MovieCounselor

Relazione Caso di Studio

Ingegneria della Conoscenza 2024/2025

Progetto a cura di:
Campanile Gabriele mat. 758278 g.campanile10@studenti.uniba.it
Lovero Nicola mat. 765733 n.lovero@studenti.uniba.it

Repository GitHub: https://github.com/Gabriele020/Gabriele020-Ingegneria_della_conoscenza_24_25.git

Indice

INTRODUZIONE

Il caso di studio ha come scopo l'analisi approfondita dei dati relativi ai film e alle serie TV disponibili su Netflix, focalizzandosi su aspetti come il genere, il cast e la regia. Il progetto si articola in tre fasi principali:

- **Preparazione dei dati**: In questa fase, i dati vengono trattati e trasformati in un formato utile per le analisi successive.
- **Base di conoscenza**: Consente all'utente di fare delle domande logiche e ottenere risposte automatiche, supportate da spiegazioni a livello di conoscenza.
- **Classificazione**: Qui viene selezionato il miglior modello di classificazione per prevedere il genere di un film fornito dall'utente.
- **Sistema di raccomandazione**: Propone film simili a quello indicato dall'utente, utilizzando metodi di clustering non supervisionato.

Per la realizzazione del progetto, il gruppo ha scelto Python come linguaggio di programmazione e GitHub come piattaforma di hosting, per i suoi ottimi strumenti di collaborazione, dove è ospitata la repository del progetto. Le librerie e le versioni utilizzate sono elencate nel file "cosaserve.txt". Le principali librerie impiegate sono:

- **scikit-learn**: libreria per tecniche di machine learning, utilizzata nelle fasi di classificazione e clustering.
- **kmodes**: libreria per il clustering usando l'algoritmo K-Modes, utilizzata nel clustering.
- **numpy**: libreria per operazioni su vettori e matrici, utilizzata in tutte le fasi del progetto.
- **matplotlib**: libreria per la visualizzazione di grafici, impiegata nelle fasi di classificazione e clustering.
- **Pandas**: libreria per la gestione e analisi dei dati, usata per il preprocessing.
- **fuzzywuzzy**: libreria per calcolare la somiglianza tra stringhe, utilizzata nel clustering.

PREPARAZIONE DEI DATI

I dataset utilizzati nel progetto derivano da Kaggle in formato CSV e sono riportati nel seguente elenco:

- Dataset film e serie TV Netflix (Netflix_serie_film.csv)
- Dataset valutazioni IMDB dei film (IMDb_valutazioni.csv)
- Dataset film Netflix (Netflix_film.csv)

Per rendere i dati idonei alle analisi successive, sono state eseguite le seguenti operazioni di preprocessing, tra cui:

- ❖ Rimozione delle colonne non pertinenti al progetto.
- ❖ Fusione dei tre dataset in un unico file finale.
- ❖ Eliminazione delle voci duplicate.
- ❖ Trasformazione dei valori nella colonna "genres" da categoriali a numerici utilizzando il metodo delle variabili dummy, per renderli pronti per l'imputazione successiva.
- ❖ Applicazione della stessa tecnica di conversione tramite label encoder alle colonne "year_range" e "title" per facilitare le operazioni successive di imputazione.
- ❖ Semplificazione della colonna "cast", mantenendo un solo attore per ciascun film.
- ❖ Uniformazione dei valori nella colonna "genres".
- ❖ Aggiunta del valore 'Movie' nella colonna "type" per le righe che presentavano valori nulli nel dataset relativo solo ai film.
- ❖ Normalizzazione dei valori nella colonna "ratings".
- ❖ Imputazione dei valori mancanti nella colonna "ratings" tramite l'uso del KNNImputer.
- ❖ Riduzione dei generi associati ad ogni film, mantenendo solo quello più frequentemente presente.
- ❖ Trasformazione dei valori nella colonna "type" da categorici a numerici usando il label encoder, per semplificare il processo di imputazione.
- ❖ Semplificazione dei valori nella colonna "country", mantenendo un unico paese per film e trasformandoli in numerici tramite il label encoder.
- ❖ Imputazione dei valori mancanti nella colonna "genre" utilizzando la tecnica di hot-deck imputation.
- ❖ Sostituzione della colonna "year" con "year_range", creando intervalli di anni.
- ❖ Eliminazione delle righe con valori mancanti per le quali non era possibile applicare l'imputazione.

2. BASE DI CONOSCENZA

Una base di conoscenza è un insieme di dati ordinati e organizzati che riflettono la comprensione di un determinato ambito. Tale conoscenza può includere informazioni come fatti, principi, concetti, relazioni e vincoli che descrivono il mondo reale o una sua specifica dimensione.
In sostanza, una base di conoscenza può essere vista come una raccolta di assiomi, ossia affermazioni che si considerano veritiere.
Nel progetto, la base di conoscenza viene utilizzata per abilitare un'interazione fluida tra l'utente e il sistema, consentendo uno scambio di domande e risposte focalizzate sul dominio di interesse, che in questo caso riguarda film e serie TV. Questo processo permette di ottenere risposte dettagliate e pertinenti sui vari aspetti relativi ai contenuti audiovisivi.
Concretamente, la base di conoscenza funge da una fonte di supporto che raccoglie e ordina una grande mole di informazioni sul settore cinematografico e televisivo. Gli utenti possono fare domande specifiche e il sistema è in grado di rispondere, elaborando le richieste e attingendo alle informazioni contenute nella base di conoscenza, per fornire risposte accurate e rilevanti.

- Un esempio di richiesta che l'utente può fare è la seguente: Verificare se il genere corrisponde al titolo di un film, utilizzando la funzione **askGenereDaTitolo**, che accetta in input entrambi i dati e restituisce una risposta affermativa o negativa. askGenereDaTitolo(titolo, genere) <=> titolo_genere;

Esempio di funzionamento **askGenereDaTitolo**(“titolo”,”genere”) ottimale:

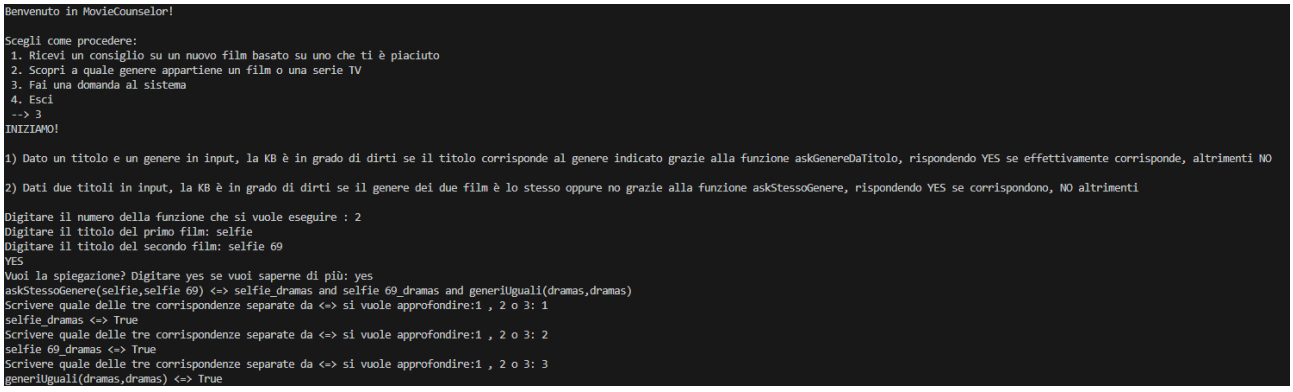
```
Benvenuto in MovieCounselor!

Scegli come procedere:
 1. Ricevi un consiglio su un nuovo film basato su uno che ti è piaciuto
 2. Scopri a quale genere appartiene un film o una serie TV
 3. Fai una domanda al sistema
 4. Esci
 --> 3
INIZIAMO!

1) Dato un titolo e un genere in input, la KB è in grado di dirti se il titolo corrisponde al genere indicato grazie alla funzione askGenereDaTitolo, rispondendo YES se effettivamente corrisponde, altrimenti NO

2) Dati due titoli in input, la KB è in grado di dirti se il genere dei due film è lo stesso oppure no grazie alla funzione askStessoGenere, rispondendo YES se corrispondono, NO altrimenti

Digitare il numero della funzione che si vuole eseguire : 1
Digitare il titolo del film: selfie
Digitare il genere del film: dramas
YES
Vuoi la spiegazione? Digitare yes se vuoi saperne di più: yes
askGenereDaTitolo(selfie,dramas) <=> selfie_dramas
selfie_dramas <=> True
```

Esempio di funzionamento **askGenereDaTitolo**(“titolo”,”genere”) non Ottimale:

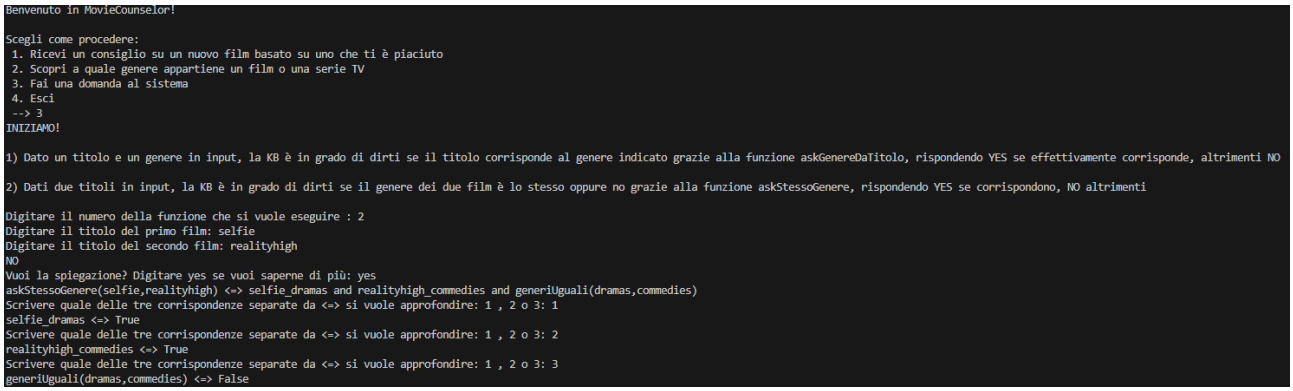
```
Benvenuto in MovieCounselor!

Scegli come procedere:
 1. Ricevi un consiglio su un nuovo film basato su uno che ti è piaciuto
 2. Scopri a quale genere appartiene un film o una serie TV
 3. Fai una domanda al sistema
 4. Esci
 --> 3
INIZIAMO!

1) Dato un titolo e un genere in input, la KB è in grado di dirti se il titolo corrisponde al genere indicato grazie alla funzione askGenereDaTitolo, rispondendo YES se effettivamente corrisponde, altrimenti NO

2) Dati due titoli in input, la KB è in grado di dirti se il genere dei due film è lo stesso oppure no grazie alla funzione askStessoGenere, rispondendo YES se corrispondono, NO altrimenti

Digitare il numero della funzione che si vuole eseguire : 1
Digitare il titolo del film: selfie
Digitare il genere del film: fantasy
NO
Vuoi la spiegazione? Digitare yes se vuoi saperne di più: yes
askGenereDaTitolo(selfie,fantasy) <=> selfie_fantasy
selfie_fantasy <=> False
```

- Verificare se film diversi appartengano ad uno stesso genere, mediante l'utilizzo della funzione **askStessoGenere**, che accetta in input i titoli dei film in questione; askStessoGenere(titolo1, titolo2) <=> titolo1_primoGenere and titolo2_secondoGenere and stessoGenere(primoGenere, secondoGenere), dove stessoGenere("genere1","genere2") indica se i generi presenti come parametri sono o meno uguali tra loro.

Esempio di funzionamento di askStessoGenere("titolo1","titolo2") caso ottimale:



Esempio di funzionamento di askStessoGenere("titolo1","titolo2") caso non ottimale:



Abbiamo adottato un approccio basato sulla spiegazione a livello di conoscenza, che permette a un sistema fondato sulla conoscenza di effettuare ragionamenti e giustificare le sue risposte.

Per ogni interrogazione, ossia ogni domanda posta per verificare se una proposizione derivi logicamente dalla base di conoscenza, il sistema risponderà con YES o NO, in base al tipo di clausola presentata.

In aggiunta, è possibile richiedere una spiegazione del risultato ottenuto tramite l'operatore how. Questo consente al sistema di fornire la logica sottostante alla risposta scelta, mostrando le clausole utilizzate per giungere a quella conclusione.

Infine, l'utente può anche domandare una prova per ciascun atomo presente nel corpo di una clausola.

3. CLASSIFICAZIONE

Uno degli obiettivi principali del Machine Learning è la classificazione, ovvero il compito di determinare la categoria a cui appartiene un nuovo elemento basandosi sulle informazioni ottenute da un set di dati di addestramento. Un sistema che svolge questa attività è chiamato classificatore. I classificatori costruiscono un modello a partire dai dati e successivamente lo utilizzano per assegnare nuove istanze a una categoria. Il processo di classificazione si articola in tre fasi principali: Addestramento, Valutazione dell'accuratezza e Applicazione del Modello. Per il nostro progetto, abbiamo deciso di suddividere i dati in un insieme di addestramento e un insieme di test, fissando quest'ultimo al 30% del totale. La classificazione è stata impiegata per il nostro caso di studio con l'obiettivo di prevedere il genere di un film fornito dall'utente. La variabile target su cui effettuare la previsione sarà quindi il "Genere". Per ottenere il risultato più accurato, sono stati confrontati tre diversi modelli di classificatori:

- KNN
- Random Forest
- Bagging

a. KNN

Uno degli algoritmi più noti nel campo del machine learning è il K-Nearest Neighbors (KNN). Questo metodo restituisce come risultato il genere a cui appartiene il film fornito in input, basando la decisione sulla maggioranza dei voti dei suoi vicini. In altre parole, la classe assegnata corrisponde a quella più frequente tra i k film più simili, identificati calcolando la somiglianza rispetto al film da classificare. Si tratta della tecnica più semplice da applicare, spesso efficace, ma caratterizzata da una lentezza e un elevato consumo di memoria, in quanto il costo computazionale cresce in maniera quadratica.

b. RANDOM FOREST L'RF

Il Random Forest Classifier è ampiamente impiegato per attività di classificazione, regressione e altre applicazioni, operando attraverso la costruzione di numerosi alberi decisionali. Per quanto riguarda la classificazione, l'output è determinato dalla classe scelta dalla maggioranza degli alberi. La foresta creata dall'algoritmo viene addestrata utilizzando tecniche di aggregazione come il bagging o il bootstrap. Il risultato finale viene stabilito combinando le predizioni dei singoli alberi decisionali. L'algoritmo effettua la previsione calcolando la media degli output dei diversi alberi e, aumentando il numero di questi ultimi, si ottiene una maggiore accuratezza. Il Random Forest

supera i limiti del Decision Tree, riducendo il problema dell'overfitting nei dataset e migliorando la precisione complessiva.

c. BAGGING

Il Bagging Classifier si basa sull'addestramento di più modelli dello stesso tipo, ciascuno su diversi sottoinsiemi casuali del dataset originale. Successivamente, combina le previsioni individuali (attraverso voto o media) per produrre una previsione finale. Ogni weak learner viene addestrato in parallelo utilizzando un set di addestramento generato casualmente, estraendo con sostituzione N campioni dal dataset originale (dove N corrisponde alla dimensione del dataset). I training set per ciascun classificatore di base sono tra loro indipendenti. Il bagging è particolarmente utile quando si vuole ridurre la varianza (overfitting) del modello, evitando che si ottenga un'elevata precisione sui dati di addestramento ma un alto tasso di errore sui dati di test. Tra i classificatori più comunemente utilizzati come base learner nel bagging ci sono gli alberi decisionali.

d. RISULTATI CLASSIFICATORI

Per analizzare le performance di ciascun classificatore, è stato effettuato un processo di tuning dei parametri. A questo scopo è stato utilizzato il metodo **GridSearchCV** della libreria *model_selection* del pacchetto **Sklearn** di Python. Di seguito vengono riportati i migliori parametri ottenuti per ogni classificatore:

- **KNN**: Migliori parametri -> {metric: 'manhattan', n_neighbors: 1, weights: 'uniform'}
- **Random Forest**: Migliori parametri -> {max_features: 'sqrt', n_estimators: 100}
- **Bagging**: Migliori parametri -> {n_estimators: 10}

Sono state effettuate più prove con diversi parametri per ogni classificatore per ottenere la corrispondenza migliore.

	precision	recall	f1-score	support
anime	0.96	1.00	0.98	555
cult	0.89	0.95	0.92	532
fantasy	1.00	1.00	1.00	537
action	0.91	1.00	0.95	560
documentary	0.95	1.00	0.97	553
nature	0.93	1.00	0.96	584
romantic	0.90	0.97	0.94	571
sport	0.95	1.00	0.97	520
thrillers	0.99	1.00	1.00	554
kids	0.93	1.00	0.97	538

Test effettuato con il classificatore KNN

	precision	recall	f1-score	support
anime	0.97	1.00	0.98	555
cult	0.90	0.95	0.92	532
fantasy	1.00	1.00	1.00	537
action	0.93	1.00	0.96	560
documentary	0.96	1.00	0.98	553
nature	0.92	1.00	0.95	584
romantic	0.90	0.97	0.93	571
sport	0.93	1.00	0.97	520
thrillers	0.99	1.00	1.00	554
kids	0.90	1.00	0.95	538
dramas	0.94	1.00	0.97	557
horror	0.82	0.83	0.83	568
standup	0.85	0.86	0.85	560
commedies	0.76	0.70	0.73	556
musical	0.68	0.30	0.41	562
accuracy			0.91	8307
macro avg	0.90	0.91	0.90	8307
weighted avg	0.90	0.91	0.89	8307

Test effettuato con il classificatore Random Forrest

	precision	recall	f1-score	support
anime	0.97	1.00	0.98	555
cult	0.87	0.94	0.90	532
fantasy	0.99	1.00	1.00	537
action	0.93	0.99	0.96	560
documentary	0.96	1.00	0.98	553
nature	0.92	0.99	0.96	584
romantic	0.91	0.97	0.94	571
sport	0.93	1.00	0.96	520
thrillers	0.98	1.00	0.99	554
kids	0.92	1.00	0.96	538
dramas	0.93	1.00	0.96	557
horror	0.82	0.82	0.82	568
standup	0.86	0.85	0.85	560
commedies	0.75	0.73	0.74	556
musical	0.63	0.27	0.37	562
accuracy			0.90	8307
macro avg	0.89	0.90	0.89	8307
weighted avg	0.89	0.90	0.89	8307

Test effettuato con il classificatore Bagging

L’esito di questo confronto ci ha portato a scegliere il Random Forest come classificatore per la predizione del genere.

Di seguito si riporta un esempio di funzionamento del classificatore:

```
Scegli come procedere:
 1. Ricevi un consiglio su un nuovo film basato su uno che ti è piaciuto
 2. Scopri a quale genere appartiene un film o una serie TV
 3. Fai una domanda al sistema
 4. Esci
 --> 2
INIZIAMO!
```

```
 Inserire il nome del film o serie TV che hai apprezzato: ghost
 ghost è un film? (s/n)
  -> s
 Inserire il paese di produzione:
  -> thailand
 Inserire l`anno di rilascio:
  -> 2016
 Inserire un membro del cast:
  -> scout taylor-compton
 Inserire un voto da 1 a 10 sul film/serie TV:
  -> 9
 Il genere del film o serie TV da te inserito è horror
```

4. Il CLUSTERING

Il clustering è una tecnica di apprendimento non supervisionato che permette di individuare e raggruppare elementi simili all'interno di dataset di grandi dimensioni, formando cluster, ossia gruppi di elementi accomunati da caratteristiche simili rispetto a punti di riferimento centrali, chiamati centroidi.

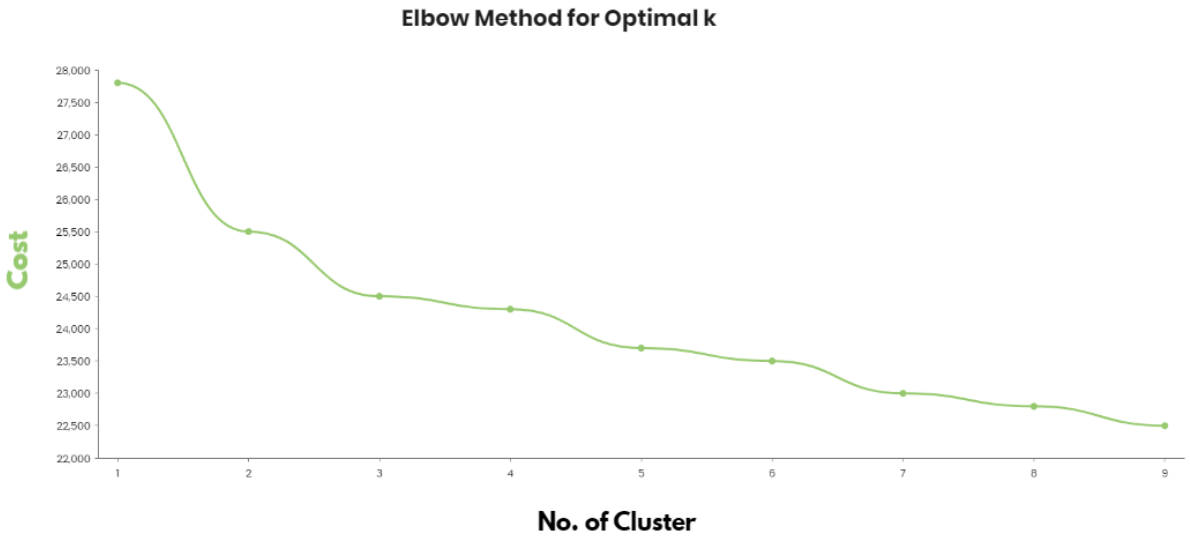
In questo contesto, abbiamo deciso di utilizzare questa metodologia per scoprire nuove correlazioni e somiglianze tra i dati che non siano legate esclusivamente al genere dei film considerati. Queste informazioni potranno poi essere sfruttate come base per la realizzazione di un sistema di raccomandazione.

e. CLUSTER

Data la significativa presenza di dati categorici nel dataset, abbiamo deciso di adottare l'algoritmo **K-Modes**. Questo algoritmo rappresenta un'estensione del K-Means, progettata per gestire dati categorici. Invece di utilizzare la media, si basa sulla **moda** e adotta un approccio basato sulla frequenza per minimizzare la funzione di costo.

Abbiamo deciso di individuare 3 cluster, o centroidi, utilizzando il **metodo del gomito**, un approccio empirico che consente di determinare il numero ottimale di cluster per un dataset all'interno di un intervallo predefinito.

In particolare, il range scelto è stato mantenuto al di sotto del numero totale di generi cinematografici presenti nel dataset, al fine di identificare correlazioni non strettamente dipendenti dal genere dei film.



f. RECOMMANDER SYSTEM

Per il sistema di raccomandazione, è stato scelto un approccio basato sui contenuti, confrontando gli attributi dei film e delle serie TV presenti nel dataset con quelli di un titolo apprezzato e fornito dall'utente.

Nello specifico, l'utente deve fornire alcune informazioni sul film da lui gradito, che vengono utilizzate per identificare il cluster più affine. In questo modo, si genera una lista di film consigliabili all'utente (una **top 10**) basata sulla similarità tra il film indicato e quelli appartenenti al cluster individuato come più simile.

Dopo il processo di clusterizzazione, per calcolare le similarità è stata impiegata la libreria **FuzzyWuzzy**, che si avvale della distanza di Levenshtein. Questa metrica misura la differenza tra due stringhe di caratteri, determinando il numero minimo di modifiche necessarie (inserimenti, eliminazioni o sostituzioni) per trasformare una stringa nell'altra.

Di seguito è riportato un esempio di utilizzo del Recommender system ed un esempio di esecuzione completa del programma:

```
Scegli come procedere:
 1. Ricevi un consiglio su un nuovo film basato su uno che ti è piaciuto
 2. Scopri a quale genere appartiene un film o una serie TV
 3. Fai una domanda al sistema
 4. Esci
 --> 1
INIZIAMO!

Immetti il nome del film o serie TV che ti e' piaciuto: breaking bad
breaking bad è un film? (s/n)
 -> n
immetti il paese di produzione:
 -> united states
Immetti l`anno di rilascio:
 -> 2015
Immetti un membro del cast:
 -> bryan cranstone
Dai un voto da 1 a 10 sul film/serie TV:
 -> 10
Inserisci il genere, scegliendo tra questi:
1 action
2 anime
3 commedies
4 cult
5 documentary
6 dramas
7 fantasy
8 horror
9 kids
10 musical
11 nature
12 romantic
13 sport
14 stand-up
15 thrillers
```

```
 ->1
Ti consigliamo di guardare:

big kill
the saint
batman: the killing joke
the 2nd
the taking of pelham 123
angel has fallen
american assassin
troy: the odyssey
extraction
in a valley of violence
Per dettagli sulle raccomandazioni restituite, digitare kb:
kb
Il cluster di appartenenza è il valore di choice: 2
Le metriche restituite tra tutti i cluster sono le seguenti: [407719, 914768, 147281]

Le singole metriche di similarità restituite per il cluster 2 sono:
       type                    title  similarity
787   movie                  big kill         384
5877  movie                 the saint         384
668   movie  batman: the killing joke         376
5191  movie                   the 2nd         373
5939  movie  the taking of pelham 123         366
453   movie           angel has fallen         365
395   movie          american assassin         364
6204  movie          troy: the odyssey         363
1766  movie                 extraction         363
2527  movie   in a valley of violence         361
```