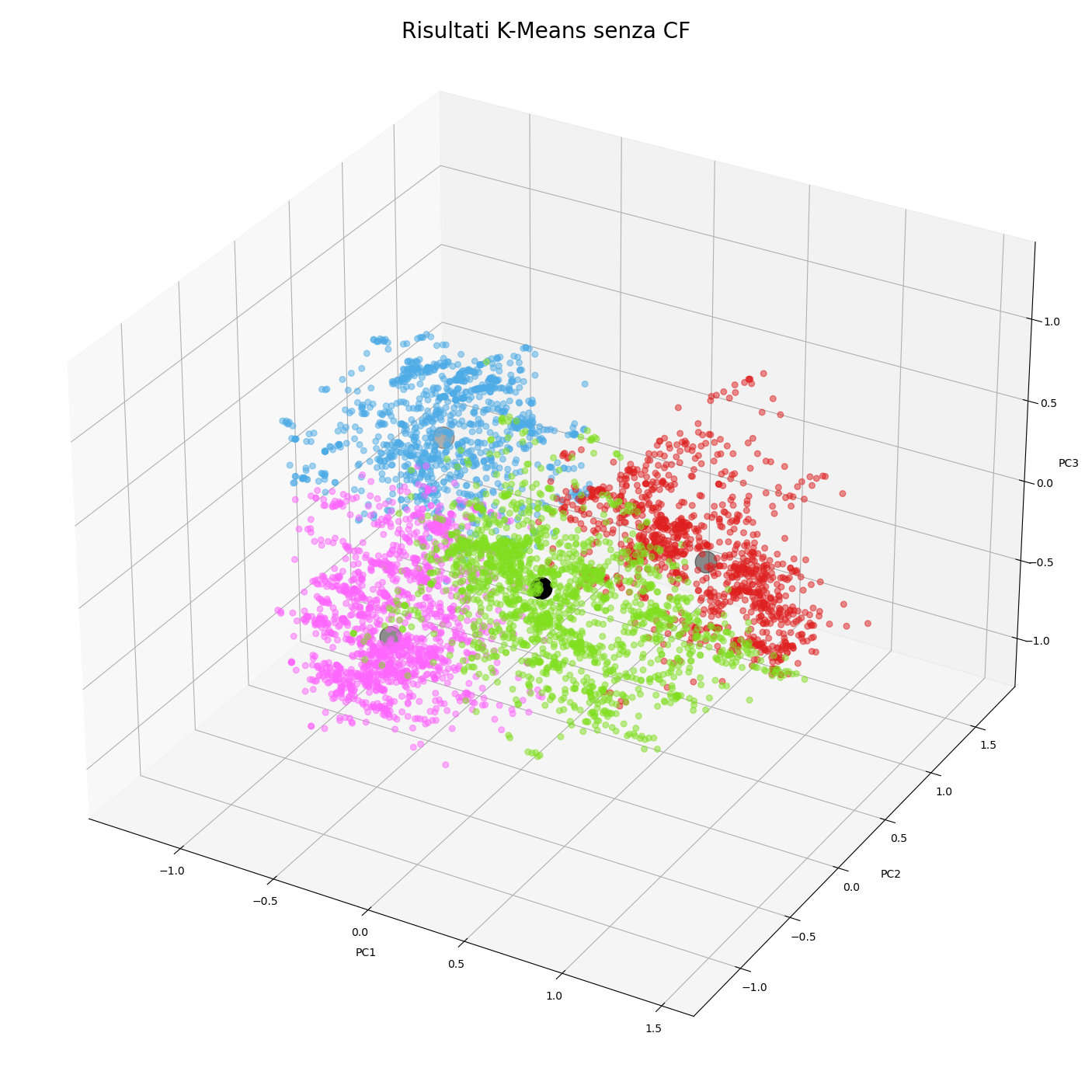


Fig7

Le ultime tecniche utilizzate sono lo spectral clustering, il quale fa una decomposizione spettrale della matrice di affinità passata come argomento (in questo caso la matrice di similarità costruita con la coseno similarity).

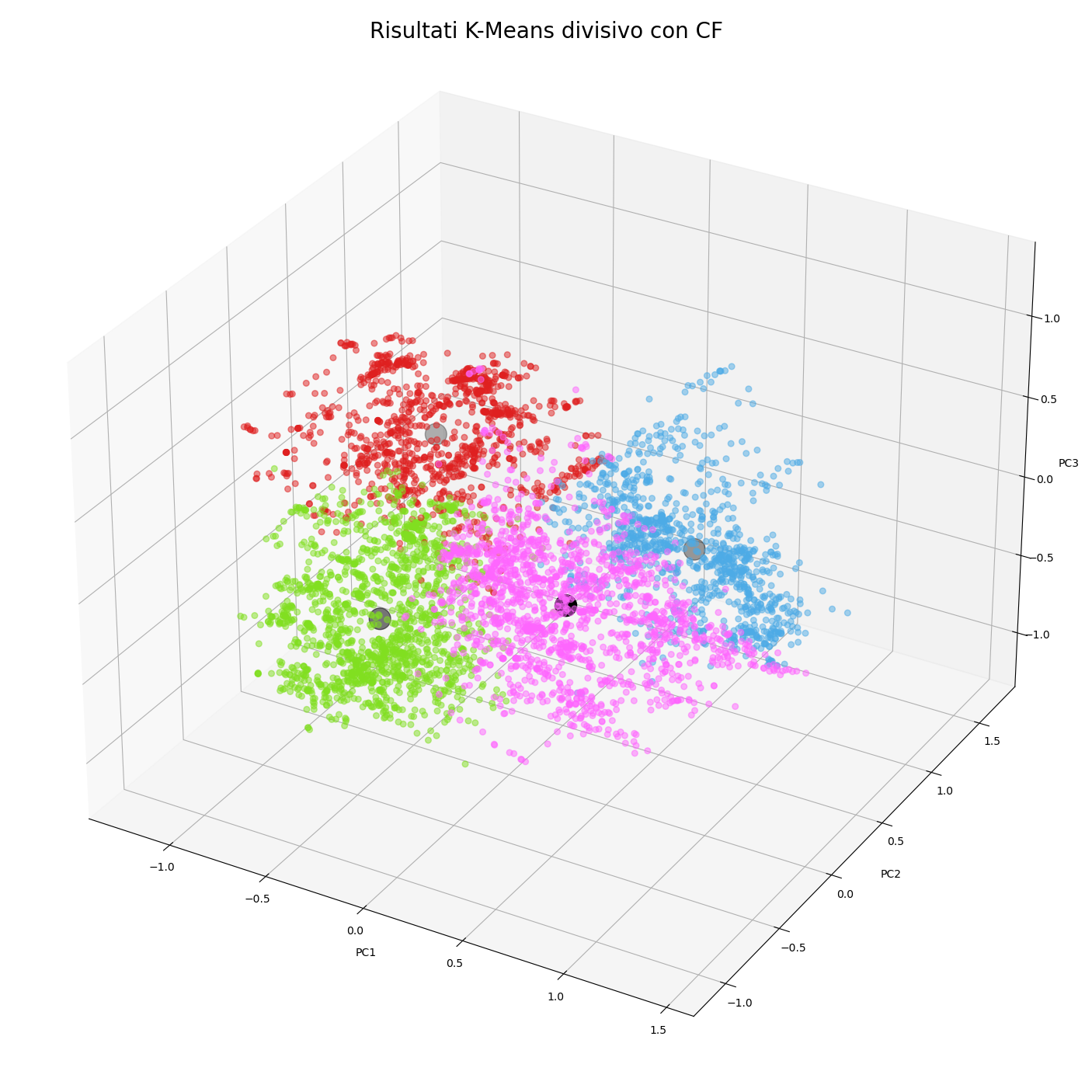
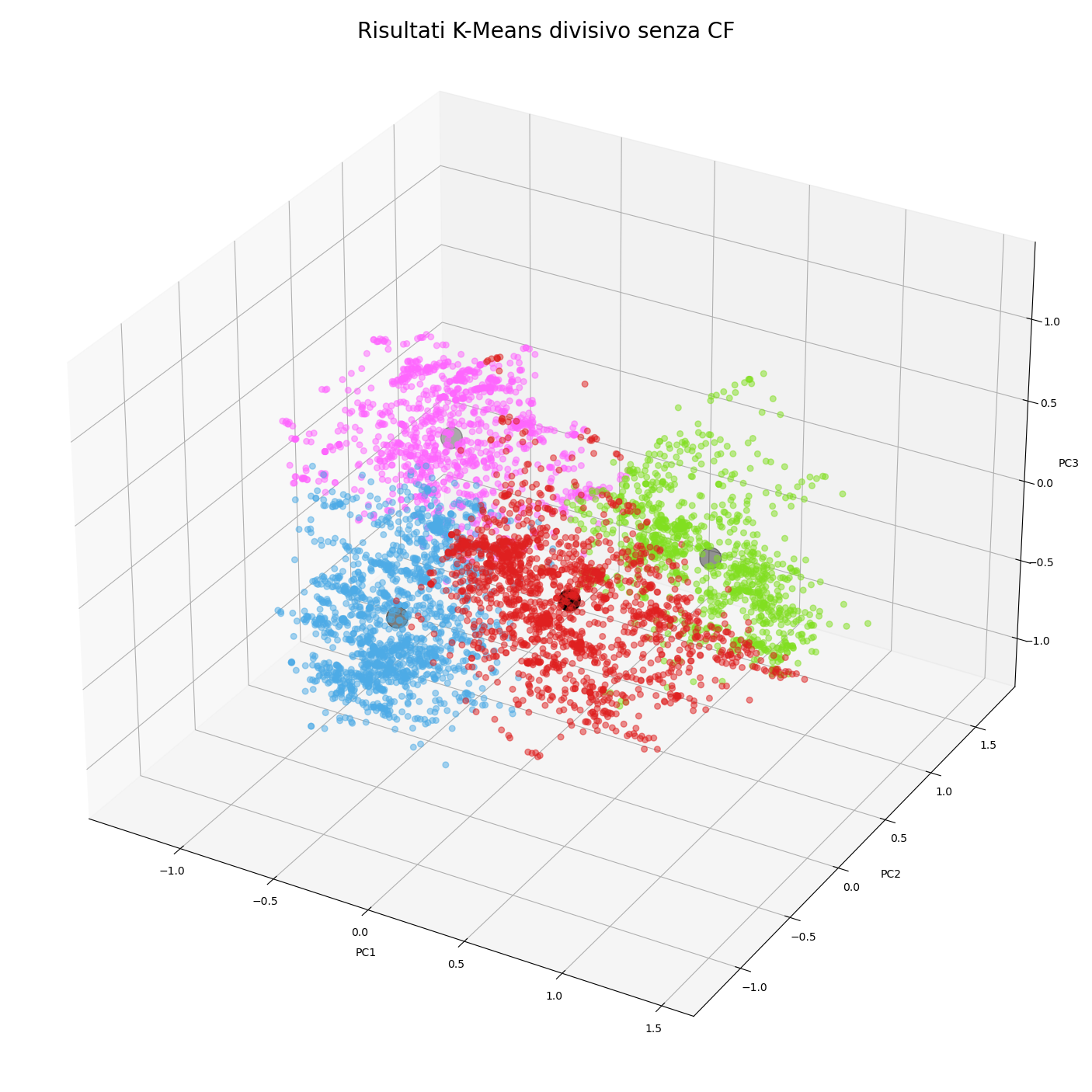
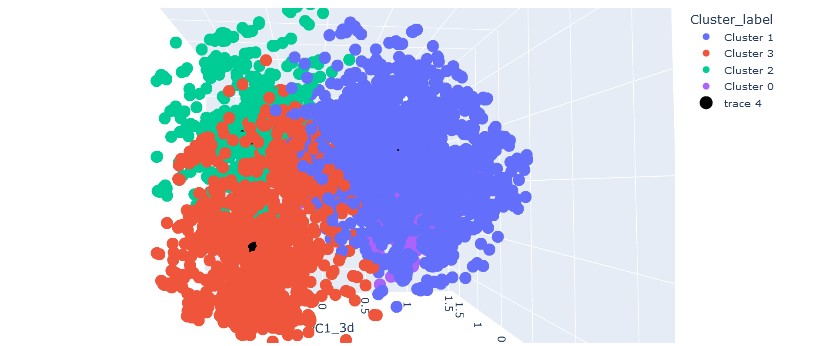
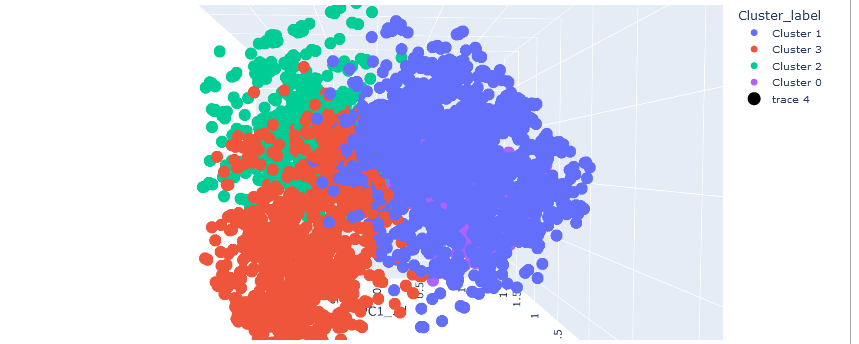
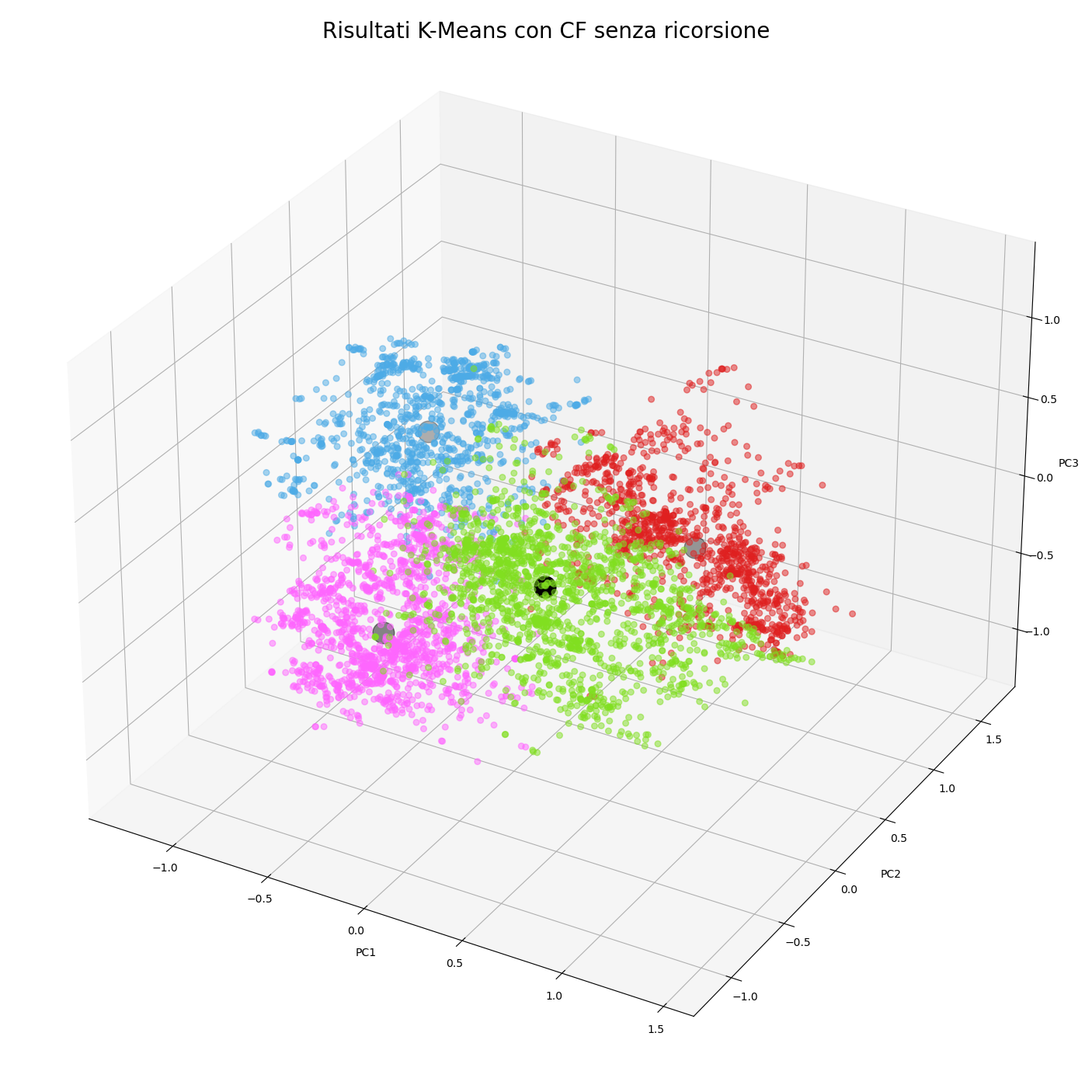
* 1. RISULTATI E DISCUSSIONI



..

.

……



CONCLUSIONI:

Possiamo notare una migliore classificazione dei punti nelle zone periferiche dei cluster, quelle chiaramente più soggette ad errore. Le tecniche che offrono migliori performance sono il KMeans e il clustering gerarchico di tipo divisivo (probabilmente perché il dataset è molto denso e quindi partendo da cluster grandi e decomponendo si ottengono buoni risultati in questi casi).

DATASET:

<https://www.kaggle.com/code/wirachleelakiatiwong/travel-review-analysis/input>

LIBRERIE USATE:

* Matplotlib
* plotly
* scikit-learn
* numpy
* pandas

CODICE DI TERZE PARTI USATO:

<https://towardsdatascience.com/item-based-collaborative-filtering-in-python-91f747200fab>

LINK SORGENTI (REPOSITORY GIT):

<https://github.com/Overkill95/progetto_ml_sii_2023>