Università degli Studi di Salerno

Dipartimento di Informatica

Corso di Laurea Magistrale in Informatica

Fondamenti di Data Science & Machine Learning

Recommender Systems & Association Rules

Autore
Simone D'Assisi
mat. 0522502038

Tutor
Prof.ssa Loredana Caruccio
Dott.ssa Eleonora Calò

Anno Accademico 2024/2025

Indice

1 Introduzione

I sistemi di raccomandazione rappresentano oggi uno degli strumenti più diffusi e strategici nell'ambito della personalizzazione dei contenuti, con applicazioni che spaziano dall'e-commerce ai servizi di streaming fino ai social network. Il loro obiettivo principale è supportare l'utente nella scelta tra un insieme molto ampio di opzioni, fornendo suggerimenti rilevanti e personalizzati sulla base delle preferenze espresse o implicite.

Secondo la rassegna proposta da S. Gupta [1], i sistemi di raccomandazione possono essere classificati principalmente in tre categorie:

- **Content-based**: sfruttano le caratteristiche degli item per proporre contenuti simili a quelli già apprezzati dall'utente;
- **Collaborative Filtering**: sfruttano le interazioni di utenti simili (user-based) o di oggetti simili (item-based);
- **Sistemi Ibridi**: combinano più strategie per mitigare i limiti dei singoli approcci. Spesso garantiscono migliori performance complessive.

Tuttavia, nonostante i significativi progressi ottenuti in termini di accuratezza predittiva, i sistemi di raccomandazione si trovano ancora ad affrontare numerose sfide aperte [2], come ad esempio:

- **Diversità**: evitare che le raccomandazioni siano troppo simili tra loro o provenienti da un unico cluster di item;
- **Over-specialization**: ridurre la tendenza a proporre esclusivamente contenuti molto vicini al profilo noto dell'utente, limitando così l'esplorazione di nuovi item;
- **Scalabilità**: garantire che le tecniche adottate restino efficienti anche al crescere del numero di utenti e di item.

Per affrontare queste problematiche, negli ultimi anni, oltre al miglioramento delle regole estratte, si è sviluppata un'attenzione crescente verso approcci ibridi che combinano regole di associazione con algoritmi di machine learning. Questi approcci mirano a sfruttare sia la capacità predittiva dei modelli statistici sia la conoscenza implicita nelle regole, offrendo raccomandazioni più accurate e diversificate. Alla luce di queste considerazioni, il lavoro si concentra sulle seguenti Research Questions:

> **RQ1**: Quanto è efficace il post-processing delle regole di associazione nel migliorare la qualità degli item suggeriti?
> **RQ2**: Un approccio ibrido, che combina regole di associazione e algoritmi di machine learning, è in grado di migliorare la qualità delle raccomandazioni rispetto al solo post-processing delle regole?

2 Strumenti di Sviluppo

2.1 Dataset

Per la sperimentazione si utilizza il dataset **Netflix Movie Rating** disponibile su Kaggle[1]. Il dataset è composto da due file CSV principali:

- `Netflix_Dataset_Movie.csv`: contiene tre colonne, ovvero *Movie_ID*, un identificatore numerico univoco per ciascun film, *Year*, l'anno di pubblicazione, e *Name*, il titolo del film. Il dataset include un totale di 17.770 film unici.
- `Netflix_Dataset_Rating.csv`: contiene tre colonne, ovvero *User_ID*, che identifica univocamente ciascun utente, *Movie_ID*, l'identificativo del film valutato, e *Rating*, che indica la valutazione assegnata dall'utente. Complessivamente, il dataset raccoglie circa 17,3 milioni di valutazioni.

Questi dati saranno utilizzati per generare le regole di associazione e per testare le tecniche di post-processing proposte nel presente lavoro.

2.2 Hardware e Ambiente di Esecuzione

Gli esperimenti sono stati eseguiti su un computer con le seguenti caratteristiche hardware e software:

- Processore: Intel(R) Core(TM) i5-12600KF, 12 core;
- Memoria RAM: 16 GB;
- Sistema Operativo: Windows 11.

L'ambiente software comprende Python 3.10 con le principali librerie per l'analisi dei dati, la generazione delle regole di associazione e il calcolo delle metriche di valutazione, come pandas, numpy e sklearn.

2.3 Approcci Utilizzati

Per la sperimentazione sono stati adottati due approcci principali per il post-processing delle regole di associazione, mirati a migliorare la qualità, la diversità e la rilevanza delle raccomandazioni.
Il primo approccio, definito *ibrido*, combina le regole di associazione con modelli di machine learning supervisionati, come Gradient Boosting e XGBoost, trasformando il problema delle raccomandazioni in un compito di classificazione utente–item. In questo modo, le raccomandazioni generate dalle regole vengono filtrate e riordinante in base alla probabilità stimata che un utente apprezzi un determinato film.
Il secondo approccio si concentra su tecniche di post-processing basate su regole e algoritmi di diversità, come il post-mining delle regole e *FairMatch*. Queste strategie agiscono direttamente sulle liste di raccomandazioni già generate, rimuovendo regole ridondanti, generalizzando pattern e incrementando la diversità aggregata degli item suggeriti, senza alterare il modello sottostante.

[1] `https://www.kaggle.com/datasets/rishitjavia/netflix-movie-rating-dataset?select=Netflix_Dataset_Movie.csv`

2.3.1 Estrazione delle Regole di Associazione

Come suggerito in letteratura [3], invece di estrarre le regole direttamente dagli utenti, si è deciso di raggrupparli in cluster in base alle loro caratteristiche, al fine di ridurre il rumore e velocizzare il processo di estrazione delle regole. L'estrazione delle regole è basata sul *association rule mining* (ARM) [4], ed in particolare si è scelto di utilizzare l'algoritmo *FP-Growth* [5], noto per essere meno oneroso rispetto all'algoritmo *Apriori*, in quanto evita la generazione esplicita di candidati. Il processo di FP-Growth inizia rimuovendo gli item che non raggiungono la soglia minima di supporto, riducendo così il rumore e lo spazio di calcolo. Le transazioni rimanenti vengono quindi rappresentate in una struttura ad albero compatta chiamata FP-Tree, nella quale i prefissi comuni condividono i nodi, ottimizzando la memoria. L'albero viene poi esplorato in modo ricorsivo, combinando gli item lungo i percorsi dell'albero per identificare i pattern frequenti. Infine, dai pattern frequenti possono essere generate regole di associazione, filtrando quelle che non soddisfano la soglia minima di confidenza. Grazie alla sua capacità di rappresentare le transazioni in modo compatto e di evitare la generazione di candidati, FP-Growth risulta particolarmente efficiente e scalabile anche su dataset di grandi dimensioni, come in questo caso.

2.3.2 Post-Processing con Modelli di Machine Learning

Questa sezione illustra l'approccio di post-processing basato su modelli di machine learning, finalizzato a migliorare i risultati ottenuti dalle regole di associazione. L'obiettivo è trasformare il problema delle raccomandazioni in un classico problema di classificazione supervisionata, dove ogni coppia utente–film rappresenta un esempio da prevedere.

Strategia Ibrida 1: Regole di Associazione e Gradient Boosting. In questa strategia, le raccomandazioni non derivano esclusivamente dai film più popolari del cluster, ma vengono filtrate e arricchite attraverso un modello di Gradient Boosting. Il **Gradient Boosting** è una tecnica di ensemble learning che costruisce un modello predittivo combinando molti modelli più deboli, tipicamente alberi decisionali poco profondi. L'idea chiave è addestrare i modelli in sequenza, dove ciascun modello successivo corregge gli errori del modello precedente. Il modello viene addestrato a stimare la probabilità che un utente apprezzi un determinato film.

Strategia Ibrida 2: Regole di Associazione e XGBoost. Un'alternativa consiste nell'integrare le regole di associazione con un modello supervisionato basato su **XGBoost**, come suggerito in letteratura [6]. XGBoost (Extreme Gradient Boosting) è un'implementazione ottimizzata del Gradient Boosting, progettata per essere efficiente e scalabile. Il funzionamento di XGBoost può essere sintetizzato come segue:

1. Viene costruito un albero iniziale per predire i target.
2. Si calcolano i residui dell'errore o il gradiente della funzione di perdita rispetto alla predizione corrente.
3. Si addestra un nuovo albero per predire i residui o il gradiente.
4. La predizione finale viene aggiornata combinando tutti gli alberi addestrati, pesati da un *learning rate*.

5. Il processo viene ripetuto per un numero predefinito di alberi o fino al raggiungimento di un criterio di stop.

Per ottimizzare le prestazioni, gli iperparametri di XGBoost vengono selezionati tramite **RandomizedSearchCV** con validazione incrociata a 3 fold. RandomizedSearchCV è una tecnica di ricerca degli iperparametri che esplora un numero predefinito di combinazioni casuali degli iperparametri specificati. Per ciascuna combinazione, le performance del modello vengono valutate tramite cross-validation, una tecnica che stima la capacità di generalizzazione su dati non visti.

2.3.3 Post-Processing Basato su Regole

Rule Base. L'articolo "*Post-Mining on Association Rule Bases*" [7] evidenzia come le regole di associazione grezze non siano direttamente utilizzabili nei sistemi reali, poiché spesso contengono rumore, ridondanze o risultano troppo generiche. L'estrazione tradizionale di regole di associazione genera un ampio numero di regole, molte delle quali ridondanti o di scarso valore, fenomeno noto come *pattern explosion*, che rende difficile per gli esperti del dominio trarre conclusioni significative. L'articolo esplora tecniche di *post-mining*, ossia metodi applicati dopo l'estrazione delle regole per affinare e ottimizzare i risultati. Queste tecniche includono la rimozione di regole ridondanti, la generalizzazione delle regole per migliorarne la generalizzabilità e la valutazione delle regole in base alla loro rilevanza e utilità pratica.

FairMatch. L'articolo "*FairMatch: A Graph-based Approach for Improving Aggregate Diversity in Recommender Systems*" [8] propone una strategia per affrontare il problema della bassa diversità aggregata nei sistemi di raccomandazione tradizionali, che tendono a privilegiare gli item più popolari. L'algoritmo **FairMatch**, basato su grafi, agisce come fase di post-elaborazione dopo la generazione delle raccomandazioni, al fine di aumentare la diversità aggregata e garantire una distribuzione più equa degli articoli consigliati. In particolare, FairMatch costruisce un grafo bipartito tra utenti e articoli, dove gli archi rappresentano le raccomandazioni generate dal sistema. L'algoritmo risolve iterativamente il problema del flusso massimo in questo grafo, identificando e aggiungendo articoli di alta qualità ma poco raccomandati alle liste finali degli utenti. Questo processo consente di incrementare la diversità aggregata delle raccomandazioni senza compromettere significativamente la loro rilevanza.

2.4 Valutazione delle Raccomandazioni

Per valutare l'efficacia delle raccomandazioni generate dai modelli ibridi, sono state adottate tre metriche fondamentali: *Precision@10*, *Recall@10* e *MAP@10*. Queste metriche, ampiamente utilizzate nei sistemi di raccomandazione, permettono di misurare rispettivamente la qualità, la completezza e l'accuratezza del ranking delle raccomandazioni proposte.
La Precisione@10 quantifica la frazione di elementi rilevanti tra i primi 10 suggerimenti forniti dal sistema. In altre parole, indica la percentuale di film che l'utente ha effettivamente apprezzato tra i primi 10 raccomandati. Una precisione elevata suggerisce che il sistema è

efficace nel proporre contenuti pertinenti nelle posizioni di maggiore visibilità.

$$Precision@10 = \frac{\text{Numero di film rilevanti trai primi 10}}{10}$$

Il Recall@10, invece, misura la proporzione di film rilevanti che sono stati inclusi tra i primi 10 suggerimenti, rispetto al totale di film rilevanti disponibili. Questa metrica evidenzia la capacità del sistema di recuperare gli item pertinenti all'interno dei primi 10 suggerimenti.

$$Precision@10 = \frac{\text{Numero di film rilevanti trai primi 10}}{\text{Numero totale di film rilevanti}}$$

Infine, la *Mean Average Precision (MAP@10)* considera non solo la presenza dei film rilevanti, ma anche la loro posizione all'interno della lista dei suggerimenti. Per ogni utente si calcola la *Average Precision* considerando l'ordine dei film raccomandati, e successivamente si calcola la media su tutti gli utenti. La MAP@10 fornisce quindi una misura complessiva della qualità del ranking proposto dal sistema di raccomandazione, premiando le raccomandazioni pertinenti che compaiono nelle prime posizioni.

$$MAP@10 = \frac{1}{Q} \sum_{q=1}^{Q} AP@10_q$$

dove $AP@10_q$ è l'Average Precision, espressa nel seguente modo:

$$AP@10 = \frac{\sum_{k=1}^{10} Precision@k \cdot rel_k}{\min(R, 10)}$$

dove R è il numero totale di film rilevanti per l'utente e rel_k è una funzione indicatrice che vale 1 se il film in posizione i è rilevante, 0 altrimenti.
L'uso combinato di Precision@10, Recall@10 e MAP@10 permette di valutare in modo completo le performance del sistema di raccomandazione, considerando non solo la pertinenza e la copertura dei suggerimenti, ma anche la qualità del loro ordine nella lista raccomandata.

3 Sperimentazione

3.1 Pre-Processing dei Dati

Da una prima analisi del dataset è emerso che il numero complessivo di film è pari a 17.770, ma soltanto 1.350 di essi hanno ricevuto almeno una valutazione. Il numero totale di utenti presenti è invece pari a 143.458. Ne consegue che la matrice booleana utente–film, necessaria per l'estrazione delle regole di associazione, avrebbe dimensioni pari a $143.458 \times 17.770 \approx 2.55 \times 10^9$ celle. Data la limitata disponibilità di risorse computazionali, tale operazione conduceva inevitabilmente a un `MemoryError`. È stato quindi necessario ridurre la dimensionalità del problema.

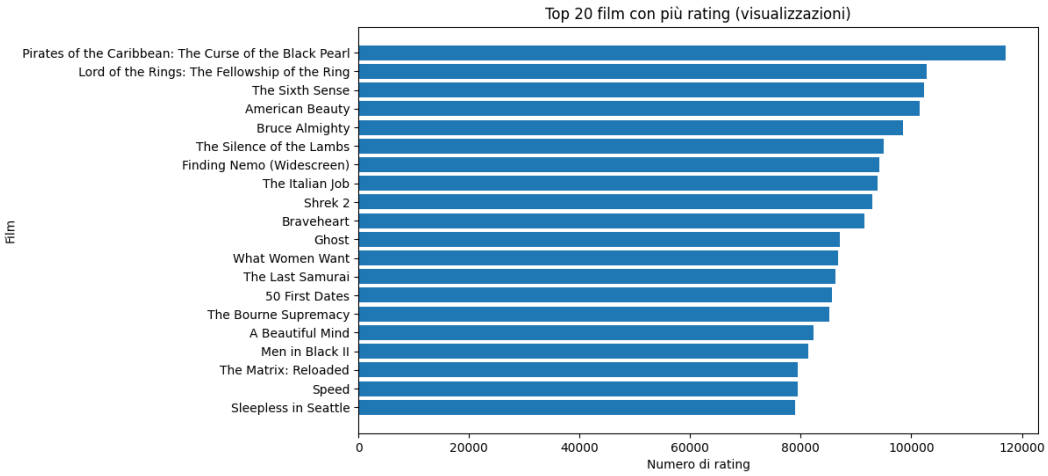

Figura 1: Nell'immagine sono stati riportati i 20 film con più visualizzazioni, cioè i film che hanno ricevuto più rating dagli utenti.

La prima strategia adottata è consistita nel filtrare gli utenti inattivi, definiti come coloro che avevano registrato meno di 50 valutazioni. Questa operazione ha ridotto il numero di utenti da 143.458 a 139.962, un valore tuttavia ancora troppo elevato per consentire l'estrazione delle regole senza errori di memoria.

3.2 Clustering degli Utenti

Per ridurre ulteriormente la complessità computazionale e facilitare il post-processing delle raccomandazioni, gli utenti sono stati raggruppati in cluster [3], applicando l'algoritmo *K-Means* alla matrice utente–film. Considerando l'elevato numero di utenti e film presenti nel dataset, si è scelto di utilizzare una variante più scalabile, `MiniBatchKMeans` della libreria `sklearn.cluster`, che elabora i dati in batch di dimensione 1000, riducendo significativamente i tempi di calcolo. L'algoritmo è stato addestrato esclusivamente sugli utenti del training set, con uno split 80-20, generando le etichette di cluster per ciascun utente di addestramento. Successivamente, per gli utenti del validation set è stata effettuata solo la fase di predizione, assegnando ciascun utente al cluster più vicino sulla base dei centroidi calcolati. In totale sono stati generati 1000 cluster, garantendo una suddivisione sufficientemente fine degli utenti pur mantenendo la scalabilità dell'operazione. Per rappresentare i cluster di tutti gli utenti in un unico array, è stato creato un vettore della stessa lunghezza del numero

totale di utenti, in cui sono state inserite le etichette di cluster corrispondenti sia al training sia al validation set. In questo modo, per ogni utente è disponibile un riferimento diretto al cluster di appartenenza, utile sia per il calcolo delle feature dei modelli di post-processing sia per la selezione dei film candidati durante la fase di raccomandazione.

3.3 Estrazione delle Regole di Associazione

Prima di procedere all'estrazione delle regole di associazione è stato necessario costruire le transazioni. In questo lavoro, ciascun cluster è stato considerato come una singola transazione, mentre ogni film rappresenta un item. Al fine di catturare soltanto le preferenze positive, sono stati inclusi unicamente i film con un valore di rating ≥ 3, interpretato come indicatore di apprezzamento da parte degli utenti. Di conseguenza, una transazione è caratterizzata da un identificativo del cluster e dall'insieme dei film apprezzati dagli utenti che vi appartengono.
Per l'estrazione delle regole è stato inizialmente valutato l'algoritmo *Apriori*, che tuttavia si è dimostrato inefficiente sia in termini di tempo di esecuzione sia in termini di utilizzo della memoria, conducendo spesso a `MemoryError`. Si è quindi optato per l'algoritmo *FP-Growth*, che si è rivelato più scalabile ed efficiente. L'algoritmo è stato configurato con una soglia minima di supporto pari al 30% (valori inferiori incrementavano drasticamente i tempi di calcolo) e una soglia di confidenza pari al 60%.
L'applicazione di FP-Growth ha portato all'individuazione di 3021 pattern frequenti, dai quali sono state derivate complessivamente **1027 regole di associazione**.

3.4 Generazione delle Raccomandazioni

3.4.1 Raccomandazioni Basate sui Cluster di Appartenenza

Come baseline per la valutazione delle raccomandazioni, si è scelto di utilizzare un approccio basato sulla popolarità dei film all'interno del cluster di appartenenza di ciascun utente. In questo contesto, per ogni utente vengono raccomandati unicamente i film più popolari nel cluster a cui appartiene, escludendo quelli già presenti nella cronologia dell'utente nel training set. Per implementare questa strategia, è stato creato un dizionario `user_history_dict` contenente, per ciascun utente del training set, la lista dei film che ha apprezzato. Allo stesso modo, per il validation set è stato costruito un dizionario `val_df_liked_dict` che funge da ground truth, indicando quali film sono stati effettivamente apprezzati dagli utenti di validazione.
Successivamente, per ogni cluster è stata calcolata la popolarità dei film basata sulla somma dei like degli utenti appartenenti al cluster e i film sono stati ordinati in ordine decrescente di popolarità. In questo modo, per ciascun cluster è disponibile una lista ordinata dei film candidati, che viene utilizzata per generare le raccomandazioni. Le raccomandazioni per ogni utente sono ottenute tramite la funzione `recommend_from_clusters`, che recupera il cluster dell'utente, seleziona i film più popolari nel cluster escludendo quelli già visti, e restituisce i primi `top_k` film come suggerimenti. Nel caso in cui l'utente non sia presente nel training set, la funzione restituisce i film più popolari a livello globale.
Per valutare la qualità delle raccomandazioni, è stata definita la funzione `evaluate_user`, che calcola la precisione e il recall per ciascun utente. Una hit viene conteggiata ogni volta

che un film raccomandato compare nella cronologia di apprezzamento presente nel validation set.
Infine, le metriche sono state calcolate su tutti gli utenti del validation set e sono state aggregate come media, ottenendo una valutazione complessiva della baseline basata sui film più popolari all'interno dei cluster. Questo approccio consente di ottenere un punto di riferimento chiaro per confrontare le performance dei modelli di post-processing successivamente introdotti, sia quelli ibridi basati su machine learning sia le strategie basate su regole e FairMatch.

Metrica	**Valore**
Precision@10	0.7833
Recall@10	0.0916
MAP@10	0.7013

Tabella 1: Risultati della Strategia Cluster Based.

3.4.2 Raccomandazioni con Association Rules: Applicazione Diretta con Priorità

Prima di valutare i risultati ottenuti a seguito dell'applicazione di strategie di post-processing, si procede con l'applicazione diretta delle regole di associazione estratte sui suggerimenti cluster-based: in questa configurazione, le raccomandazioni di base ottenute dal clustering vengono arricchite applicando tutte le regole di associazione attivabili per ciascun utente, ordinate in base al loro livello di confidence. Per ogni utente, la funzione `apply_rules_to_recs_priority` verifica quali regole sono applicabili rispetto ai film già visti e alle raccomandazioni iniziali, aggiungendo i film suggeriti dalle regole che non sono ancora stati visualizzati. In questo modo, i film derivanti dalle regole di associazione vengono privilegiati e compaiono all'inizio della lista finale di raccomandazioni. Le raccomandazioni cluster-based vengono inserite successivamente, esclusivamente per completare la lista fino al numero di elementi desiderato (`top_k`). Questo approccio garantisce che le regole di associazione abbiano priorità nel ranking finale, senza però rimuovere completamente i film popolari nel cluster.
Per valutare le performance di questa strategia, è stata definita la funzione `evaluate_user_with_rules_priority`, che calcola la Precision@10 e la Recall@10 confrontando la lista finale di raccomandazioni con i film effettivamente apprezzati dagli utenti nel validation set. In questo modo, è possibile stimare quanto l'integrazione delle regole migliori la qualità e la completezza delle raccomandazioni rispetto alla baseline cluster-based.
Questa metodologia non prevede ulteriori filtraggi o ottimizzazioni delle regole oltre alla loro ordinazione per livello di confidence: si tratta quindi di un semplice post-processing basato sulla priorità delle regole, volto a verificare l'impatto diretto dell'integrazione delle regole sulle metriche di valutazione.

Metrica	Valore
Precision@10	0.7429
Recall@10	0.0857
MAP@10	0.6336

Tabella 2: Risultati della Strategia con Regole di Associazione con Priorità.

3.4.3 Post-Processing con Modelli di Machine Learning

Come anticipato nel secondo capitolo, sono state esplorate due strategie ibride di ottimizzazione in post-processing, con l'obiettivo di valutare se e in che misura l'integrazione delle regole di associazione possa migliorare la qualità delle raccomandazioni inizialmente ottenute dal clustering. Per ogni modello, saranno necessari i seguenti elementi:

- Suddivisione del dataset in training e validation set, già eseguita in fase di clustering, per garantire che il modello venga addestrato unicamente sui dati di training e valutato su utenti e interazioni non viste in precedenza.
- Regole di associazione, memorizzate in un dizionario (`rules_dict`) in cui a ogni antecedente corrisponde la lista dei conseguenti e la relativa confidence.
- Cronologia degli utenti (solo sul training set), necessaria per attivare eventuali regole di associazione e generare raccomandazioni personalizzate.
- Popolarità dei film, calcolata sia a livello globale che locale (all'interno dei cluster). Nel secondo caso, per ogni cluster viene prodotto un ordinamento dei film più visti, utile a costruire i candidati di base.
-

A partire da queste componenti, il codice definisce una funzione di costruzione delle feature (build_features) che, per ogni coppia utente–film candidata, genera un vettore con i seguenti elementi:

- **rank di popolarità nel cluster**, che cattura l'importanza del film tra gli utenti simili;
- **popolarità globale**, che misura l'interesse generale del film;
- **contributo delle regole di associazione**, sia in termini di confidence massima che di numero di regole attivate.

Queste feature costituiscono l'input per un classificatore supervisionato che stima la probabilità che l'utente apprezzi il film candidato. I risultati ottenuti per ciascun utente vengono raccolti e salvati in un unico file `train_dataset.csv`, così da consentire l'addestramento e la valutazione dei modelli supervisionati.
Per ogni modello selezionato è prevista una funzione di valutazione (`evaluate_hybrid_ml`), che seleziona per ciascun utente un insieme di candidati, ottenuto combinando i film più popolari del cluster con quelli suggeriti dalle regole di associazione. I film candidati sono quindi ordinati in base allo score predetto e vengono calcolate le metriche di Precision@k e Recall@k sui film effettivamente apprezzati dall'utente nel validation set. In tal modo è

possibile verificare se, e in che misura, l'integrazione dei segnali derivanti dalle regole di associazione con quelli basati sulla popolarità migliori la qualità delle raccomandazioni rispetto al semplice approccio cluster-based.

Pipeline di esecuzione. La pipeline di esecuzione per le strategie Ibride con Machine Learning si articola in due fasi principali, focalizzate sulla conversione del problema di raccomandazione in un problema di ranking supervisionato. Durante la Fase 1 ci si occupa della generazione del Dataset di Training per il Modello ML. Questa fase è cruciale per la preparazione del classificatore supervisionato (XGBoost/Gradient Boosting) e viene eseguita unicamente sugli utenti del set di training (*U_train*). L'obiettivo è costruire un dataset (*D_train*) in cui ogni riga rappresenti una potenziale interazione utente-film con le relative feature e l'etichetta di gradimento (Target). La Fase 1 si articola nei seguenti passi principali:

- Identificazione dei Candidati (*M_cand*): per ciascun utente (u), si identificano i film potenzialmente raccomandabili combinando due insiemi: Candidati Cluster-Based, e cioè i film più popolari nel cluster di appartenenza C(u) (segnalati dalla popolarità locale) ed i Candidati Rule-Based, cioè consequent (M) delle regole di associazione (R) che vengono attivate dallo storico di gradimento dell'utente (*A SUBSET DI H_train(u)*).

- Estrazione delle Feature: per ogni coppia utente-film candidata (u, m), si calcolano quattro segnali distinti: *F_cluster* e *F_global* catturano i segnali di popolarità (locale e globale) ed *F_conf_max* e *F_triggered* integrano il contributo delle regole di associazione in termini di forza e robustezza del suggerimento.

- Etichettatura (Target): viene assegnata l'etichetta di gradimento (y), che corrisponde ad 1 se il film m è presente nello storico dell'utente *H_train(u)* (gradito), e 0 altrimenti (non gradito o non ancora visto).

- Addestramento: il dataset cosi generato (*D_train_final*) e utilizzato per addestrare il modello *R_ML*, che impara a stimare la probabilità di gradimento in base alla combinazione delle quattro feature.

Durante la Fase 2 ci si concentra sulla generazione delle raccomandazioni: il modello *R_ML* addestrato per generare le raccomandazioni finali sugli utenti del set di validazione (*U_val*). La Fase 2 si articola nei seguenti passi:

- Generazione dei Candidati per Predizione: per un utente di validazione, si ripete il processo della Fase 1 per ottenere il set di candidati (*D_val*) e le relative feature. In questo passaggio, vengono esclusi i film già visti.

- Predizione e Punteggio: il modello *R_ML* calcola il punteggio predetto (*S_hat*) per ogni candidato non visto. Questo punteggio rappresenta la probabilità stimata che l'utente possa apprezzare il film.

- Ranking Finale: I film vengono ordinati in una lista (*L_ranked*) in base al punteggio *S_hat* in ordine decrescente.

- Output: i Top K film di questa lista ordinata vengono emessi come raccomandazione finale, combinando i segnali derivanti dal clustering (popolarità) con l'informazione predittiva e relazionale (Rules) appresa dal modello ML.

```
Input
- U_train, U_val: Set di utenti di Training e Validation
- C(u): Cluster assegnato a utente u
- H_train(u): Storico dei film piaciuti (Rating >= 3) dall'utente u (nel train set)
- R: Set di Association Rules (filtrate/post-minate)

// FASE 1: Generazione del Dataset di Training per il Modello ML
FUNCTION GeneraDatasetML(U_set, H)
    D_train <- vuoto
    FOR EACH u IN U_set:
        // Candidati Cluster-Based (top 100 piu popolari nel cluster)
        M_cand <- Top N film in C(u)

        // Candidati Rule-Based (Consequent delle regole attivate)
        FOR EACH regola A => M IN R:
            SE A SUBSET DI H(u):
                M_cand <- M_cand UNIONE {M}

        FOR EACH candidato m IN M_cand:
            // Feature 1: Popolarita di m nel cluster C(u)
            F_cluster <- Popolarita(m | C(u))
            // Feature 2: Popolarita globale di m
            F_global <- PopolaritaGlobale(m)
            // Feature 3: Max Confidence delle regole che suggeriscono m
            F_conf_max <- MAX(Confidence(A => m)) per tutte le regole attivate
            // Feature 4: Conteggio delle regole attivate che suggeriscono m
            F_triggered <- Conteggio Regole(m)
            // Target
            y <- SE m IN H(u) ALLORA 1 ALTRIMENTI 0

            // Aggiungi al dataset
            D_train <- D_train UNION{(u, m, F_cluster, F_global, F_conf_max, F_triggered, y)}

    RETURN D_train

// Addestramento Modello
D_train_final <- GeneraDatasetML(U_train, H_train)
R_ML <- Addestra(XGBoost / Gradient Boosting, D_train_final)

// FASE 2: Raccomandazione e Ranking (Validation/Test)
FOR EACH u IN U_val:
    // Genera dataset di candidati per la predizione (stesso processo della FASE 1)
    D_val <- GeneraDatasetML({u}, H_train)
    L_ranked <- vuoto

    FOR EACH record (u, m, F_cluster, F_global, F_conf_max, F_triggered) IN D_val:
        // Escludi film gia visti
        SE m NON IN H_train(u):
            // Predizione: probabilita di gradimento
            S_hat <- R_ML(F_cluster, F_global, F_conf_max, F_triggered)
            L_ranked <- L_ranked UNIONE { (m, S_hat) }

```

```
    L_final <- Ordina(L_ranked per S_hat decrescente)

    Output: Top K film di L_final
```

Codice 1: Pipeline Post Processing con Modelli di ML.

Strategia Ibrida 1: Regole di Associazione e Gradient Boosting. Questa strategia integra i segnali derivanti dalle regole di associazione con quelli basati sulla popolarità dei film, all'interno di un modello supervisionato di tipo Gradient Boosting. Partendo dal `train_dataset.csv`, il modello impara a stimare la probabilità che un film venga gradito da un utente. La funzione `evaluate_hybrid_ml` viene quindi utilizzata per generare un insieme di dieci raccomandazioni per ciascun utente del validation set. I risultati ottenuti sono riportati nella Tabella 3.

Metrica	**Valore**
Precision@10	0.7525
Recall@10	0.0875
MAP@10	0.6560

Tabella 3: Risultati della Strategia Ibrida 1.

Strategia Ibrida 2: Regole di Associazione e XGBoost. Per questa strategia, si parte dal caricamento del dataset di addestramento `train_dataset.csv`. In una prima fase, si cerca di identificare i migliori iperparametri per l'addestramento del modello XGBoost, comprendenti numero di alberi, profondità massima, learning rate, frazione di campioni e feature da usare, peso minimo dei figli e parametro di regolarizzazione `gamma`. Questi valori possibili sono definiti nel dizionario `param_dist`.
Successivamente, viene lanciata una `RandomizedSearchCV`, che esplora 15 combinazioni casuali con validazione incrociata a 3 fold. La metrica ottimizzata è l'AUC ROC, che rappresenta la probabilità che il modello assegni un punteggio più alto a un positivo rispetto a un negativo scelto a caso. Al termine della ricerca, il miglior modello viene salvato in un file `.pkl`.
Prima dell'addestramento definitivo, si calcola il peso di bilanciamento delle classi (`scale_pos_weight`), utile nei casi in cui i film apprezzati siano molto meno numerosi rispetto a quelli non apprezzati.
Il modello ottimizzato viene quindi ricaricato e riaddestrato sui dati di training, monitorando le performance sul validation split. Terminata questa fase, si esegue la valutazione sul validation set completo tramite la funzione `evaluate_hybrid_ml`. I risultati sono riportati nella Tabella 4.

Metrica	**Valore**
Precision@10	0.7397
Recall@10	0.0861
MAP@10	0.6397

Tabella 4: Risultati della Strategia Ibrida 2.

3.4.4 Post-Processing Basato su Regole

A titolo di confronto, sono state analizzate strategie che non impiegano modelli di Machine Learning, ma mirano a incrementare la qualità delle regole di associazione estratte e dei suggerimenti generati. Tali approcci si concentrano sul filtraggio, la selezione e la combinazione intelligente delle regole, al fine di produrre raccomandazioni più rilevanti e specifiche per ciascun utente.

Adattamento della Strategia Rule Base. L'articolo di Seipel [7] evidenzia come le regole di associazione grezze possano risultare ridondanti, generiche o poco informative a causa del fenomeno della *pattern explosion*. Per questo motivo, è stata introdotta una fase di *post-mining*, ossia un insieme di trasformazioni e filtraggi successivi all'estrazione delle regole, volta a ottenere un insieme più compatto e significativo. In questo adattamento della strategia *Rule Base*, il processo si articola nei seguenti passaggi:

- **Precalcolo dei cluster dei film**: per ogni film viene costruita una mappa dei cluster degli utenti che lo hanno visto, così da identificare le comunità più rappresentative rispetto ai contenuti.
- **Post-mining delle regole**: vengono rimosse le regole con antecedenti troppo brevi (lunghezza minima `min_len`) e ciascuna regola viene assegnata al cluster più frequente tra gli utenti che hanno visto i film dell'antecedente. In questo modo, ogni regola è specializzata per un cluster specifico.
- **Selezione dei candidati per utente**: per ogni utente del validation set, vengono selezionati i film più popolari nel cluster di appartenenza, escludendo quelli già presenti nella cronologia dell'utente e rispettando un limite massimo di suggerimenti per film (`max_per_item`).
- **Applicazione delle regole post-minate**: vengono considerati solo i film suggeriti dalle regole assegnate al cluster dell'utente, attivando una regola solo se tutti gli item dell'antecedente sono presenti nella cronologia dell'utente. Se più regole suggeriscono lo stesso film, viene mantenuto quello con la confidenza massima.
- **Combinazione dei segnali**: le raccomandazioni derivanti dai candidati del cluster e dalle regole vengono fuse tramite una media pesata, con un peso maggiore ($\alpha = 0.7$) assegnato alle regole. Questo bilancia la conoscenza esplicita fornita dalle regole con la preferenza implicita derivante dalla popolarità locale.

In questo modo, la strategia *Rule Base* aggiornata garantisce che i suggerimenti siano rilevanti, specifici rispetto al cluster dell'utente e non ridondanti, integrando efficacemente informazioni esplicite e implicite. I risultati prodotti dalla funzione `evaluate_rule_base` sono contenuti nella Tabella 5.

Metrica	**Valore**
Precision@10	0.7839
Recall@10	0.0917
MAP@10	0.7026

Tabella 5: Risultati della Strategia Rule Base.

Adattamento della Strategia FairMatch. FairMatch [8] è originariamente un algoritmo basato su grafo che estende le liste di raccomandazioni generate da un modello base, costruendo un grafo bipartito utenti-item al fine di migliorare la copertura e promuovere item con bassa visibilità. Nel nostro lavoro, la strategia è stata adattata per bilanciare la popolarità locale e la diversità delle raccomandazioni, ottimizzando al contempo l'efficienza computazionale. Le principali modifiche introdotte rispetto all'algoritmo originale sono le seguenti:

- **Cluster al posto del grafo globale**: invece di costruire un grafo utenti-item su tutto il dataset, per ciascun utente vengono considerati solo i film più popolari nel proprio cluster. Questo riduce significativamente la complessità computazionale.
- **Quota massima per film**: attraverso il parametro `max_per_item_ratio`, si limita il numero di utenti ai quali lo stesso film può essere raccomandato, prevenendo una concentrazione eccessiva sugli item più popolari e favorendo la diversità.
- **Selezione greedy dei candidati**: se un film raggiunge il limite massimo di raccomandazioni consentite, non viene più selezionato per altri utenti. Questa scelta rappresenta un'approssimazione del matching globale implementato da FairMatch tramite il grafo, ma è molto più scalabile.
- **Batch processing**: l'algoritmo viene applicato su blocchi di utenti (batch di 10.000) per evitare di processare l'intera matrice utenti-item in un'unica soluzione, garantendo scalabilità su dataset di grandi dimensioni.

La versione adattata di FairMatch mantiene quindi la filosofia originale di aumentare copertura e diversità, ma utilizza una strategia greedy a livello di cluster combinata con vincoli di quota massima per film e gestione a batch, consentendo un'applicazione efficiente anche a dataset di grandi dimensioni. i Risultati sono riportati nella Tabella 6.

Metrica	**Valore**
Precision@10	0.7069
Recall@10	0.0800
MAP@10	0.6099

Tabella 6: Risultati della Strategia FairMatch.

3.4.5 Perfezionamento dei Modelli con Rule Base

Tra le strategie di post-mining delle regole di associazione, l'approccio *Rule Base* si è dimostrato il più efficace, ottenendo i migliori valori in *Precision@10*, *Recall@10* e *MAP@10*. A partire da questo risultato, si è deciso di integrare il dizionario di regole prodotto da Rule Base all'interno dei modelli di Machine Learning, con l'obiettivo di migliorare ulteriormente le loro prestazioni.
Le funzioni di costruzione delle feature e di valutazione sono state opportunamente modificate per utilizzare il nuovo dizionario, pur mantenendo invariato il workflow complessivo adottato in precedenza. L'aggiunta della MAP@10 consente di valutare anche la qualità del ranking dei film raccomandati, premiando i suggerimenti rilevanti posizionati nelle prime posizioni della lista.

I risultati, riportati in Tabella 7, evidenziano come l'inclusione delle regole di post-mining generi un incremento, seppur contenuto, in tutti e tre gli indicatori considerati. In particolare, il *Gradient Boosting* ottiene un miglioramento sia in precisione (da 0.7525 a 0.7546) che in recall (da 0.0875 a 0.0878) e in MAP@10 (da 0.6560 a 0.6592), mentre l'*XGBoost* ottimizzato tramite *Random Search CV* mostra un analogo incremento (Precision@10: da 0.7397 a 0.7411; Recall@10: da 0.0861 a 0.0863; MAP@10: da 0.6397 a 0.6392).
Questi risultati confermano che il post-mining con Rule Base, pur non stravolgendo le performance dei modelli, contribuisce in maniera coerente e costante a migliorarne l'accuratezza complessiva e la qualità del ranking dei suggerimenti.

Strategia	**Precision@10**	**Recall@10**	**MAP@10**
Gradient Boosting	0.7525	0.0875	0.6560
Rule Base + Gradient Boosting	0.7546	0.0878	0.6592
Random Search CV + XGBoost	0.7397	0.0861	0.6397
Rule Base + Random Search CV + XGBoost	0.7411	0.0863	0.6392

Tabella 7: Confronto dei risultati ottenuti dai modelli con e senza utilizzo di Rule Base.

3.5 Confronto Risultati Prodotti

Dall'analisi comparativa delle diverse strategie di raccomandazione (Tabella 8) emergono alcuni punti chiave, che permettono di trarre considerazioni utili sulle prestazioni e sui limiti di ciascun approccio.

- **Cluster-Based**: rappresenta una baseline solida, con risultati elevati in termini di *Precision@10* (0.7833), *Recall@10* (0.0916) e *MAP@10* (0.7013). La forza principale risiede nella semplicità: le raccomandazioni si basano sulla popolarità dei film osservata all'interno dei cluster di utenti. Tuttavia, privilegia i film più popolari e già noti, riducendo la diversità dei suggerimenti.

- **Regole di Associazione con Priorità**: l'uso diretto delle regole estratte dal dataset, ordinate per priorità, produce risultati discreti ma inferiori alla baseline cluster-based (Precision@10: 0.7429; Recall@10: 0.0857; MAP@10: 0.6336). Questo conferma che le regole catturano pattern utili, ma la loro capacità di generalizzazione in fase di validazione è limitata.

- **Gradient Boosting e XGBoost (Random Search CV)**: l'introduzione di modelli di Machine Learning consente di catturare relazioni più complesse tra feature di cluster, popolarità e regole. I risultati mostrano performance competitive, leggermente inferiori alla baseline, ma con maggiore flessibilità e potenziale miglioramento su dataset più grandi o feature più ricche.

- **Rule Base**: conferma di essere la strategia più efficace, con i valori più alti di Precision@10 (0.7839), Recall@10 (0.0917) e MAP@10 (0.7026). Il post-mining filtra e raffina le regole di associazione, migliorando la qualità delle raccomandazioni rispetto alle regole grezze e al cluster-based puro. È tuttavia più complesso e costoso computazionalmente.

- **FairMatch**: i risultati sono più bassi rispetto ad altre strategie (0.7069 / 0.0800 / 0.6099), ma il metodo introduce maggiore copertura e diversità, grazie ai vincoli sulla quota massima per item e alla selezione basata sui cluster. È quindi utile quando la varietà delle raccomandazioni è prioritaria rispetto alla massimizzazione delle metriche di accuratezza.

- **Integrazione Rule Base + Machine Learning**: combinare le regole post-minate con i modelli di Machine Learning genera un miglioramento marginale ma costante per ogni modello. Questo evidenzia come le informazioni provenienti dalle regole possano raffinare i modelli predittivi, migliorando precisione, recall e qualità del ranking (MAP), anche se in maniera incrementale.

In conclusione, i risultati ottenuti confermano che nessuna strategia è universalmente ottimale: la scelta dipende dagli obiettivi del sistema di raccomandazione. Il cluster-based rimane una baseline semplice ed efficace, mentre il rule base, grazie al post-mining, si dimostra la soluzione più accurata. I modelli di machine learning, pur non superando le performance della baseline, garantiscono maggiore flessibilità e stabilità, soprattutto quando combinati con le regole. Infine, FairMatch, pur sacrificando la precisione, introduce diversità e copertura, rivelandosi utile in scenari in cui la varietà ha un ruolo centrale.

Strategia	**Precision@10**	**Recall@10**	**MAP@10**
Cluster-Based	0.7833	0.0916	0.7013
Regole di Associazione con Priorità	0.7429	0.0857	0.6336
Gradient Boosting	0.7525	0.0875	0.6560
Random Search CV + XGBoost	0.7397	0.0861	0.6397
Rule Base	0.7839	0.0917	0.7026
FairMatch	0.7069	0.0800	0.6099
Rule Base + Gradient Boosting	0.7546	0.0878	0.6392
Rule Base + Random Search CV + XGBoost	0.7411	0.0863	0.6592

Tabella 8: Confronto dei risultati ottenuti con ogni strategia.

Riferimenti bibliografici

[1] Shraddha Gupta. «A literature review on recommendation systems». In: *Int. Res. J. Eng. Technol* 7 (2020), pp. 3600–3605.

[2] Marwa Hussien Mohamed, Mohamed Helmy Khafagy e Mohamed Hasan Ibrahim. «Recommender systems challenges and solutions survey». In: *2019 international conference on innovative trends in computer engineering (ITCE)*. IEEE. 2019, pp. 149–155.

[3] Fakhri Fauzan, Dade Nurjanah e Rita Rismala. «Apriori association rule for course recommender system». In: *Indonesian Journal on Computing (Indo-JC)* 5.2 (2020), pp. 1–16.

[4] Jeff J Sandvig, Bamshad Mobasher e Robin Burke. «Robustness of collaborative recommendation based on association rule mining». In: *Proceedings of the 2007 ACM conference on Recommender systems*. 2007, pp. 105–112.

[5] Jiawei Han, Jian Pei e Yiwen Yin. «Mining frequent patterns without candidate generation». In: *ACM sigmod record* 29.2 (2000), pp. 1–12.

[6] Gopal Behera et al. «Hybrid collaborative filtering using matrix factorization and XGBoost for movie recommendation». In: *Computer Standards & Interfaces* 90 (2024), p. 103847.

[7] Dietmar Seipel et al. «Post–mining on Association Rule Bases». In: *European Conference on Artificial Intelligence*. Springer. 2023, pp. 23–35.

[8] Masoud Mansoury et al. «Fairmatch: A graph-based approach for improving aggregate diversity in recommender systems». In: *Proceedings of the 28th ACM conference on user modeling, adaptation and personalization*. 2020, pp. 154–162.