

## Relazione Caso di Studio Ingegneria Della Conoscenza

## LINK GITHUB : https://github.com/manu0033/Icon.git

INTRODUZIONE

L’obiettivo di questo caso di studio è sfruttare in maniera approfondita le informazioni relative a film e serie TV presenti sulla piattaforma Netflix.  
Lo studio si propone di esaminare diversi aspetti dei contenuti disponibili, come il genere, il cast e la regia.  
In particolare, il progetto è articolato in quattro sezioni principali:

Preprocessing: si concentra sull’adattamento dei dati per renderli più idonei all’analisi successiva.

Base di conoscenza: consente all’utente di porre domande sulle conseguenze logiche e ricevere risposte generate dalla macchina, grazie all’impiego di spiegazioni a livello di conoscenza.

Classificazione: identifica il classificatore più efficace per prevedere il genere di un film fornito dall’utente.

Sistema di raccomandazione: propone all’utente film simili a quello indicato, sfruttando tecniche di clustering basate su apprendimento non supervisionato.

Strumenti utilizzati nel progetto:  
Il linguaggio scelto per lo sviluppo è Python .  
Come piattaforma di hosting è stato selezionato GitHub, dove risiede il repository del progetto.  
Tutte le librerie e le versioni impiegate sono elencate nel file “requirements.txt”.

In particolare:

Pandas: libreria utile per la gestione e l’analisi dei dati, impiegata nella fase di preprocessing.

scikit-learn: libreria per l’apprendimento automatico, utilizzata nelle sezioni di classificazione e clustering.

matplotlib: libreria per la realizzazione di grafici, impiegata nelle fasi di classificazione e clusterizzazione.

kmodes: libreria per il clustering tramite l’algoritmo K-Modes, utilizzata nella sezione di clusterizzazione.

fuzzywuzzy: libreria per calcolare la somiglianza tra stringhe, utilizzata nella fase di clustering.

numpy: libreria per operazioni su vettori e matrici, impiegata in tutte le fasi del progetto.

1. PREPROCESSING

I dataset utilizzati in questo caso di studio sono stati reperiti dal sito Kaggle in formato CSV e comprendono:

Netflix\_film.csv: dataset relativo ai film presenti su Netflix.

Netflix\_serie\_film.csv: dataset contenente sia film che serie TV di Netflix.

IMDb\_valutazioni.csv: dataset con le valutazioni dei film presenti su IMDb.

Per rendere i dati coerenti e pronti per le analisi successive, sono state eseguite diverse operazioni di preprocessing:

Unione dei dataset: i tre dataset sono stati combinati in un unico file finale.

Rimozione delle colonne non necessarie: sono state eliminate le informazioni superflue ai fini del progetto.

Eliminazione dei duplicati: per garantire l’unicità dei record.

Discretizzazione del campo year: trasformato in year\_range per facilitare l’analisi.

Riduzione dei generi: per ciascun film è stato mantenuto un solo genere, selezionato sulla base delle occorrenze più frequenti nel dataset.

Riduzione degli attori: nella colonna cast è stato mantenuto un solo attore per film.

Compilazione del campo type: per le righe mancanti provenienti dal dataset film è stato inserito il valore “Movie”.

Conversione dei generi in numerici: la colonna genres è stata trasformata in variabili dummy per facilitare le operazioni successive.

Ridenominazione dei valori in genres.

Conversione della colonna type in numerico: tramite Label Encoder, per prepararla all’imputation.

Conversione del campo year\_range in numerico: anch’esso tramite Label Encoder.

Conversione del campo title in numerico: con Label Encoder.

Riduzione e codifica del campo country: mantenendo un solo paese per film e trasformandolo in valori numerici tramite Label Encoder.

Imputation dei valori mancanti in ratings: effettuata tramite KNNImputer.

Standardizzazione dei valori di ratings.

Imputation dei valori mancanti in genres: eseguita mediante hot-deck imputation.

Eliminazione delle righe irrecuperabili: i record per cui non era possibile effettuare l’imputation sono stati rimossi.

2. BASE DI CONOSCENZA

Una base di conoscenza è un insieme strutturato e organizzato di informazioni che rappresentano la conoscenza relativa a un determinato ambito.  
Può includere fatti, regole, concetti, relazioni e vincoli che descrivono il mondo reale o una sua specifica parte.

In termini più tecnici, una base di conoscenza (KB) può essere vista come un insieme di assiomi, ossia proposizioni considerate vere.

Nel contesto di questo caso di studio, la base di conoscenza è stata utilizzata per facilitare un’interazione dinamica tra l’utente e il sistema, permettendo di porre domande e ricevere risposte pertinenti riguardo a film e serie TV.  
Grazie a questa struttura, gli utenti possono ottenere informazioni dettagliate su diversi aspetti dei contenuti presenti sulla piattaforma Netflix.

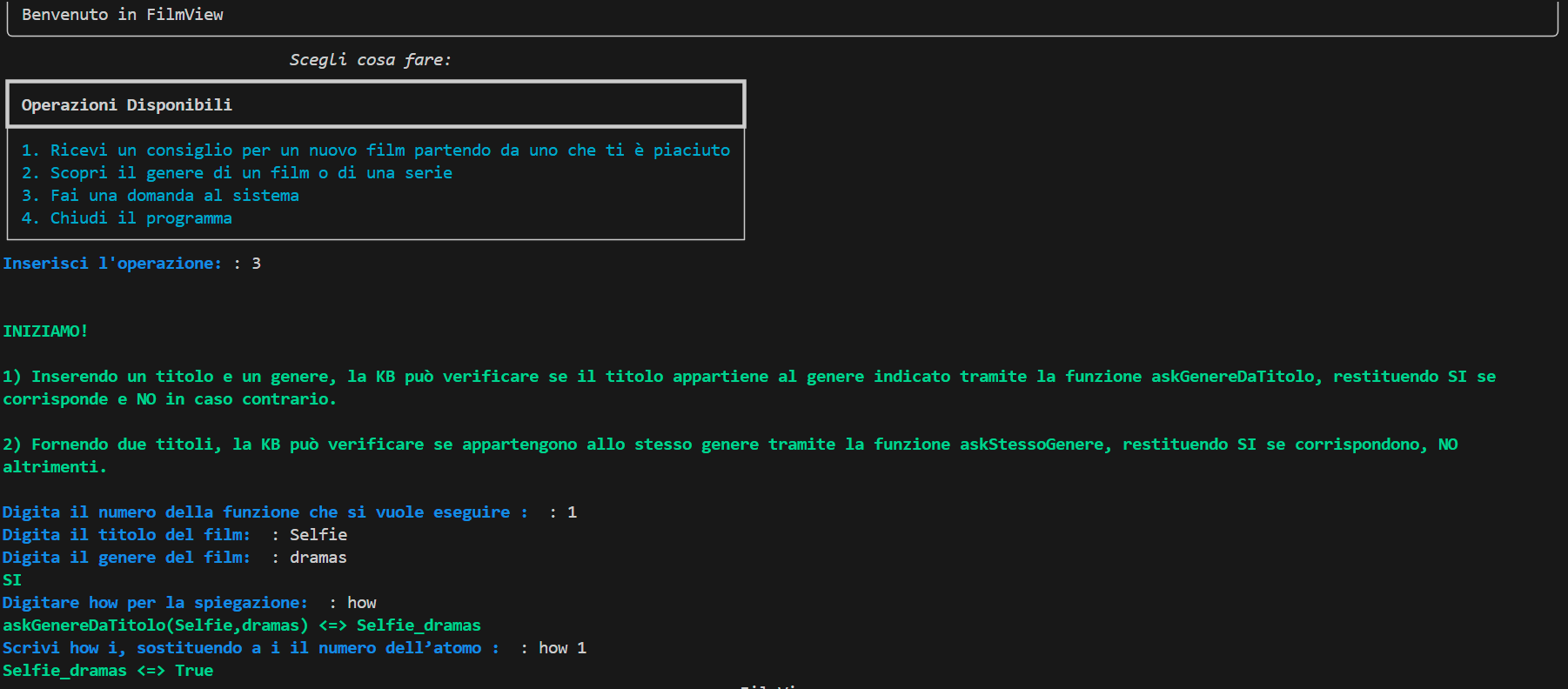
In pratica, la KB funge da supporto informativo, raccogliendo e organizzando un ampio insieme di dati relativi al mondo cinematografico e televisivo. Il sistema è in grado di elaborare le richieste dell’utente attingendo alle informazioni presenti nella KB, restituendo risposte precise e rilevanti.

Nello specifico, una delle funzionalità disponibili consente di:

Verificare la corrispondenza tra titolo e genere di un film: tramite la funzione askGenereDaTitolo, che prende in input il titolo e il genere e restituisce una risposta positiva o negativa.

askGenereDaTitolo(titolo, genere) <=> titolo\_genere

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



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

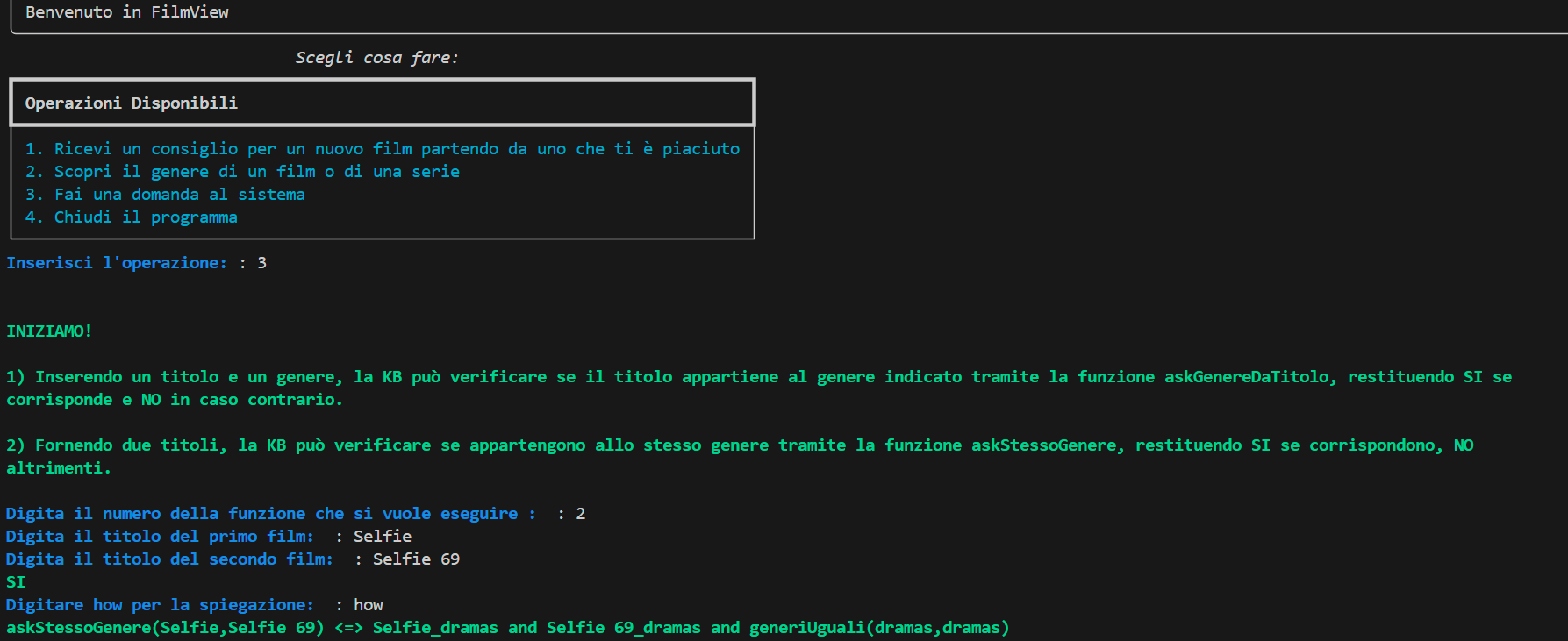


* + Verificare se film diversi appartengano ad uno stesso genere, mediante l’utilizzo della funzione **askStessoGenere**, che accetta in input i titoli dei film in questione;

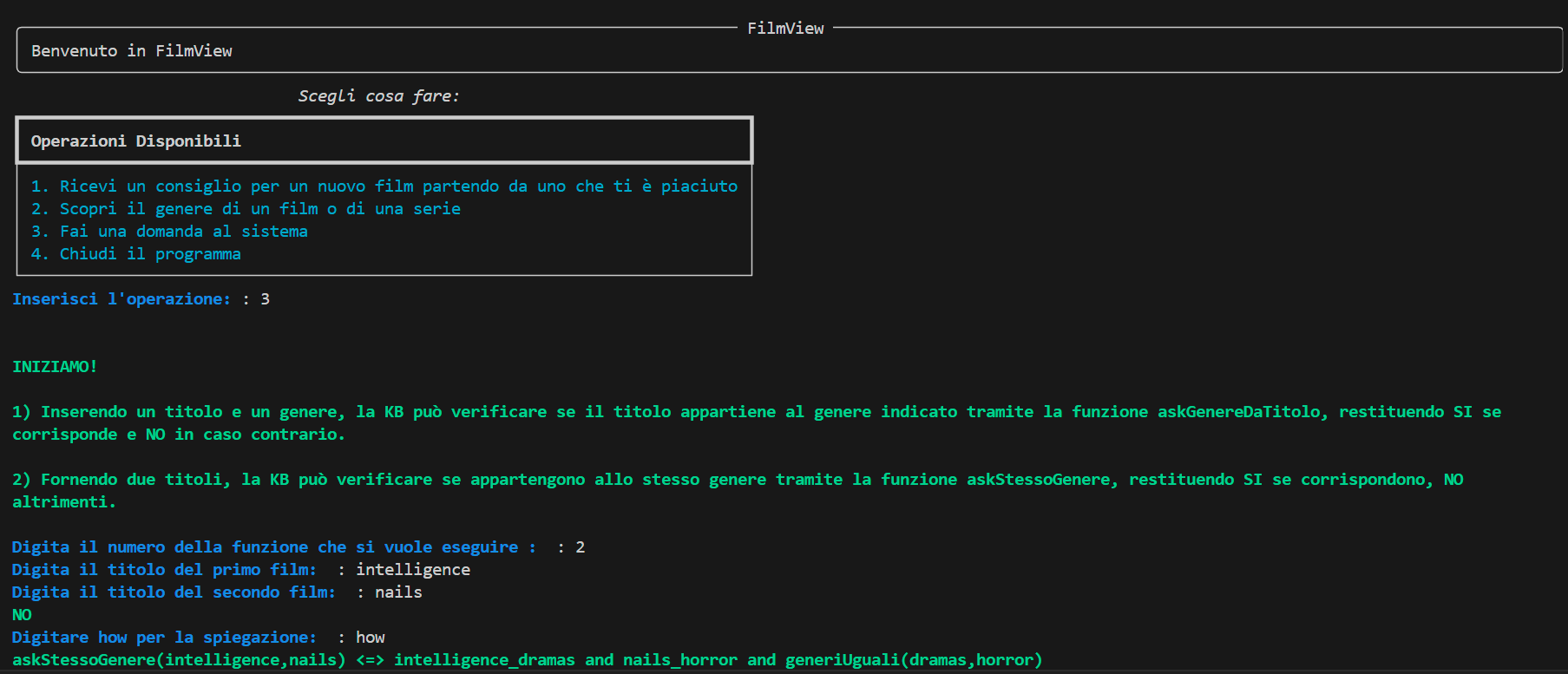
askStessoGenere(titolo1, titolo2) <=> titolo1\_primoGenere and titolo2\_secondoGenere and stessoGenere(primoGenere, secondoGenere),

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 utilizzato la spiegazione a livello di conoscenza che è un approccio che consente a un sistema basato su conoscenza di fornire ragionamenti e giustificazioni per le sue risposte.

Per ogni query che viene eseguita, ossia ogni interrogazione posta in modo tale da sapere se una proposizione sia conseguenza logica della base di conoscenza, la KB risponderà con SI oppure NO a seconda del tipo di clausola che le viene presentata.

Inoltre, si potrà chiedere la motivazione secondo la quale si è ottenuto un determinato risultato attraverso l’operatore **how** - in questo modo, la KB potrà fornire la motivazione alla base della restituzione di una certa risposta rispetto ad un’altra mostrando le clausole utilizzate per dedurre la risposta.

**3. CLASSIFICAZIONE**

Uno degli obiettivi principali del **Machine Learning** è la classificazione, ovvero il compito di determinare la classe di un nuovo elemento basandosi sulle informazioni apprese da un **dataset di addestramento.**  
Un sistema in grado di eseguire questa operazione è chiamato **classificatore**. I classificatori creano un modello dai dati disponibili, che poi viene utilizzato per classificare nuove istanze.  
Il processo di classificazione può essere suddiviso in tre fasi principali:

**Addestramento**

**Valutazione dell’accuratezza**

**Applicazione del modello**

Per questo progetto, i dati sono stati suddivisi in **training set e test set,** riservando il 30% dei dati a quest’ultimo.  
Lo scopo della classificazione nel caso di studio è predire il **genere di un film** fornito dall’utente. La variabile target sarà quindi il campo “Genere”.

Per ottenere le migliori performance, sono stati confrontati tre diversi classificatori:

**K-Nearest Neighbors (KNN)**

**Random Forest (RF)**

**Bagging Classifier**

**a. KNN**

Il **K-Nearest Neighbors (KNN)** è uno degli algoritmi più noti nel machine learning.  
Il classificatore assegna a un film il genere più comune tra i suoi **k vicini più simili,** calcolati in base alla somiglianza con il film dato in input.  
È un metodo semplice e spesso efficace, ma può risultare lento e richiedere molta memoria, poiché il costo computazionale cresce in modo quadratico con il numero di istanze.

**b. Random Forest**

Il **Random Forest Classifier (RF)** è ampiamente utilizzato per compiti di classificazione e regressione.  
L’algoritmo costruisce numerosi **alberi decisionali**, e per classificare un’istanza considera la classe più votata tra tutti gli alberi della foresta.  
La foresta viene generata utilizzando tecniche di **bagging** o **bootstrap**, che consistono nell’addestrare ciascun albero su un sottoinsieme casuale dei dati.  
Rispetto ai singoli decision tree, il Random Forest riduce l’**overfitting** e migliora la precisione complessiva.

**c. Bagging**

Il **Bagging Classifier** addestra più modelli dello stesso tipo su sottoinsiemi casuali del dataset originale, e poi aggrega le loro previsioni tramite voto o media.  
Ogni modello base (weak learner) viene addestrato in parallelo su un campione casuale estratto con sostituzione dal dataset originale.  
Questa tecnica è utile soprattutto per ridurre la **varianza,** evitando che un modello si adatti troppo ai dati di training a scapito delle prestazioni sui dati di test.  
Gli **alberi decisionali** sono spesso utilizzati come modelli base nel bagging.

**d. Risultati dei classificatori**

Per valutare le prestazioni di ciascun modello, è stato effettuato un **tuning dei parametri** tramite il metodo **GridSearchCV** della libreria sklearn.model\_selection.  
I migliori parametri ottenuti per ciascun classificatore sono stati:

**KNN:** {metric: 'manhattan', n\_neighbors: 1, weights: 'uniform'}

**Random Forest:** {max\_features: 'sqrt', n\_estimators: 100}

**Bagging:** {n\_estimators: 10}

Sono state condotte più prove con parametri differenti per ciascun classificatore, al fine di ottenere le performance ottimali.

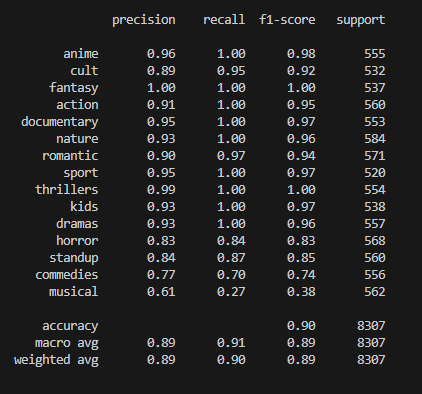


Figura 1: KNN

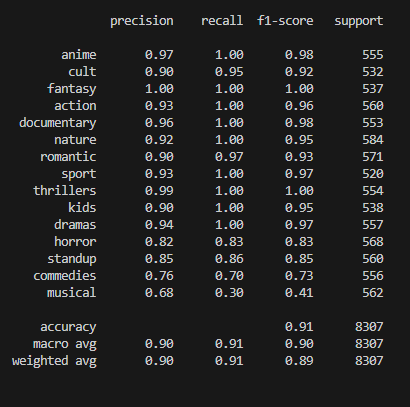


Figura 2: RF

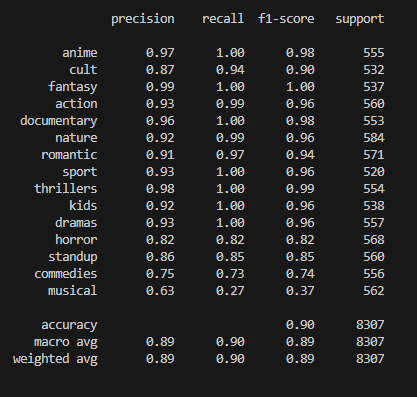
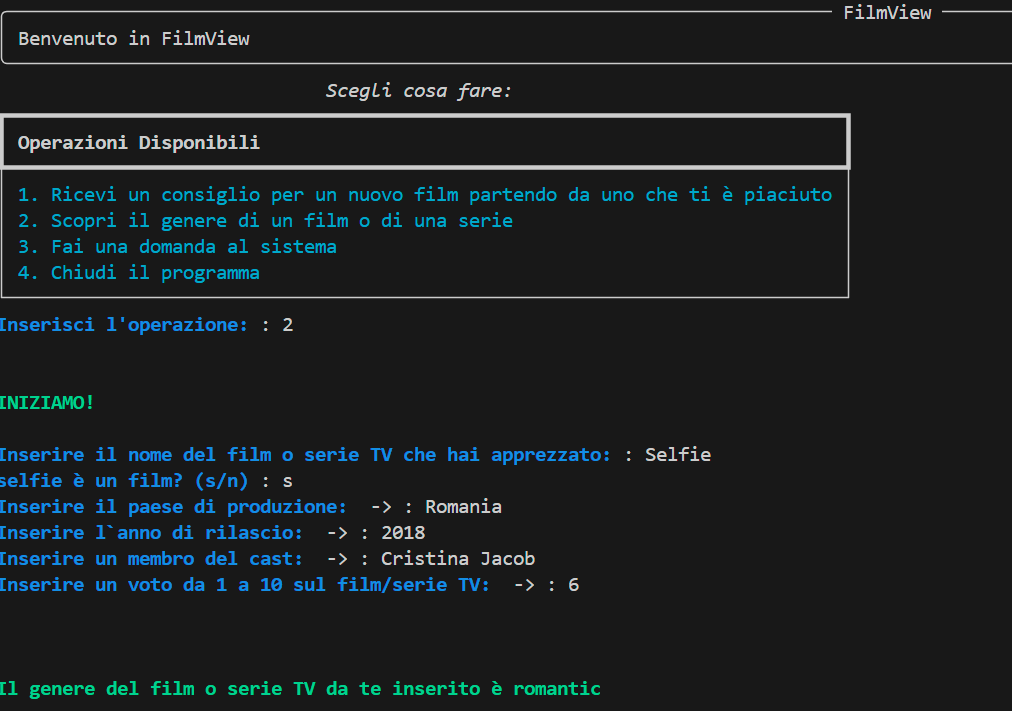


Figura 3: Bagging

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:



**4. CLUSTERING**

Il **clustering** è una tecnica di apprendimento non supervisionato che permette di identificare e raggruppare elementi simili all’interno di grandi dataset. I gruppi creati, detti **cluster,** sono rappresentati da elementi centrali chiamati **centroidi,** che sintetizzano le caratteristiche dei membri del gruppo.

Nel contesto del progetto, il clustering è stato utilizzato per scoprire **nuove similarità e correlazioni** tra i dati, andando oltre il semplice criterio del genere dei film. Queste informazioni sono poi sfruttate come base per lo sviluppo di un **sistema di raccomandazione.**

**a. Scelta dell’algoritmo**

Poiché il dataset contiene un gran numero di **dati categorici**, è stato scelto l’algoritmo **K-Modes**.  
Questo metodo è un’estensione di K-Means pensata per dati categorici:

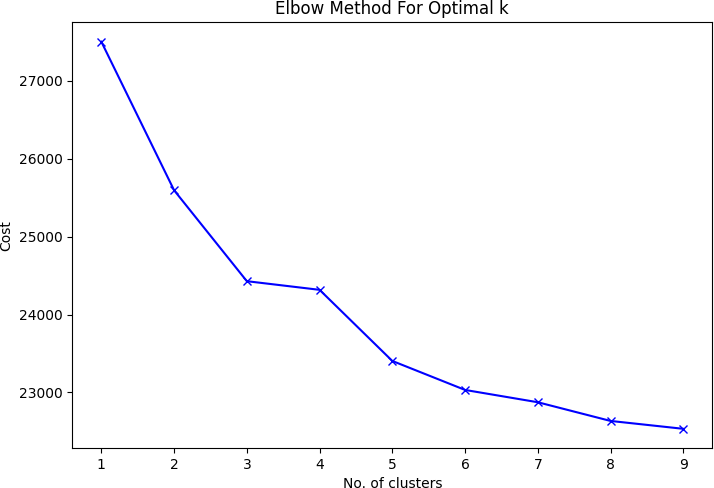
Sostituisce la media con la **moda** per calcolare i centroidi.

Utilizza una misura di similarità specifica per elementi categorici.

Adotta un approccio **frequency-based** per minimizzare la funzione di costo.

**b. Determinazione dei cluster**

Per individuare il numero ottimale di cluster, è stato utilizzato il **metodo del gomito**, una tecnica empirica che aiuta a scegliere il numero più adatto di gruppi all’interno di un certo intervallo.  
In questo caso, il numero di cluster selezionato è **3**, inferiore al numero totale di generi dei film, in modo da evidenziare correlazioni significative **non strettamente legate al genere.**



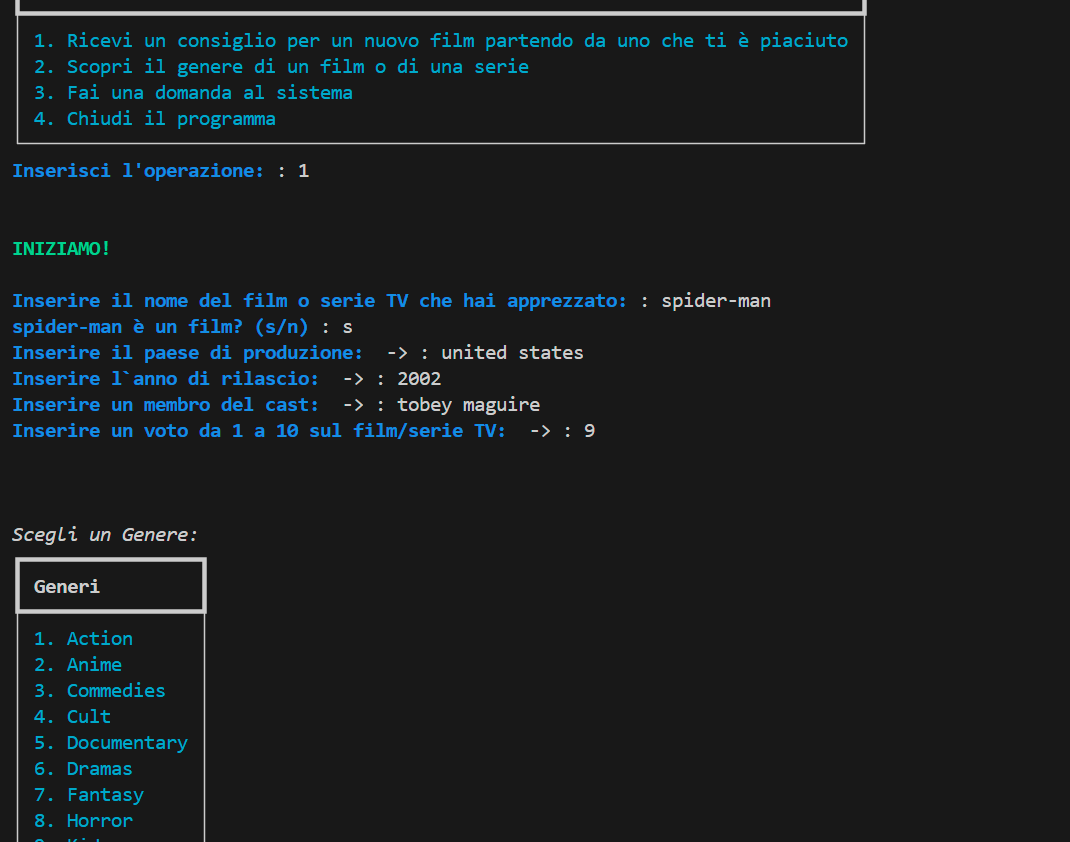
**RECOMMENDER SYSTEM**

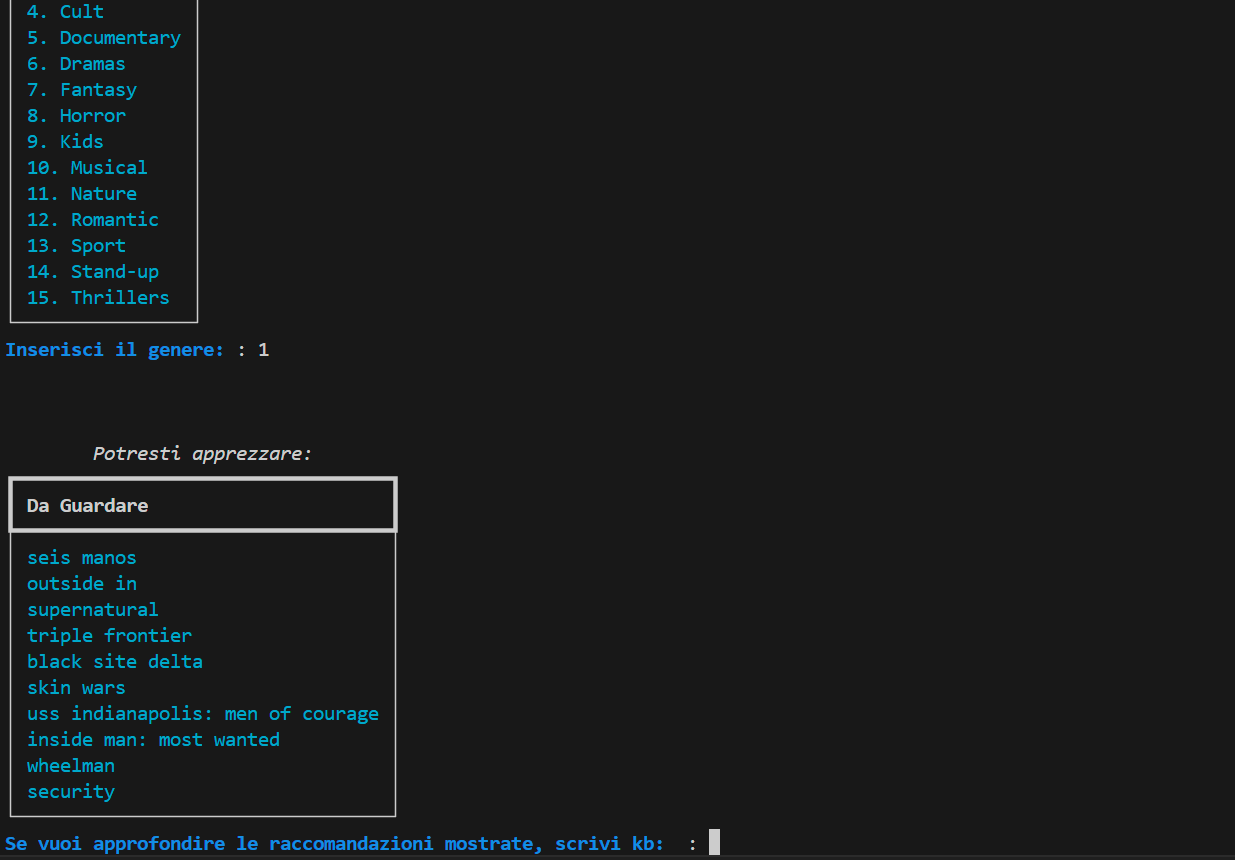
Per il sistema di raccomandazione è stato adottato un **approccio basato sui contenuti**, confrontando le caratteristiche dei film e delle serie TV presenti nel dataset con quelle indicate dall’utente.

In pratica, all’utente viene richiesto di fornire informazioni relative a un film che ha apprezzato. Questi dati vengono utilizzati per identificare il **cluster più simile,** e, di conseguenza, generare una **lista di suggerimenti personalizzata** (top 10), basata sulla somiglianza tra il film indicato e quelli presenti nel cluster selezionato.

Per calcolare la **similarità tra titoli,** è stata impiegata la libreria **FuzzyWuzzy,** che utilizza la **distanza di Levenshtein.** Questa metrica misura quanto due sequenze di caratteri siano diverse, calcolando il numero minimo di modifiche necessarie (inserimenti, cancellazioni o sostituzioni di singoli caratteri) per trasformare una parola nell’altra.

Di seguito è riportato un esempio di utilizzo del Recommender system:





Infine, riportiamo un esempio di esecuzione completa del programma:

