UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

TMDbIcon

Raccomandazione di film basata sulla comunità di TMDB

Gruppo di lavoro

- Francesco Papagno, 735596, f.papagno3@studenti.uniba.it
- Piersilvio Spicoli, 736519, p.spicoli@studenti.uniba.it
- Danilo Brugnola, 745960, d.brugnola@studenti.uniba.it

URL Report: https://github.com/Gli-Amari/TMDbIcon

AA 2022-2023

Introduzione

L'industria dei servizi online e del commercio elettronico ha sperimentato una rapida crescita, portando a una vasta quantità di dati e prodotti disponibili per gli utenti. In questo contesto, i sistemi di raccomandazione hanno acquisito crescente importanza, fornendo suggerimenti personalizzati agli utenti per migliorare l'esperienza di navigazione e l'engagement. La raccomandazione è una delle task chiave nell'ambito dell'apprendimento automatico, mirata a prevedere le preferenze degli utenti su elementi come prodotti, film, musica o contenuti digitali.

L'attuale **stato dell'arte** nel campo dei modelli di raccomandazione si basa ampiamente su tecniche avanzate di machine learning. Tra queste, i modelli di **Collaborative Filtering (CF)** sfruttano i pattern di comportamento degli utenti o delle entità simili per fare previsioni (U2U e I2I), mentre le tecniche di **Content-Based Filtering (CBF)** analizzano le caratteristiche degli elementi raccomandati e i profili degli utenti per generare raccomandazioni personalizzate.

La domanda che noi membri del team ci siamo posti è la seguente:

è possibile realizzare e implementare un task di raccomandazione usando strategie ben diverse da quelle offerte dallo stato d'arte?

Il presente progetto propone un approccio innovativo alla task di raccomandazione **attraverso un processo di inferenza guidato da algoritmi di machine learning regressori**. Piuttosto che affidarsi agli standard dell'attuale stato dell'arte dei modelli di raccomandazione, il nostro approccio si basa sull'inferenza dei gusti e delle preferenze degli utenti (che appartengono nel nostro caso alla community di TMDB) utilizzando algoritmi di regressione. Questo approccio permette una comprensione più profonda e granulare dei comportamenti degli utenti, consentendo di offrire raccomandazioni altamente personalizzate e adattabili. E, oltre questo, per comprendere appieno le potenzialità offerte dalle tecniche implementate da noi stessi.

In questo documento verranno illustrate, pertanto, le fasi per la realizzazione del progetto TMDB ICON relativo al corso di studi Ingegneria della Conoscenza a.a. 2022/2023. Per comprendere appieno il problema di raccomandazione, abbiamo dovuto riformulare il problema come se fosse un **problema di ranking basato su stime numeriche** che ogni film deve avere e successivamente, ordinare tale ranking. Per poter realizzare e offrire le soluzioni a questo tipo di problema, sono state formulate due task principali a fini sperimentali.

Task 1: Ranking di preferenze generiche per l'intera community del problema.

In particolare

1. **Inferenza delle Preferenze degli Utenti.** Per prima cosa, la task iniziale coinvolge l'inferenza delle preferenze degli utenti basandosi sui dati storici della community nel loro insieme con l'obiettivo di comprendere i modelli di comportamento comuni che emergono dall'intera community. Utilizzando, successivamente, algoritmi di machine learning regressori, si cerca di identificare tendenze e relazioni generalizzate tra gli elementi raccomandati e la community nel suo complesso.
2. **Generazione di Raccomandazioni Generiche per la Community**. Una volta che i modelli regressori sono stati addestrati sull'inferenza delle preferenze degli utenti nella community, la task successiva consiste nella generazione del ranking per gli elementi raccomandati. I modelli regressori, come Support Vector Regression (SVR), Stocastic Gradient Descent Regressor (SGDR) e Linear Regression, vengono utilizzati per prevedere i valori di rating di preferenza per ciascun elemento. Queste previsioni sono poi utilizzate per ordinare gli elementi in un ranking, in modo che gli elementi con le previsioni più alte vengano posizionati in cima al ranking, indicando un maggiore grado di preferenza stimato dalla community.

Task 2: Generazione del Ranking basato sulle Preferenze Utente tramite Constraint Optimization Problem.

Nel task 2, il sistema richiede all'utente di fornire le proprie preferenze per gli elementi specifici. Queste preferenze possono essere espresse in vari modi, ad esempio attraverso valutazioni numeriche, like/dislike, o classificazioni qualitative. Una volta ottenute le preferenze dall'utente, il sistema formulare un problema di ottimizzazione con vincoli per generare un ranking personalizzato.

Il problema di ottimizzazione con vincoli prende in considerazione le preferenze fornite dall'utente come vincoli. Ad esempio, se l'utente indica che preferisce l'elemento A rispetto all'elemento B, il sistema incorpora questa preferenza come un vincolo nel problema di ottimizzazione. Il ranking viene quindi generato ottimizzando le preferenze fornite dall'utente insieme a eventuali altri vincoli specifici, come il budget massimo o il numero massimo di elementi da visualizzare.

Il risultato di questo processo di ottimizzazione sarà un ranking personalizzato basato sulle preferenze esplicitamente indicate dall'utente. Gli elementi vengono ordinati nel ranking in base a quanto soddisfano le preferenze e i vincoli specifici forniti dall'utente.
Questo approccio mette l'utente al centro del processo decisionale, consentendo una personalizzazione estrema delle raccomandazioni basate sulle preferenze espresse, garantendo così che il sistema fornisca raccomandazioni altamente rilevanti e soddisfacenti per l'utente in questione.

Task 3: Classificazione delle preferenze dei film.
Invece di trattare il problema come una regressione o come un task di raking, adotteremo un ottica di classificazione delle preferenze basate su una colonna che indica se un film piace o meno.
L'obiettivo di questa task è classificare i film in base alle nostre preferenze, determinando se un film è gradito o meno, utilizzando le informazioni presenti nei dati relativi ai film.

Nel nostro contesto, ci concentriamo sulla creazione di un sistema di raccomandazione per gli utenti della piattaforma TMDB (The Movie Database). TMDB è una vasta e diversificata piattaforma online che offre informazioni dettagliate sui film, le serie TV e le persone del mondo dell'intrattenimento. Gli utenti di TMDB possono esplorare una vasta gamma di contenuti, dalle ultime uscite cinematografiche ai classici del cinema, e possono anche contribuire attraverso recensioni, valutazioni e discussioni sulla community.
Attraverso l'utilizzo di avanzati modelli di machine learning, vogliamo creare raccomandazioni pertinenti e adattabili, basate sulle preferenze e le interazioni degli utenti, migliorando così la soddisfazione e l'engagement degli utenti sulla piattaforma TMDB. Con l'accesso a una vasta quantità di dati sulle preferenze degli utenti e i dettagli dei contenuti, TMDB offre un contesto ricco e stimolante per affrontare questa sfida, contribuendo a migliorare significativamente l'esperienza di esplorazione e scoperta dei suoi utenti.

Elenco argomenti di interesse

- Processing e data exploration
- Knowledge Base inference
- Machine Learning Regression
- Reasoning on Constraint optimization problem
- Machine Learning Classification

Processing & data exploration

Le fasi di processing e data exploration sono fondamentali nel processo di sviluppo di un sistema di raccomandazione basato su dati. Queste fasi consentono di preparare e comprendere i dati in modo approfondito prima di applicare modelli di machine learning.
La **fase di processing** possiamo suddividerla nelle seguenti fasi:

1. *Raccolta dei Dati*: In questa fase, i dati da diverse fonti vengono raccolti e aggregati. Per un sistema di raccomandazione su TMDB, i dati includono informazioni sui film (come titolo, genere, attori, regista, valutazioni degli utenti, etc.) e dati sugli utenti (come preferenze, valutazioni e recensioni).
   Le features con le quali abbiamo lavorato sono state fornite da due dataset: tmdb_5000_credits.csv e *tmdb_5000_movies.csv.*
   Il dataset tmdb_5000_credits.csv contiene le seguenti features:
   - *movie_id: feature discreta che contiene l'ID univoco per ogni film*
   - *title: feature che contiene una stringa che descrive il nome del film;*

- *cast: contiene diverse informazioni riguardanti il cast del film;*
- *crew: contiene diverse informazioni riguardanti la crew di produzione del film;*

Mentre, il dataset *tmdb_5000_movies.csv* contiene le seguenti features:

- *budget: feature discreta che* contiene il budget utilizzato per realizzare il film;
- *genres: feature discreta che* contiene i generi del film;
- *hompage: feature che* contiene il sito web del film;
- *id: feature discreta che* contiene l'ID del film;
- *keywords: feature che* contiene alcune parole chiave inerenti al film;
- *original_language: feature che* contiene la lingua originale del film
- *original_title: feature che* contiene il titolo originale del film
- *overview:feature che* contiene la trama del film
- *popularity: feature discreta che* contiene un valore che indica la popolarità del film
- *production_companies:feature che* contiene le compagnie di produzione del film
- *production_countries: feature che* contiene i continenti nella quale il film è stato prodotto
- *release_date: feature che* contiene la data di rilascio de film
- *revenue:*
- *runtime: feature che* contiene la durata del film indicata in minuti
- *spoken_languages: feature che* contiene le lingue che vengono parlate nel film
- *status: feature che* indica se il film è stato rilasciato o no
- *tagline: feature che* contiene un ulteriore tag del film
- *title: feature che* contiene il titolo del film in lingua inglese
- *vote_average: feature che* indica la media dei voti attribuiti al film
- *vote_count:feature che* indica il numero di voti che il film ha ricevuto

2. *Pulizia dei Dati*: I dati possono essere sporchi o incompleti. La pulizia dei dati comporta l'individuazione e la correzione di errori, la rimozione di valori nulli e la standardizzazione dei formati per garantire la coerenza dei dati. Dopo questa fase, abbiamo unito i due dataset ottenendo un unico dataset denominato *datasetMerged* sul quale verrà applicata l'operazione di scalatura e normalizzazione. Durante la fase di pulizia, alcune feature come ad esempio *genres, keywords, production_countries, spoken_languages e roduction_companies*, sono state convertire in formato string in quanto erano in formato JSON, e per farlo, abbiamo specificato le keys che erano necessario al nostro dominio. In modo simile è stato svolto lo stesso processo con *crew e cast*.

	original_title	original_language	overview	genres	cast	keywords	director	status	vote_average	vote_count	popularity
0	Avatar	en	In the 22nd century, a paraplegic Marine is di...	['Action', 'Adventure', 'Fantasy', 'ScienceFic...	['Sam Worthington', 'Zoe Saldana', 'Sigourney ...	['culture clash', 'future', 'space war', 'spac...	James Cameron	Released	7.2	11800	150.437577
1	Pirates of the Caribbean: At World's End	en	Captain Barbossa, long believed to be dead, ha...	['Action', 'Adventure', 'Fantasy']	['Johnny Depp', 'Orlando Bloom', 'Keira Knight...	['ocean', 'drug abuse', 'exotic island', 'east...	Gore Verbinski	Released	6.9	4500	139.082615
2	Spectre	en	A cryptic message from Bond's past sends him o...	['Action', 'Adventure', 'Crime']	['Daniel Craig', 'Christoph Waltz', 'Léa Seydo...	['spy', 'based on novel', 'secret agent', 'seq...	Sam Mendes	Released	6.3	4466	107.376788
3	The Dark Knight Rises	en	Following the death of District Attorney Harve...	['Action', 'Crime', 'Drama', 'Thriller']	['Christian Bale', 'Michael Caine', 'Gary Oldm...	['dc comics', 'crime fighter', 'terrorist', 's...	Christopher Nolan	Released	7.6	9106	112.312950

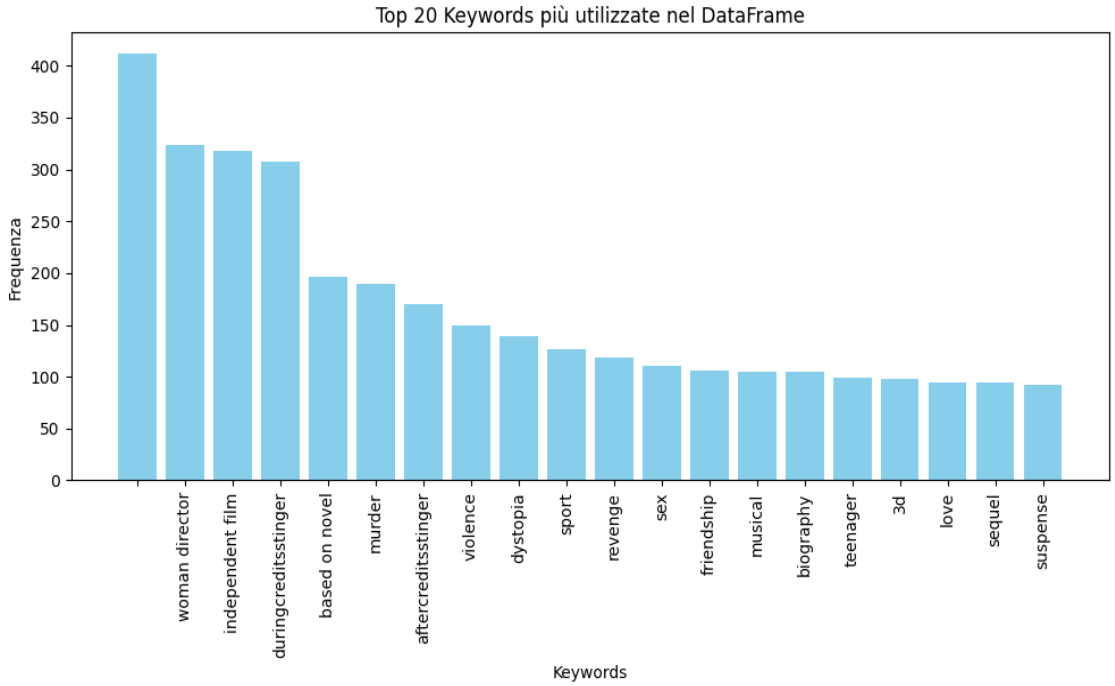
3. *Scalatura e Normalizzazione*: Le variabili possono avere scale diverse. Per garantire che i modelli di machine learning funzionino correttamente, è spesso necessario scalare o normalizzare le features in modo che abbiano una scala comune. Sulla base del dataset precedente, abbiamo generato un nuovo dataset normalizzato e scalato che ci servirà per svolgere le task progettuali. La tecnica di normalizzazione che abbiamo adottato sono
   - *minMaxScaler* il quale ci ha permesso di scalare tutti i valori numerici in un range prefissato. Nel nostro contesto, abbiamo applicato il minMaxScaler in un intervallo $[0,1]$.

- *LabelEncoder* per assegnare ad ogni label presente nel dataset un valore $x \in [0,1]$ successivamente scalato con MinMaxScaler

Dopo aver svolto questo processo, abbiamo salvato i risultati in un dataset nuovo denominato *normalizedDataset* dalla forma

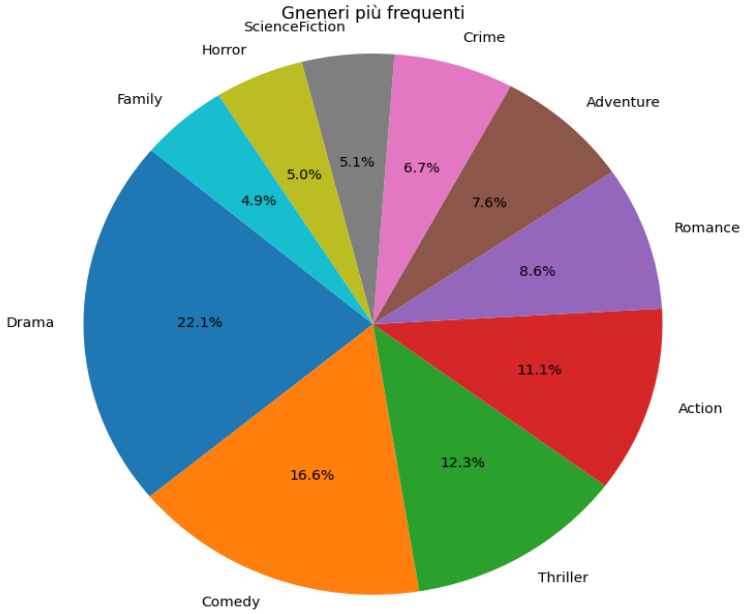
	original_title	original_language	overview	genres	cast	keywords	director	status	vote_average	vote_count	popularity
0	Avatar	0.194	In the 22nd century, a paraplegic Marine is di...	0.129	0.837	0.243	0.377	1	0.72	0.858	0.172
1	Pirates of the Caribbean: At World's End	0.194	Captain Barbossa, long believed to be dead, ha...	0.122	0.482	0.652	0.327	1	0.69	0.327	0.159
2	Spectre	0.194	A cryptic message from Bond's past sends him o...	0.064	0.203	0.862	0.839	1	0.63	0.325	0.123
3	The Dark Knight Rises	0.194	Following the death of District Attorney Harve...	0.254	0.168	0.255	0.153	1	0.76	0.662	0.128
4	John Carter	0.194	John Carter is a war-weary, former military ca...	0.157	0.912	0.086	0.040	1	0.61	0.154	0.050
5	Spider-Man 3	0.194	The seemingly invincible Spider-Man goes up ag...	0.122	0.926	0.286	0.840	1	0.59	0.260	0.132
6	Tangled	0.194	When the kingdom's most wanted-and most charmi...	0.557	0.998	0.431	0.116	1	0.74	0.242	0.056
7	Avengers: Age of Ultron	0.194	When Tony Stark tries to jumpstart a dormant p...	0.157	0.796	0.561	0.501	1	0.73	0.492	0.153
8	Harry Potter and the Half-Blood Prince	0.194	As Harry begins his sixth year at Hogwarts, he...	0.490	0.207	0.985	0.226	1	0.74	0.385	0.113
9	Batman v Superman: Dawn of Justice	0.194	Fearing the actions of a god-like Super Hero l...	0.122	0.077	0.257	0.995	1	0.57	0.509	0.178

La **Data Exploration** è una fase fondamentale nel processo di analisi dei dati che coinvolge l'ispezione, l'analisi e l'interpretazione dei dati disponibili. Questa fase ha l'obiettivo di ottenere una comprensione approfondita delle caratteristiche dei dati e delle loro distribuzioni. L'esplorazione dei dati è essenziale per prendere decisioni informate sulle analisi successive, per individuare pattern, outlier e tendenze nei dati, e per preparare i dati per l'analisi e la modellazione.

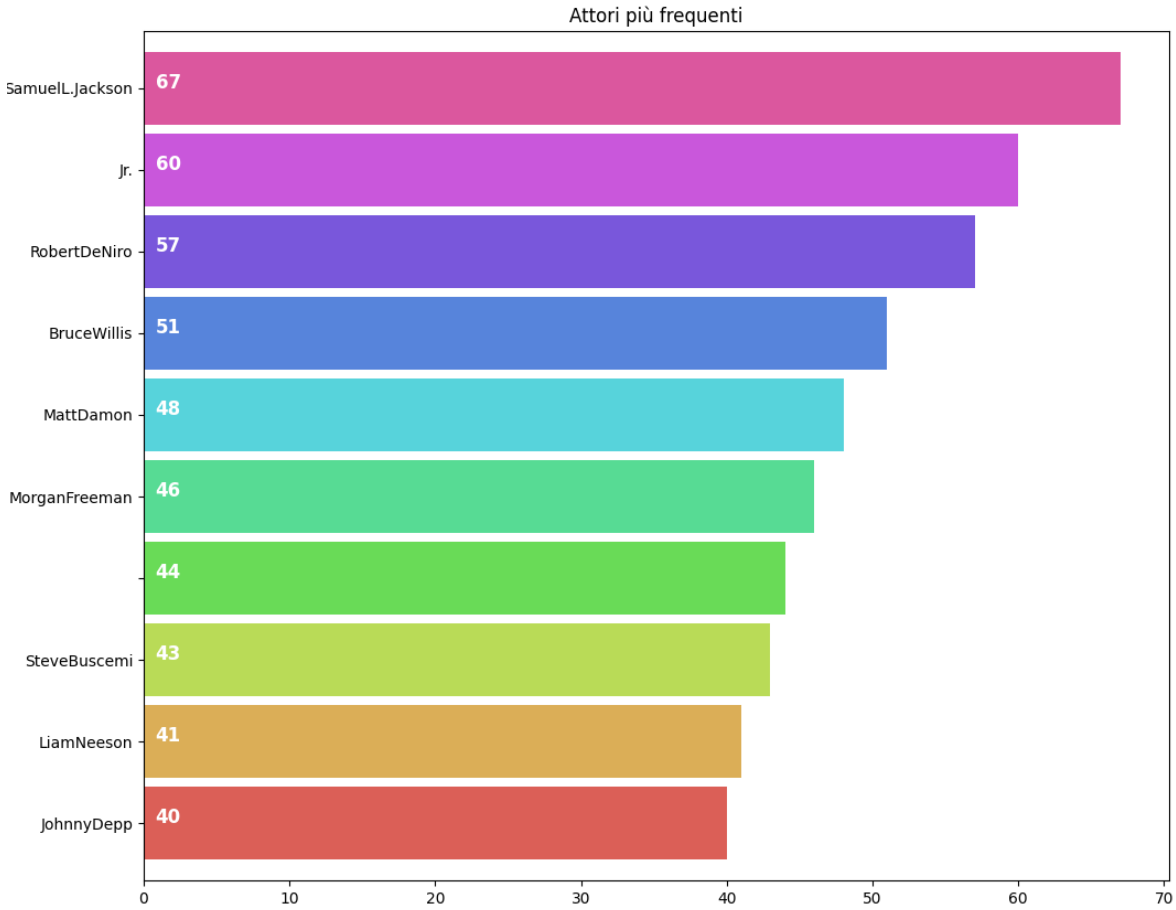


Il grafico a barre, sopra riportato, presenta le parole chiavi 20 più frequenti utilizzate nelle descrizioni dei film. “woman director” spicca come la parola chiave principale, con una frequenza di utilizzo pari a 400, mentre la

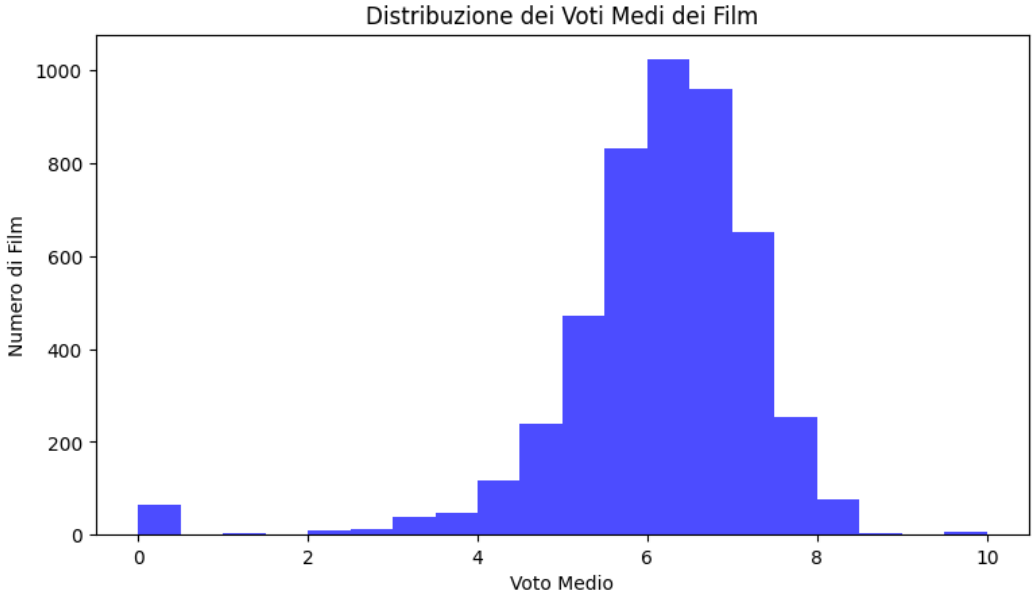
parola chiave meno utilizzata è “suspense” con un valore di frequenza vicino a 100.



Il grafico a torta, sopra illustrato, rappresenta le preferenze cinematografiche della community di TMDB. I film di genere Drammatico risultano essere i preferiti, con il 22,1% delle preferenze, seguito dai film Comedy con una percentuale di gradimento pari al 16,6%. Mentre il genere meno gradito è Family con una percentuale di gradimento pari al 4,9%
Questa distribuzione offre un’indicazione chiara delle preferenze cinematografiche dominanti all’interno della community TMDB.



Il grafico a barre evidenzia i 10 attori più frequentemente presenti all'interno del dataset analizzato. Samuel Jackson si distingue come l'attore con il maggior numero di apparizioni, avendo recitato in ben 67 film. L'attore con il minor numero di apparizione nella top 10 è, invece, Johnny Depp con ben 40 apparizioni.
Questa rappresentazione visiva offre una chiara panoramica degli attori più frequenti nell'industria cinematografica, evidenziando la loro frequenza nelle produzioni filmiche.



L'istogramma sopra riportato rappresenta la distribuzione dei voti medi dei film nel dataset analizzato.
Il grafico mostra la distribuzione di vote_average dei film all'interno del dataset. L'asse x rappresenta i diversi

intervalli di voto, mentre l'asse y indica il numero di film presenti in ciascun intervallo. La barra blu rappresenta la frequenza di film per ogni intervallo di vote_average. Dal grafico possiamo che la distribuzione dei voti forma una "campana", ciò ci suggerisce una distribuzione "normale" dei voti con picco nell'intervallo tra 6 e 7.

Knowledge Base

Per la prima task progettuale, abbiamo creato una regola inferenziale attraverso l'uso della programmazione logica in Prolog. Per calcolare il rating di ogni film, abbiamo voluto inferenziare direttamente la regola su *normalizedDataset* andando a creare una feature target denotata come *ratio_likeable* che conterrà per ogni tupla di film presente nel dataset un valore $x \in [0,1]$ di gradimento. Per inferenziare questo valore, abbiamo assegnato tre bias per ogni feature su cui doveva essere calcolata, ovvero *vote_count, vote_average e popularity* e successivamente è stata calcolata la loro combinazione lineare. Una volta determinata la combinazione lineare i-esima della tupla corrente, si è deciso di passare in input ad un'altra regola (chiamata in modo ricorsivo nella stessa in cui si è calcola la combinazione lineare) che normalizza il valore della combinazione attraverso la *funzione sigmoide* in intervallo [0,1]. Formalmente la regola è descritta come segue

$$ratio(VoteAv, VoteC, Pop, Rating) \ :- \\ RawRating \ is \ (VoteC * 0.5 + VoteAv * 0.5 + Pop * 0.7) \wedge \\ sigmoide(RawRating, Rating).$$

$$sigmoide(X, Y) \ :- \\ Y \ is \ 1/(1 + exp(-X))).$$

Per avere un ranking più coerente, si è deciso di pesare maggiormente la popolarità di un film piuttosto che il comportamento di rating svolto dagli utenti della commmunity. Una volta che la feature *ratio_likeable* è stata ingegnerizzata, si è voluto stampare a video il ranking dei primi 10 film raccomandati per l'intera community e si è voluto salvarlo in un file csv per poterlo confrontare con i ranking offerti dai modelli di apprendimento e COP.

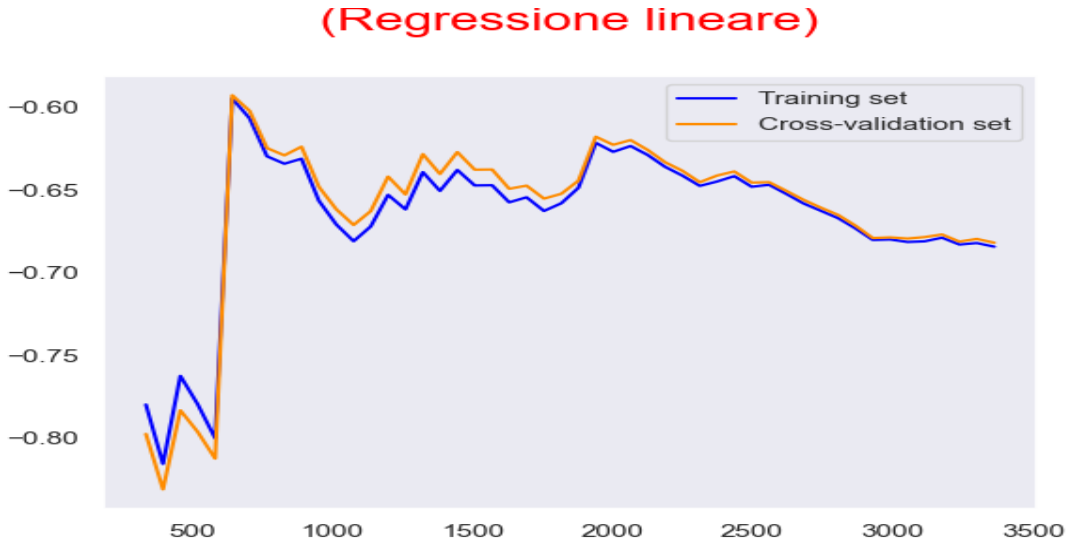
	original_title	ratio_likeable
0	Avatar	0.712877
1	Pirates of the Caribbean: At World's End	0.650173
2	Spectre	0.637285
3	The Dark Knight Rises	0.690103
4	John Carter	0.602765
5	Spider-Man 3	0.626540
6	Tangled	0.629530
7	Avengers: Age of Ultron	0.672188
8	Harry Potter and the Half-Blood Prince	0.655115
9	Batman v Superman: Dawn of Justice	0.660181

Da questo ranking si può osservare come sia abbastanza coerente in termini di risultati ottenuti in quanto sia altamente probabile che tali film possano piacere all'intera community di TMDB. La regola è presente in un file prolog *supervized_KB.pl.*

MACHINE LEARNING REGRESSION

- **Linear Regression:** è un modello di machine learning utilizzato per analizzare la relazione tra una variabile dipendente e uno o più variabili indipendenti. L'obiettivo della regressione lineare è trovare la linea (o iperpiano) che approssima al meglio la relazione tra le variabili.

All'interno della classe **LineareRegressionModel** andiamo a chiamare il metodo **evaluate_model** che prende come parametro un seed, ossia un valore che inizializza il generatore di numeri casuali.
All'interno di evaluate_model definiamo un parametro **param_Grid** che contiene una griglia di parametri da testare per la regressione lineare. Utilizziamo la **StratifielShuffleSplit** per creare cinque fold di cross-validation.
Effettuiamo una ricerca della griglia **GridSearchCV** per trovare i migliori parametri per il modello di regressione lineare utilizzando i set di addestramento (x_train, y_train).
Una volta che abbiamo trovato i migliori parametri creiamo un modello con questi e attraverso una curva di apprendimento valutiamo le performance del modello su dati di addestramento e validazione.
Tramite pilot mostriamo la curva di apprendimento:




```
R2 Score della Regressione lineare : 0.365273606949468
MAE:  0.044924494174370166
MSE:  0.008258701951734343
RMSE:  0.09087740066559091
```

Le metriche di valutazione sopra riportate indicano:
R2: coefficiente di determinazione, misura quanto bene il modello di regressione lineare si adatta ai dati, circa il 36%;
MAE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore merio del 0,04%;
MSE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore quadratico medio dello 0,008%;
RMSE: è la radice quadrata del MSE e serve per interpretare l'errore in termini di scala delle variabili, circa lo 0,09%

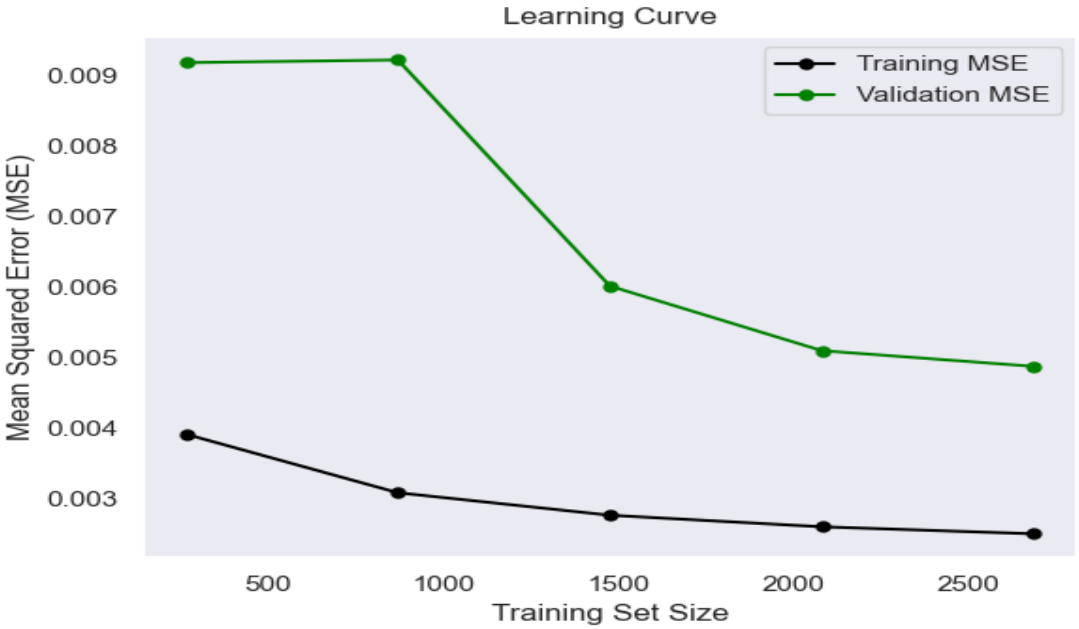
OSS: I risultati della regressione lineare mostrano una precisione inferiore e una capacità più limitata nel modellare la relazione tra le variabili.

Addestriamo il modello migliore trovato con i dati di addestramento completi (x_train, y_train) predicati sui valori x_test e valutiamo le performance del modello.
Risultato di questo addestramento è l'aggiunta di una nuova colonna **predict_rank** al dataframe **df_apart** con i valori predetti.
Infine, mostriamo una lista dei primi 10 film predetti in ordine decrescente rispetto alla previsione:

original_title	overview	Predicted
Minions	Minions Stuart, Kevin and Bob are recruited by...	1.085633
Mad Max: Fury Road	An apocalyptic story set in the furthest reach...	1.067428
Avatar	In the 22nd century, a paraplegic Marine is di...	1.016641
Jurassic World	Twenty-two years after the events of Jurassic ...	0.991666
The Dark Knight	Batman raises the stakes in his war on crime. ...	0.989420
Interstellar	Interstellar chronicles the adventures of a gr...	0.986666
The Avengers	When an unexpected enemy emerges and threatens...	0.975772
Guardians of the Galaxy	Light years from Earth, 26 years after being a...	0.957720
Inception	Cobb, a skilled thief who commits corporate es...	0.955074
Deadpool	Deadpool tells the origin story of former Spec...	0.922308

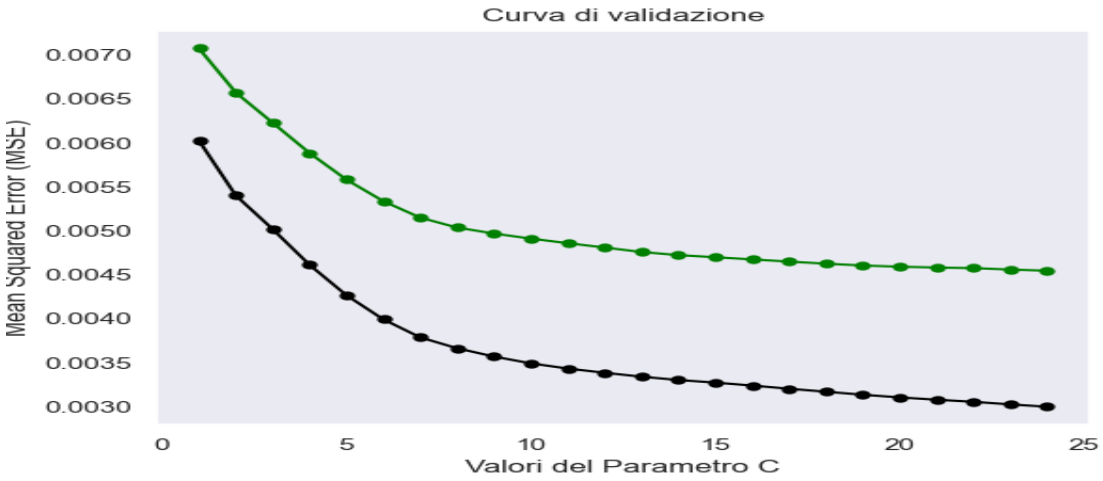
- **SVR (Support Vector Regression):** La regressione vettoriale di supporto è una tecnica di apprendimento automatico utilizzata per la regressione, cioè la predizione di valori numerici. È un'estensione del Support Vector Machine (SVM) utilizzato per problemi di classificazione. L'obiettivo principale di SVR è adattare una funzione che approssimi al meglio i dati disponibili. SVR cerca di trovare una funzione o un iperpiano in uno spazio di alta dimensionalità che massimizzi la larghezza della "fascia" (margin) attorno al limite di supporto (support vectors) e allo stesso tempo mantenga gli errori di predizione entro un certo margine di tolleranza, chiamato "tube".

  All'interno della classe SVRmodel verrà richiamato il metodo **evaluate_model** che prende come input un parametro seed, ossia un valore che inizializza il generatore di numeri casuali.
  Viene definitoun dizionario **grid_par** che contiene le combinazioni dei parametri per la ricerca nella griglia, utilizzando **GridSearchCV** per trovare i parametri migliori per l'SVR utilizzando la combinazione specificata e il numero di fold di cross-validation.
  Dopo che i parametri migliori sono stati trovati, viene creato un modello con questi parametri e viene realizzata una curva di apprendimento (**learning_curve**) per poter valutare le performance del modello in termini di errore quadratico medio (MSE). Otteniamo la seguente curva:



  Viene poi utilizzata una curva di validazione (**validation_curve**) per valutare il parametro **C** presente nella **grid_par** influisce sul MSE sia su dati di addestramento che validazione, stampando poi i risultati

della curva. Otteniamo la seguente curva:




```
MAE : 0.04093649719441135
MSE : 0.0038156165683298346
RMSE: 0.0617706772532877
```

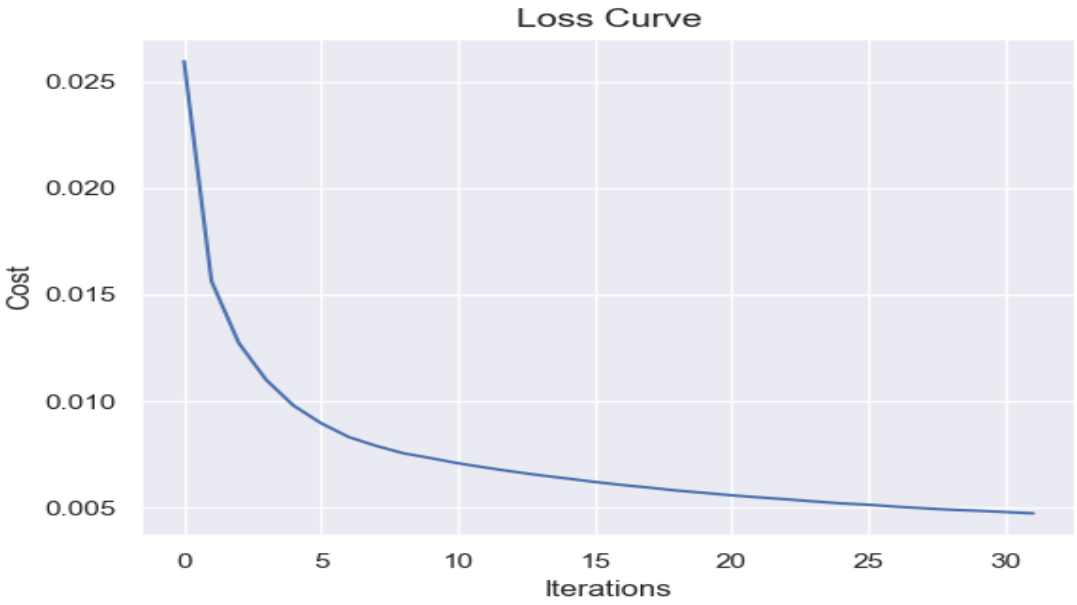
Le metriche sopra riportate indicano:
MAE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore merio del 0,04%;
MSE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore quadratico medio dello 0,003%;
RMSE: è la radice quadrata del MSE e serve per interpretare l'errore in termini di scala delle variabili, circa lo 0,06%

OSS: In generale, valutando questi risultati, sembrerebbe che il modello SVR abbia prestazioni abbastanza buone, specialmente se stai affrontando un problema in cui queste differenze rappresentano una buona adattabilità alle tue esigenze.

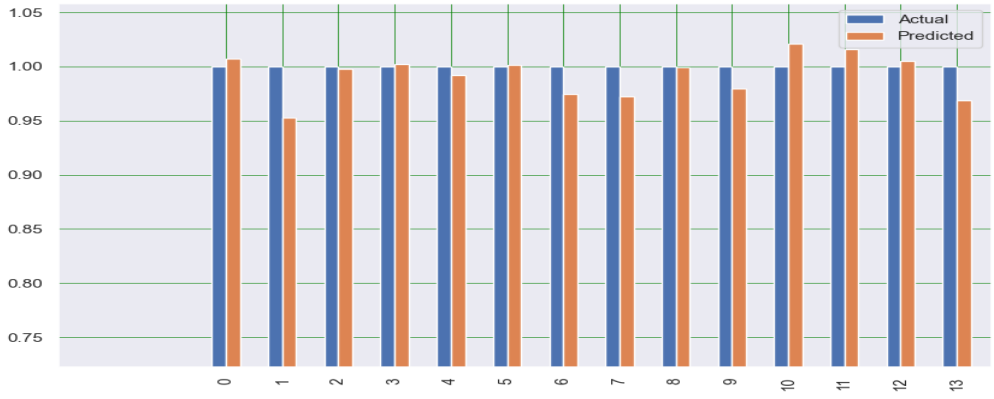
Viene poi addestrato il modello migliore trovato con i dati di addestramento completi (**x_train** e **y_train**) predicendo i valori su **x_test** e valutando la performance del modello usando l'errore medio assoluto (MAE), l'errore quadratico medio (MSE) e la radice quadrata dell'errore quadratico medio (RMSE).
Infine, mostriamo una lista dei primi 10 film predetti in ordine decrescente rispetto alla previsione:

original_title	overview	Predicted
Avatar	In the 22nd century, a paraplegic Marine is di...	1.035415
Jurassic World	Twenty-two years after the events of Jurassic ...	1.030245
Inception	Cobb, a skilled thief who commits corporate es...	1.028956
Deadpool	Deadpool tells the origin story of former Spec...	1.026385
The Dark Knight	Batman raises the stakes in his war on crime. ...	1.026146
Minions	Minions Stuart, Kevin and Bob are recruited by...	0.996804
Guardians of the Galaxy	Light years from Earth, 26 years after being a...	0.980259
Mad Max: Fury Road	An apocalyptic story set in the furthest reach...	0.978786
Interstellar	Interstellar chronicles the adventures of a gr...	0.952934
The Avengers	When an unexpected enemy emerges and threatens...	0.943429

- **MLP (Multi-Layer Perceptor):** E' una rete neurale artificiale feedforward composta da almeno tre strati di neuroni: uno strato di input, uno o più strati nascosti e uno strato di output.
  All'interno della classe **MLPRegressorModel** verrà richiamato il metodo **evaluate_model** al cui interno: definiamo un set di iperparametri per la MLP attraverso **param_grid,** una griglia di ricerca **GridSearchCV** che viene utilizzata per trovare la combinazione ottimale di iperparametri tramite la cross-validation, e visualizziamo la curva di perdita per il modello con i migliori parametri:



Inoltre, generiamo e visualizziamo una curva di apprendimento che mostra come le prestazioni del modello variano in base alle dimensioni del set di addestramento.
Addestriamo nuovamente il modello attraverso i migliori iperparametri e effettuiamo previsioni sul set di test (x_test), tali risultati li combiniamo con il dataframe **df_apart**.
Infine, visualizziamo un grafico a barre delle prime 30 previsioni:




```
MAE : 0.03633020530266263
MSE : 0.007672974690582832
RMSE: 0.08759551752562932
```

MAE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore merio del 0,03%;
MSE: è la media delle differenze quadrate tra i valori predetti e quelli osservati, errore quadratico medio dello 0,007%;

RMSE: è la radice quadrata del MSE e serve per interpretare l'errore in termini di scala delle variabili, circa lo 0,08%

OSS: i risultati dell'MLP sembrano essere competitivi rispetto all'SVR, con un MAE inferiore ma con MSE e RMSE leggermente più alti.

Per concludere, come per gli altri modelli mostriamo la predizione dei primi 10 film:

original_title	overview	Predicted
Minions	Minions Stuart, Kevin and Bob are recruited by...	1.062485
Mad Max: Fury Road	An apocalyptic story set in the furthest reach...	1.055808
Avatar	In the 22nd century, a paraplegic Marine is di...	1.012231
The Dark Knight	Batman raises the stakes in his war on crime. ...	1.005320
Jurassic World	Twenty-two years after the events of Jurassic ...	0.998319
The Avengers	When an unexpected enemy emerges and threatens...	0.977471
Interstellar	Interstellar chronicles the adventures of a gr...	0.976187
Inception	Cobb, a skilled thief who commits corporate es...	0.971233
Guardians of the Galaxy	Light years from Earth, 26 years after being a...	0.964002
Deadpool	Deadpool tells the origin story of former Spec...	0.948218

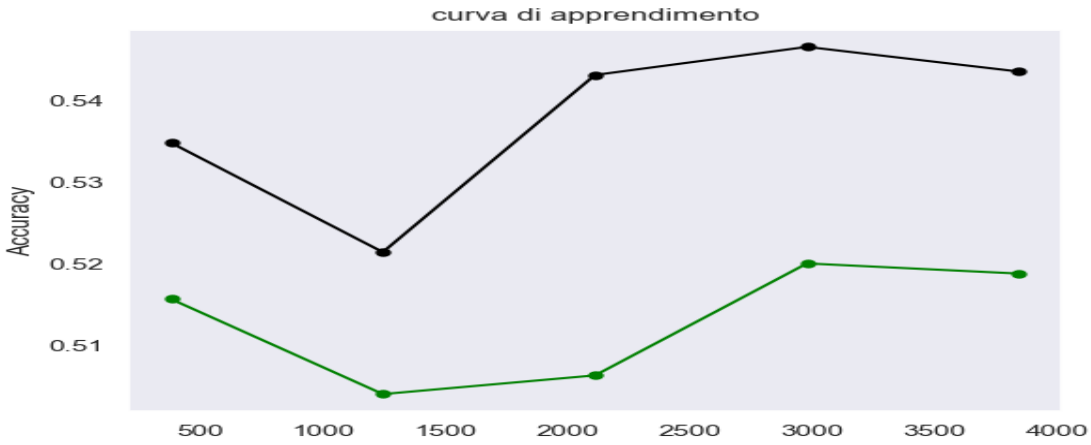
Reasoning on Constraint Optimization Problem

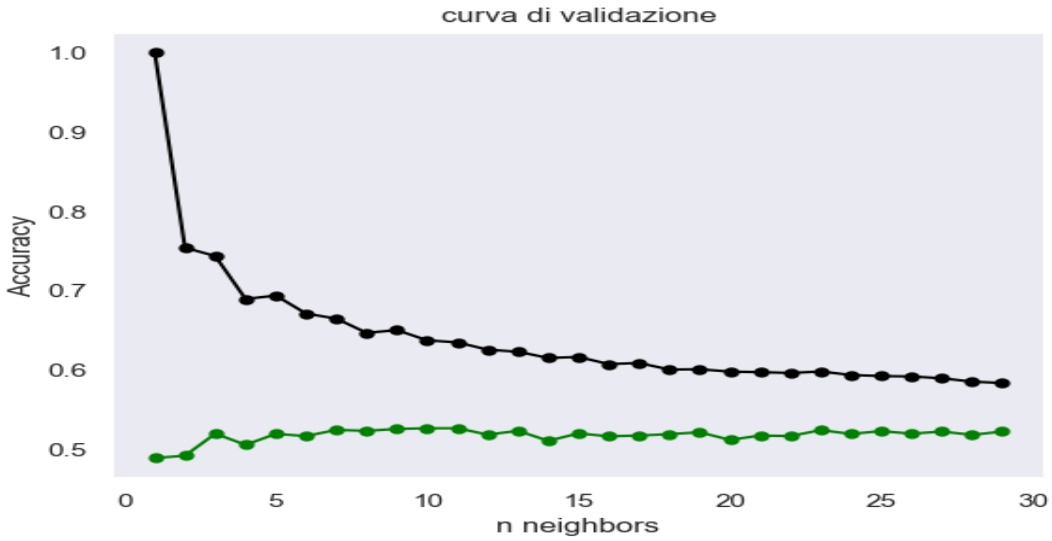
Il "Reasoning on Constraint Optimization Problem" (Ragionamento sui Problemi di Ottimizzazione con Vincoli) è un processo che coinvolge l'utilizzo di ragionamento logico e inferenza per risolvere problemi di ottimizzazione in presenza di vincoli. In questo contesto, i vincoli rappresentano le condizioni o le restrizioni che devono essere soddisfatte mentre si cerca di massimizzare o minimizzare una funzione obiettivo. Il ragionamento sui problemi di ottimizzazione con vincoli si basa su tecniche di inferenza per dedurre soluzioni ottimali che rispettino i vincoli dati.

Questa classe si basa sull'algoritmo di Simulated Anneling: iniziamo con una selezione casuale di film, ad ogni iterazione, genriamo una "mossa" cambiando casulaente un film con un altro che soddisfa i vincoli definiti (geners, vote_averag, popularity).
Calcoliamo il punteggio della nuova selezione e confrontiamo il punteggio ottenuto con quello della selezione attuale: se il punteggio migliora o soddisfa una certa probabilità basata sulla distribuzione di Gibbs , la selezione corrente viene aggiornata con quella migliore.
Il sistema si raffredda gradualmente e continua il processo finchè non raggiunge un valore minimo.

Machine Learning Classification:

All'interno della classe KNN verrà richiamato il metodo **evaluate_model** al cui interno:eseguiamo una ricerca dei migliori iperparametri (n_neighbors) utilizzando la grid search e la cross-validation. Generiamo e visualizziamo la curva di apprendimento del modello, mostrando l'accuratezza del trining e della validation al variare delle dimensioni dataset.
Creo e mostra la curva di validazione per l'iperparametro n_neighbors valutando l'accuratezza del modello al vriare di tale parametro. Esegue il modello migliore trovato dell'ottimizzazionesu dati di test e ne stampa la previsione. Genera una lista di film sulla base della predizione del modello KNN. Se il modello prevede che un film sia gradito(classe 1), il film viene aggiunto a una lista. Viene ,pstrata solo una lista dei primi 30 film predetti come graditi.




curva di validazione
Accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0
5
10
15
20
25
30
n neighbors

Conclusioni

Nello sviluppo del sistema per lo svolgimento delle task designate abbiamo ottenuto dei buoni risultati nello svolgimento della prima task, in quanto è stato possibile attraverso la regole designata individuare i film idonei alle richieste della community e di conseguenza ci ha reso possibile effettuare una regressione dei diversi modelli valutandoli di conseguenza.
Nello sviluppo della seconda task, invece, abbiamo ottenuto dei risultati meno accurati in quanto abbiamo designato una task meno tracciabile con dati, ma dipendenti dalla opinione soggettiva dell'utente, di conseguenza non esiste un modello in grado di valutare in maniera corretta ed accurata questa task.