MOvie recommender system

Gruppo di Lavoro: Zagaria Riccardo

Matricola: 760475

A.A. 2024 - 2025

E-mail: [r.zagaria18@studenti.uniba.it](mailto:r.zagaria18@studenti.uniba.it)

Link GitHub: https://github.com/venicebeachl/IconProject.git

Sommario

[1) Introduzione 2](#_Toc188956627)

[1.1) Requisiti funzionali 2](#_Toc188956628)

[2) Creazione del Dataset 3](#_Toc188956629)

[2.2) Descrizione delle Feature 3](#_Toc188956630)

[3) Ricerca 4](#_Toc188956631)

[3.1) Algoritmo Random Walk su CSP 4](#_Toc188956632)

[3.2) Risultati di Precisione e Recall 5](#_Toc188956633)

[3.3) Possibili Miglioramenti 6](#_Toc188956634)

[4) Apprendimento Supervisionato e Ragionamento Probabilistico 7](#_Toc188956635)

[4.1) K-Fold Cross Validation e Random Search 7](#_Toc188956636)

[4.1.1) K-Fold Cross Validation 7](#_Toc188956637)

[4.1.2) Random Search: Ottimizzazione degli Iperparametri 8](#_Toc188956638)

[4.1.3) Parallelizzazione: Velocizzare il Processo 8](#_Toc188956639)

[4.2) Risultati: Precisione, Recall e ROC AUC 8](#_Toc188956640)

[4.2.1) Deviazioni Standard: Stabilità del Modello 8](#_Toc188956641)

[4.2.2) Grafico di Precision e Recall: Bilanciamento tra Accuratezza e Copertura 9](#_Toc188956642)

[4.2.3) Curva ROC: Capacità di Discriminazione 9](#_Toc188956643)

[4.3) Discussione dei Risultati: Prestazioni e Trade-off 10](#_Toc188956644)

[4.3.1) Trade-off tra Precision e Recall 10](#_Toc188956645)

[4.3.2) Ottimizzazione degli Iperparametri: Un Passo Avanti 10](#_Toc188956646)

[4.4) Possibili Miglioramenti: Verso un Modello Ancora Migliore 11](#_Toc188956647)

[5) Confronto delle Soluzioni 12](#_Toc188956648)

[5.1) Soluzione 1: Random Walk su CSP (Constraint Satisfaction Problem) 12](#_Toc188956649)

[5.2) Soluzione 2: K-Fold Cross Validation con Random Search 13](#_Toc188956650)

[5.3) Soluzione Selezionata: K-Fold Cross Validation 13](#_Toc188956651)

[5.4) Sviluppi Futuri 14](#_Toc188956652)

[Riferimenti Bibliografici 14](#_Toc188956653)

# Introduzione

L’obiettivo di questo progetto è quello di sviluppare un Recommender System in grado di consigliare dei film all’utente. L’origine delle raccomandazioni può essere Content-Based, ovvero basato sulla similarità tra due o più prodotti, film in questo caso, o User-Based, ovvero si basa sulla similarità tra due o più utenti.

## Requisiti funzionali

Il progetto è stato realizzato in Python, ad oggi uno dei linguaggi di programmazione più utilizzati, poiché questo linguaggio mette a disposizione numerose librerie che permettono di esaminare facilmente i dati, o di realizzare facilmente dei grafici, utili per analizzare statisticamente i risultati ottenuti da determinate soluzioni.

La versione di Python utilizzata è la versione 3.12.8. L’IDE utilizzato per lo sviluppo del progetto è, invece, Visual Studio Code. Si può, però, utilizzare un IDE qualsiasi.

Tra le librerie di Python utilizzate, troviamo invece:

* **matplotlib**, per la realizzazione dei grafici;
* **numpy**, per la gestione degli array;
* **pandas**, per l’importazione dei dataset nel formato *.csv*;
* **seaborn**, per la realizzazione di altri grafici;
* **sklearn**, per l’apprendimento automatico;
* **cornac**, per la creazione e l'uso di sistemi di recommender system;
* **Imblearn**, per l’oversampling tramite SMOTE;
* **joblib**: per la gestione della parallelizzazione e il salvataggio di modelli;
* **scipy**: per calcoli scientifici, come distribuzioni di probabilità e metriche statistiche.

# Creazione del Dataset

Il dataset utilizzato è il **MovieLens 100K**, che contiene informazioni su utenti, film e valutazioni. I dati sono stati caricati direttamente da file CSV disponibili online e preprocessati per adattarli alle esigenze del progetto. Di seguito sono descritti i passaggi chiave.

## 2.1) Caricamento dei Dati

I dati sono stati caricati da tre file separati:

* **u.user**: Contiene informazioni demografiche degli utenti (UserID, Age, Gender, Occupation, Zip Code).
* **u.item**: Contiene informazioni sui film (ItemID, Title, Release Date, Generi).
* **u.data**: Contiene le valutazioni degli utenti (UserID, ItemID, Rating, Timestamp).

I dati sono stati caricati in tre DataFrame:

* user\_df: Informazioni sugli utenti.
* item\_df: Informazioni sui film, inclusi i generi.
* feedback\_df: Valutazioni degli utenti.

## 2.2) Preprocessing dei Dati

Dopo il caricamento, i dati sono stati preprocessati per garantire la qualità e l'integrità del dataset:

* **Filtraggio degli utenti e dei film**: Sono stati rimossi gli utenti e i film con meno di 5 valutazioni (min\_ratings = 5) per evitare raccomandazioni basate su dati insufficienti.
* **Mappatura degli ID**: Gli ID degli utenti e dei film sono stati mappati in un intervallo continuo per semplificare l'elaborazione successiva.
* **Conversione dei feedback**: Le valutazioni sono state convertite in una lista di tuple (UserID, ItemID, Rating) per facilitare l'uso nei modelli di raccomandazione.

## 2.3) Calcolo della Similarità tra Film (Soluzione 1)

Nella seconda soluzione, è stata calcolata la similarità tra i film utilizzando la **matrice TF-IDF** sui generi dei film. I generi sono stati rappresentati come vettori TF-IDF, e la similarità è stata calcolata utilizzando la **cosine similarity**. Questo approccio permette di identificare film simili in base ai generi, che è uno dei fattori chiave per le raccomandazioni.

## 2.4) Gestione dei Vincoli (Soluzione 1)

Nella soluzione basata su **Random Walk su CSP**, i vincoli sono stati definiti in base alle preferenze dell'utente, come i generi preferiti (ad esempio, "Action", "Comedy", "Drama"). Solo i film che soddisfano questi vincoli sono stati considerati durante il Random Walk.

# Ricerca

L'obiettivo principale della macroarea della ricerca è stato esplorare tecniche basate sulla rappresentazione grafica per identificare film potenzialmente interessanti per gli utenti. Dopo una valutazione approfondita delle soluzioni possibili, è stato scelto un approccio di **Random Walk su CSP** (Constraint Satisfaction Problem), che combina la ricerca su grafo con vincoli basati sulle preferenze dell'utente.

## 3.1) Algoritmo Random Walk su CSP

L'obiettivo della ricerca è stato quello di applicare tecniche di **Constraint Satisfaction Problem (CSP)** per generare raccomandazioni di film personalizzate. In particolare, è stato implementato un **Random Walk su CSP**, dove i vincoli sono rappresentati dalle preferenze dell'utente (ad esempio, i generi preferiti) e la similarità tra i film è utilizzata per guidare l'esplorazione del grafo.

**Vincoli (CSP):**

* **Generi Preferiti**: I vincoli sono definiti in base ai generi cinematografici preferiti dall'utente (ad esempio, "Action", "Comedy", "Drama"). Solo i film che soddisfano questi vincoli vengono considerati durante il Random Walk.
* **Similarità tra Film**: La similarità tra i film è calcolata utilizzando la **cosine similarity** su vettori TF-IDF dei generi. Questo approccio è stato scelto per rappresentare i film in modo strutturato e per guidare l'esplorazione del grafo verso film simili, ma sempre rispettando i vincoli imposti dalle preferenze dell'utente.

**Random Walk su CSP:**

* **Esplorazione Guidata dai Vincoli**: Il Random Walk inizia da un set di film che l'utente ha valutato positivamente e si muove nel grafo seguendo le connessioni tra film simili, ma solo se questi film soddisfano i vincoli (ad esempio, appartengono ai generi preferiti).
* **Bilanciamento tra Esplorazione e Vincoli**: L'algoritmo bilancia l'esplorazione di nuovi film con il rispetto dei vincoli, garantendo che le raccomandazioni siano sia pertinenti che diversificate.

**Perché è un CSP?**

* **Problema di Vincoli**: Il problema di raccomandazione è stato modellato come un CSP, dove i vincoli sono le preferenze dell'utente e l'obiettivo è trovare un insieme di film che soddisfino questi vincoli.
* **Random Walk come Strategia di Ricerca**: Il Random Walk è stato utilizzato come strategia di ricerca per esplorare lo spazio delle soluzioni (i film) in modo efficiente, rispettando i vincoli.

**TF-IDF e Cosine Similarity:**

* **Supporto alla Rappresentazione della Conoscenza**: TF-IDF e cosine similarity sono state utilizzate come strumenti per rappresentare i generi dei film in modo strutturato e per calcolare la similarità tra di essi. Queste tecniche non sono il focus principale del progetto, ma sono state integrate per supportare la risoluzione del CSP.
* **Attinenza con ICon**: L'uso di TF-IDF e cosine similarity è funzionale alla rappresentazione della conoscenza (generi dei film) e alla risoluzione di un problema di CSP.

## 3.2) Risultati di Precisione e Recall

Le performance del metodo sono state valutate utilizzando le metriche **Precision** e **Recall**, che sono utilizzate per confrontare i film raccomandati con quelli effettivamente apprezzati dall'utente. I valori ottenuti sono i seguenti:

* **Precisione**: 0.83
* **Recall**: 0.06

Questi risultati mostrano che il sistema ha un'alta precisione, indicando che circa l'83% dei film raccomandati sono stati effettivamente apprezzati dall'utente. Tuttavia, il valore di **Recall** molto basso (0.06) suggerisce che il sistema non è riuscito a coprire una vasta gamma di film che potrebbero essere di interesse per l'utente. Questo potrebbe essere dovuto al fatto che l'algoritmo si concentra troppo sui film più simili a quelli già apprezzati, senza esplorare sufficientemente nuove possibilità.

Immagine che contiene testo, schermata, Rettangolo, diagramma

Descrizione generata automaticamente

**Discussione dei Risultati**

I risultati mostrano un chiaro trade-off tra precisione e recall. L'introduzione dei vincoli basati sui generi preferiti ha migliorato la precisione, ma ha limitato la varietà delle raccomandazioni, riducendo il recall. Questo potrebbe essere mitigato in futuro introducendo una maggiore flessibilità nei vincoli o esplorando tecniche di bilanciamento tra precisione e recall.

## 3.3) Possibili Miglioramenti

Per migliorare i risultati, una strategia possibile sarebbe quella di integrare altre tecniche di apprendimento supervisionato o di esplorazione più ampia dei grafi, che possano bilanciare meglio la **Precision** e il **Recall**. Un'altra possibilità sarebbe quella di ottimizzare ulteriormente il processo di selezione dei film simili, utilizzando una **normalizzazione** delle similarità o un sistema di **pesatura dinamica** dei generi, in modo da considerare con maggiore attenzione le preferenze espresse dall'utente.

# Apprendimento Supervisionato e Ragionamento Probabilistico

In questa sezione, uniamo i concetti di **Apprendimento Supervisionato** e **Ragionamento Probabilistico** per migliorare le prestazioni del sistema di raccomandazione. L'obiettivo è stato quello di ottimizzare il modello attraverso tecniche avanzate di validazione e ottimizzazione degli iperparametri, garantendo al contempo una valutazione robusta delle prestazioni. Questo approccio ci ha permesso di bilanciare la precisione delle raccomandazioni con la capacità di coprire una vasta gamma di preferenze dell'utente.

## 4.1) K-Fold Cross Validation e Random Search

Per garantire che il modello fosse affidabile e generalizzabile, abbiamo implementato una **K-Fold Cross Validation** combinata con una **Random Search** per l'ottimizzazione degli iperparametri. Questo approccio ci ha permesso di ridurre il rischio di overfitting e di ottenere una stima più accurata delle prestazioni del modello.

### 4.1.1) K-Fold Cross Validation

La K-Fold Cross Validation è una tecnica che suddivide il dataset in KK sottoinsiemi (detti "fold"). Ogni fold viene utilizzato alternativamente come set di test, mentre gli altri K−1K−1 fold vengono usati per l'addestramento. Nel nostro caso, abbiamo scelto K=5K=5, il che significa che il dataset è stato diviso in 5 parti. Il modello è stato addestrato e testato 5 volte, utilizzando ogni volta una parte diversa come set di test. Questo ci ha permesso di valutare le prestazioni del modello in modo più robusto, riducendo la variabilità dei risultati.

### 4.1.2) Random Search: Ottimizzazione degli Iperparametri

Per ottimizzare gli iperparametri del modello, abbiamo utilizzato una **Random Search**. A differenza della Grid Search, che esplora tutte le combinazioni possibili di iperparametri, la Random Search campiona in modo casuale un sottoinsieme di combinazioni. Questo approccio è particolarmente utile quando lo spazio degli iperparametri è ampio, poiché riduce notevolmente i tempi di esecuzione senza sacrificare troppo la qualità dei risultati.

Gli iperparametri che abbiamo ottimizzato includono:

* **k**: Il numero di fattori latenti nel modello di fattorizzazione.
* **max\_iter**: Il numero massimo di iterazioni per l'addestramento.
* **learning\_rate**: Il tasso di apprendimento, che controlla quanto velocemente il modello si adatta ai dati.
* **lambda\_reg**: Il parametro di regolarizzazione, che aiuta a prevenire l'overfitting penalizzando i pesi troppo grandi.

### 4.1.3) Parallelizzazione: Velocizzare il Processo

Per rendere il processo di ottimizzazione più efficiente, abbiamo parallelizzato la Random Search utilizzando la libreria **Joblib**. Questo ci ha permesso di eseguire più valutazioni di modelli in parallelo, riducendo significativamente i tempi di esecuzione. In pratica, invece di valutare una combinazione di iperparametri alla volta, ne abbiamo valutate diverse contemporaneamente, sfruttando al massimo le risorse computazionali disponibili.

## 4.2) Risultati: Precisione, Recall e ROC AUC

Dopo aver eseguito la K-Fold Cross Validation e la Random Search, abbiamo ottenuto i seguenti risultati:

* **Soglia ottimale**: -0.2993412911891937
* **ROC AUC medio**: 0.572
* **Precision media**: 0.850
* **Recall medio**: 0.731

### 4.2.1) Deviazioni Standard: Stabilità del Modello

Le deviazioni standard delle metriche calcolate sui diversi fold della cross-validation sono state:

* **Deviazione standard ROC AUC**: 0.015
* **Deviazione standard Precision**: 0.004
* **Deviazione standard Recall**: 0.080

Queste deviazioni standard relativamente basse indicano che il modello è stabile e le prestazioni sono consistenti tra i diversi fold. In altre parole, il modello non è troppo sensibile alla scelta del set di addestramento e test, il che è un buon segno di robustezza.

### 4.2.2) Grafico di Precision e Recall: Bilanciamento tra Accuratezza e Copertura

Il grafico seguente mostra i valori di **Precision** e **Recall** ottenuti con il miglior set di iperparametri:

Immagine che contiene testo, schermata, diagramma, Rettangolo

Descrizione generata automaticamente

* **Precision**: 0.850
* **Recall**: 0.731

La **precisione** molto alta (0.850) indica che la maggior parte dei film raccomandati sono effettivamente apprezzati dall'utente. Questo è un risultato importante, poiché significa che il sistema è in grado di fare suggerimenti pertinenti. D'altra parte, il **recall** (0.731) ci dice che il modello è anche in grado di coprire una buona parte delle preferenze dell'utente, suggerendo una gamma abbastanza ampia di film.

### 4.2.3) Curva ROC: Capacità di Discriminazione

La curva ROC mostra la capacità del modello di distinguere tra valutazioni positive e negative. L'Area Under Curve (AUC) è stata calcolata come 0.572, che, sebbene non eccellente, è comunque migliore di un modello casuale (AUC = 0.5).

Immagine che contiene testo, linea, Diagramma, diagramma

Descrizione generata automaticamente

La curva ROC conferma che il modello ha una certa capacità di discriminazione, anche se c'è margine di miglioramento. Un AUC più alto indicherebbe una migliore capacità del modello di distinguere tra film apprezzati e non apprezzati.

## 4.3) Discussione dei Risultati: Prestazioni e Trade-off

I risultati ottenuti dimostrano che il modello ha una buona capacità di fare raccomandazioni pertinenti (alta precisione) e di coprire una buona parte delle preferenze dell'utente (recall moderatamente alto). Tuttavia, il valore di ROC AUC suggerisce che il modello potrebbe migliorare nella discriminazione tra valutazioni positive e negative.

### 4.3.1) Trade-off tra Precision e Recall

Uno degli aspetti più interessanti dei sistemi di raccomandazione è il **trade-off tra precision e recall**. Nel nostro caso, il modello ha una precisione molto alta, il che è positivo, ma il recall potrebbe essere migliorato per coprire una gamma più ampia di film che l'utente potrebbe apprezzare. Questo trade-off è comune in molti problemi di machine learning e spesso richiede un bilanciamento tra accuratezza e copertura.

### 4.3.2) Ottimizzazione degli Iperparametri: Un Passo Avanti

L'uso della Random Search ha permesso di trovare una configurazione di iperparametri che bilancia bene le prestazioni del modello. Tuttavia, ulteriori miglioramenti potrebbero essere ottenuti esplorando un numero maggiore di combinazioni o utilizzando tecniche di ottimizzazione più avanzate come la **Bayesian Optimization**, che potrebbe trovare configurazioni ancora migliori in meno tempo.

## 4.4) Possibili Miglioramenti: Verso un Modello Ancora Migliore

Sebbene i risultati siano già soddisfacenti, ci sono diverse direzioni in cui il modello potrebbe essere migliorato:

* **Aumento del numero di iterazioni**: Esplorare un numero maggiore di combinazioni di iperparametri potrebbe portare a una configurazione ancora migliore.
* **Modelli più complessi**: Utilizzare modelli più avanzati come reti neurali o modelli di fattorizzazione più sofisticati potrebbe migliorare ulteriormente le prestazioni.
* **Espansione del dataset**: Utilizzare un dataset più grande, come MovieLens 1M o 10M, potrebbe migliorare la generalizzazione del modello.
* **Ottimizzazione avanzata**: Tecniche come la Bayesian Optimization potrebbero rendere la ricerca degli iperparametri più efficiente ed efficace.

# Confronto delle Soluzioni

In questo capitolo, confrontiamo le diverse soluzioni implementate nel progetto, analizzandole alla luce delle macroaree del corso di "Ingegneria della Conoscenza": **Ricerca**, **Apprendimento Supervisionato** e **Ragionamento (Logico e Probabilistico)**. Ogni soluzione è stata valutata in base a metriche standard come **precisione** e **recall**, ma anche in termini di aderenza ai principi teorici e metodologici affrontati durante il corso.ù

## 5.1) Soluzione 1: Random Walk su CSP (Constraint Satisfaction Problem)

**Descrizione**

La prima soluzione implementata si basa su un approccio di **Ricerca su Grafo** con **Vincoli (CSP)**, utilizzando un algoritmo di **Random Walk** per esplorare lo spazio delle soluzioni. Questo metodo combina tecniche di ricerca informata con vincoli derivati dalle preferenze dell'utente, come i generi cinematografici preferiti. La similarità tra i film è stata calcolata utilizzando la **cosine similarity** su vettori TF-IDF dei generi, un approccio che rientra nella **Rappresentazione della Conoscenza** e nella **Risoluzione di CSP**.

**Punti di Forza**

* **Ricerca Guidata da Vincoli**: L'algoritmo sfrutta i vincoli derivati dalle preferenze dell'utente per guidare la ricerca, garantendo che le raccomandazioni siano pertinenti.
* **Esplorazione del Grafo**: Utilizzando un Random Walk, il sistema esplora il grafo dei film in modo efficiente, bilanciando esplorazione e sfruttamento delle informazioni disponibili.
* **Alta Precisione**: Con una precisione di **0.83**, il sistema dimostra di essere in grado di fare raccomandazioni altamente accurate.

**Limitazioni**

* **Recall Basso**: Il valore di recall è risultato molto basso (**0.06**), indicando che il sistema non copre una vasta gamma di film potenzialmente interessanti per l'utente. Questo è dovuto alla forte focalizzazione sui vincoli, che limita l'esplorazione di nuove possibilità.
* **Esplorazione Limitata**: L'algoritmo tende a concentrarsi troppo sui film più simili a quelli già apprezzati, senza esplorare sufficientemente nuove aree dello spazio delle soluzioni.

## 5.2) Soluzione 2: K-Fold Cross Validation con Random Search

**Descrizione**

La seconda soluzione implementata si basa su tecniche di **Apprendimento Supervisionato**, utilizzando una **K-Fold Cross Validation** combinata con una **Random Search** per l'ottimizzazione degli iperparametri. Questo approccio permette di bilanciare la precisione delle raccomandazioni con la capacità di coprire una vasta gamma di preferenze dell'utente, sfruttando modelli di apprendimento automatico.

**Punti di Forza**

* **Robustezza Metodologica**: La K-Fold Cross Validation garantisce una stima robusta delle prestazioni del modello, riducendo il rischio di overfitting.
* **Ottimizzazione degli Iperparametri**: La Random Search permette di esplorare un ampio spazio di iperparametri in modo efficiente, migliorando le prestazioni del modello.
* **Bilanciamento tra Precisione e Recall**: Con una precisione di **0.85** e un recall di **0.73**, il sistema dimostra un buon equilibrio tra accuratezza e copertura delle preferenze.

**Limitazioni**

* **Risorse Computazionali**: L'ottimizzazione degli iperparametri richiede risorse computazionali significative, soprattutto quando si utilizza la parallelizzazione.
* **Margine di Miglioramento**: Sebbene i risultati siano soddisfacenti, il valore di ROC AUC (**0.572**) suggerisce che il modello potrebbe migliorare nella discriminazione tra valutazioni positive e negative.

## 5.3) Soluzione Selezionata: K-Fold Cross Validation

**Motivazioni della Scelta**

La soluzione basata su **K-Fold Cross Validation** è stata selezionata come la migliore per il sistema di raccomandazione, grazie ai seguenti motivi:

* **Equilibrio tra Precisione e Recall**: L'alta precisione (**0.85**) garantisce raccomandazioni pertinenti, mentre un recall moderatamente alto (**0.73**) assicura una buona copertura delle preferenze.
* **Robustezza Metodologica**: La validazione incrociata riduce il rischio di overfitting, fornendo una stima accurata delle prestazioni del modello.
* **Flessibilità**: Il metodo può essere ulteriormente migliorato integrando feature più complesse o modelli avanzati, come reti neurali o ensemble di modelli.

## 5.4) Sviluppi Futuri

Per migliorare ulteriormente il sistema di raccomandazione, sono possibili diverse direzioni di sviluppo, tra cui:

* **Feature Engineering Avanzato**: Integrare nuove variabili, come informazioni contestuali o dati derivati da fonti esterne (ad esempio, recensioni o classifiche di film).
* **Ottimizzazione Avanzata**: Utilizzare tecniche come la **Bayesian Optimization** per migliorare ulteriormente le prestazioni del modello.
* **Espansione del Dataset**: Integrare dataset più grandi, come MovieLens 1M o 10M, per aumentare la generalizzazione del modello.
* **Modelli Avanzati**: Sperimentare con algoritmi di apprendimento profondo, come reti neurali, per catturare relazioni complesse tra le feature.

# Riferimenti Bibliografici

**Ricerca**: D. Poole, A. Mackworth: Artificial Intelligence: Foundations of Computational Agents. 3/e. Cambridge University Press (Ch. 3)

**Apprendimento** **Supervisionato**: D. Poole, A. Mackworth: Artificial Intelligence: Foundations of Computational Agents. 3/e. Cambridge University Press (Ch. 7)

**Ragionamento** **Logico**: D. Poole, A. Mackworth: Artificial Intelligence: Foundations of Computational Agents. 3/e. Cambridge University Press (Ch. 5)

**Ragionamento** **Probabilistico**: D. Poole, A. Mackworth: Artificial Intelligence: Foundations of Computational Agents. 3/e. Cambridge University Press (Ch. 9-10)