Home Serie TV Film Nuovi e Popolari La mia lista

NETFLIX PRIZE

Clicca qui per vedere la tua presentazione

+

TABLE OF CONTENTS

01 NETFLIX PRIZE

Home Serie TV Film Nuovi e Popolari La mia lista

OBIETTIVI DELLA COMPETIZIONE

SCOPO PRINCIPALE

Migliorare l'algoritmo di raccomandazione di Netflix, *Cinematch*.

TASK

Incremento del 10% nella precisione delle previsioni delle valutazioni rispetto all'algoritmo esistente.

PREMIO

1 milione di dollari per il team vincitore

STORIA DELLA COMPETIZIONE

LANCIO

Ottobre 2006 avviene il lancio della competizione

PARTECIPAZIONE GLOBALE

Oltre 40.000 team da 186 paesi

VITTORIA

Il Team *BellKor's Pragmatic Chaos* vince con un miglioramento del 10.06%.

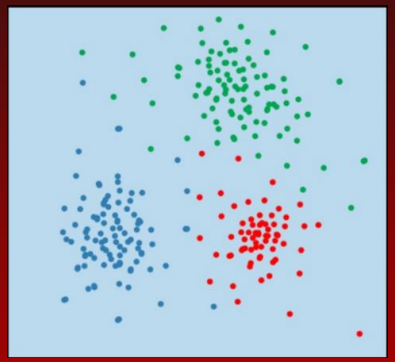
BACKGROUND

Home Serie TV Film Nuovi e Popolari La mia lista

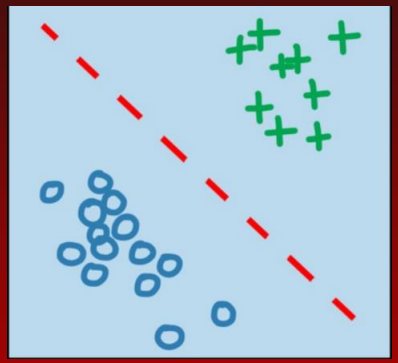
MACHINE LEARNING

Il machine learning è una branca dell'intelligenza artificiale che permette ai computer di apprendere da dati senza essere esplicitamente programmati per ogni compito.

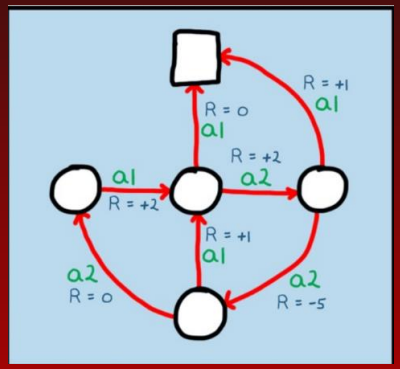
UNSUPERVISED LEARNING



SUPERVISED LEARNING



REINFORCEMENT LEARNING



Home Serie TV Film Nuovi e Popolari La mia lista

UNDERFITTING E OVERFITTING

UNDERFITTING

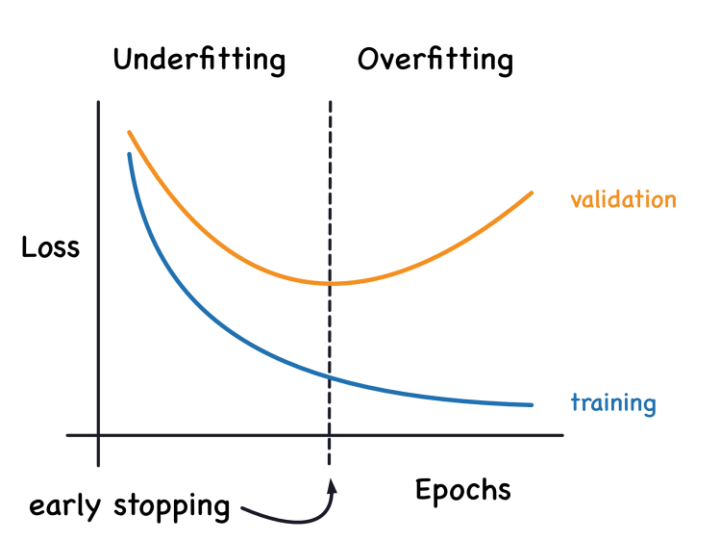
Il modello non è in grado di apprendere sufficientemente dalle informazioni presenti nel dataset, risultando in prestazioni scarse sia sui dati di addestramento (train set) che sui dati di test (test set) e presentando, quindi, un bias elevato.

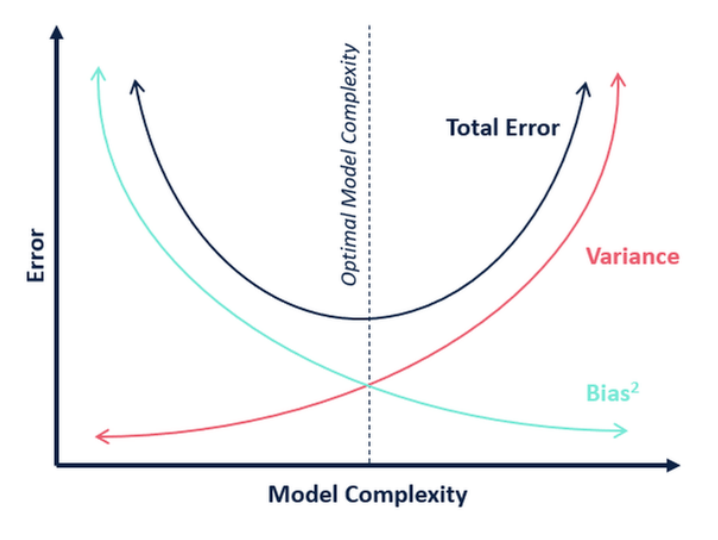
OVERFITTING

Il modello apprende eccessivamente dalle informazioni presenti nel dataset di addestramento (train set), inclusi dettagli irrilevanti e rumore, risultando in prestazioni ottime sui dati di addestramento ma scarse sui dati di test (test set). Questo comporta una capacità ridotta di generalizzare su nuovi dati e una varianza elevata.

Home Serie TV Film Nuovi e Popolari La mia lista

UNDERFITTING E OVERFITTING





Home Serie TV Film Nuovi e Popolari La mia lista

SISTEMI DI RACCOMANDAZIONE

COSA SONO ?

Sistemi che suggeriscono contenuti basati su preferenze e comportamenti degli utenti.

APPLICAZIONI

E-commerce, streaming, social media.

TIPOLOGIE

Collaborative Filtering: Basato su comportamenti simili tra utenti.
Content-Based Filtering: Suggerisce contenuti simili a quelli già apprezzati.

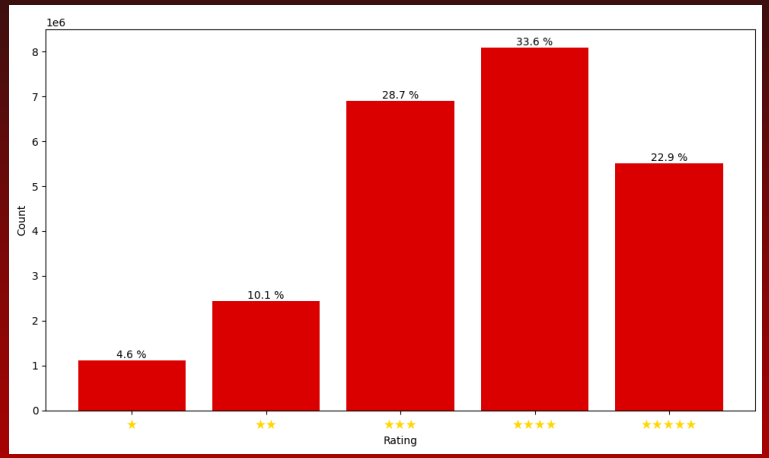
03

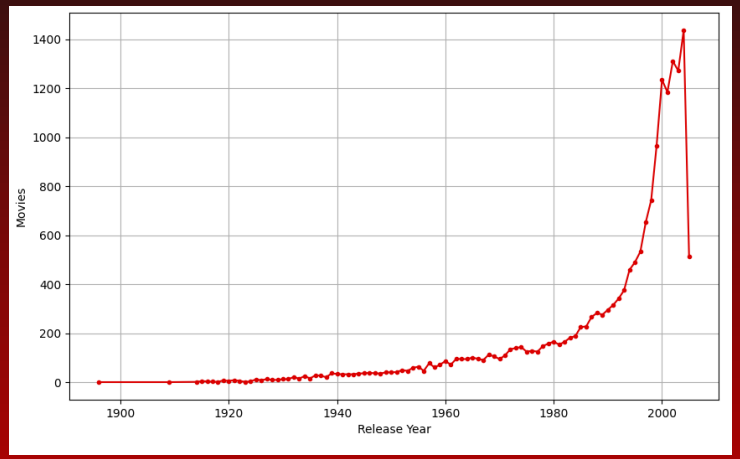
PREPROCESSING

ANALISI DEL DATASET

OBIETTIVO: previsione dei rating che gli utenti darebbero ai film, basandosi sui dati storici delle loro valutazioni (task di regressione).

Il Dataset completo è formato da oltre 100 milioni di valutazioni da parte di circa 480.000 clienti.





COMBINED_DATA_1.TXT

Movie ID 1:

Cust_ID	Rating	Date
1488844	3	2005-09-06
822109	5	2005-05-13
885013	4	2005-10-19
30878	4	2005-12-26
...	...	...

Il file *combined_data_1.txt* è uno dei quattro file che compongono il dataset principale della Netflix Prize Competition. La prima riga contiene l'id del film seguito da due punti. Ogni riga successiva del file corrisponde a una valutazione di un cliente e alla sua data.

MOVIE_TITLES.TXT

Il file *movie_titles.txt* contiene informazioni sui film inclusi nel dataset della Netflix Prize Competition. Ogni riga del file rappresenta un film e include il suo id, l'anno di rilascio e il titolo de film.

ID	Year	Title
1	2003	Dinosaur Planet
2	2004	Isle of Man TT 2004 Review
3	1997	Character
4	1994	Paula Abdul's Get Up & Dance
5	2004	The Rise and Fall of ECW
...	...	...

PREPROCESSING

Il *preprocessing* è la fase iniziale dell'analisi dei dati in cui il dataset viene preparato per l'addestramento del modello. Include operazioni come pulizia dei dati, gestione dei valori mancanti, normalizzazione, codifica delle variabili categoriali e riduzione della dimensionalità, al fine di migliorare l'accuratezza e l'efficienza del modello di machine learning.

Movie ID 1:

Cust_ID	Rating	Date
1488844	3	2005-09-06
822109	5	2005-05-13
885013	4	2005-10-19
30878	4	2005-12-26
...	...	...

Cust_ID	Rating	Movie_ID
1488844	3	1
822109	5	1
885013	4	1
30878	4	2
95786	5	2
...	...	...

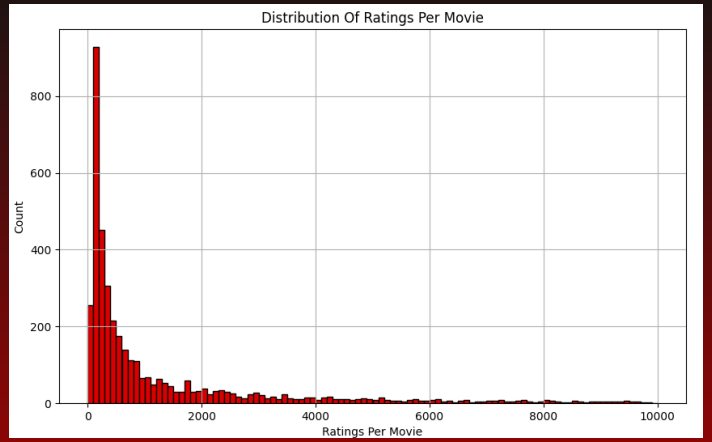
Movie_ID	1	2	3	4	5	6	7	8	9
Cust_ID									
30878	0	4	0	1	0	0	0	4	0
95786	0	4	0	0	0	0	0	0	0
822109	5	0	0	0	0	0	0	0	0
885013	4	0	4	0	0	0	3	0	0
1488844	3	0	2	0	0	0	0	0	0

Numero iniziale di utenti: 470758

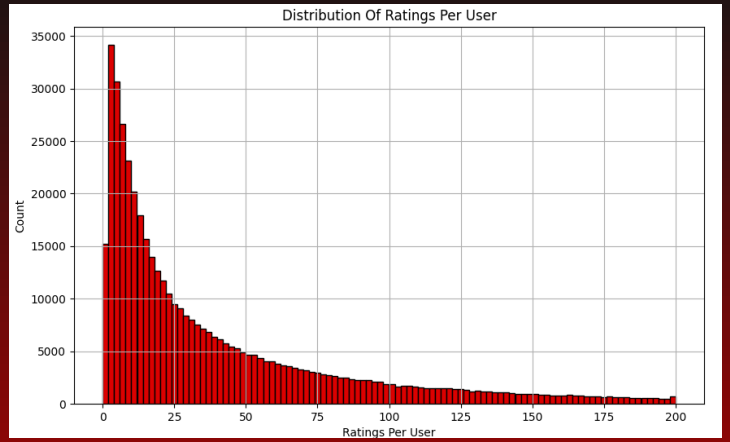
Numero iniziale di film: 4499

PREPROCESSING

Il quantile di ordine p (p –*quantile*) di un insieme ordinato di dati $X = \{x_1, x_2, \dots, x_n\}$ dove $x_1 \leq x_2 \leq \cdots \leq x_n$, è il valore q_p tale che una certa proporzione $p\,(0 \leq p \leq 1)$ dei dati è minore o uguale a q_p e la rimanente proporzione $1 - p$ è maggiore o uguale a q_p.



$p = 0.8$



Numero di rating minimo per film: 3884

Numero finale di film: 900

Numero di rating minimo per utente: 79

Numero finale di utenti: 95325

SVD

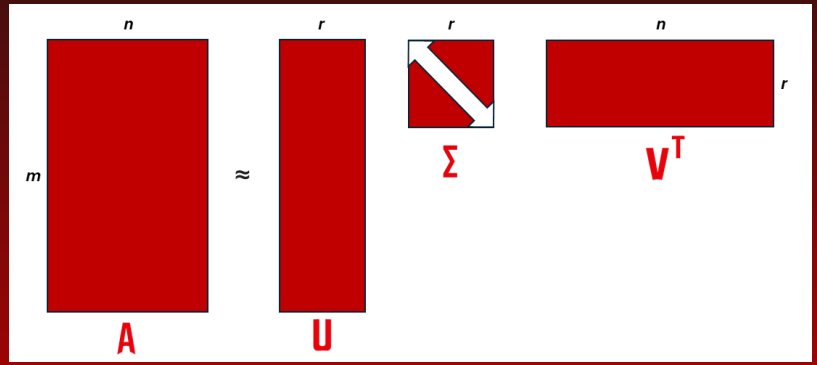
La Singular Value Decomposition (SVD) è una tecnica matematica utilizzata per scomporre una matrice in tre componenti fondamentali:

$$A_{m\times n} = U_{m\times r} \cdot \Sigma_{r\times r} \cdot V^T{}_{r\times n} \qquad \text{con: } r = rank(A) \leq min(m,n)$$

U Matrice ortogonale le cui colonne rappresentano i vettori singolari sinistri, dunque gli autovettori associati alle colonne di A.

Σ Matrice diagonale che contiene i valori singolari di A ordinati in ordine decrescente lungo la diagonale. Essi riflettono l'importanza relativa delle componenti e forniscono informazioni sulla varianza dei dati.

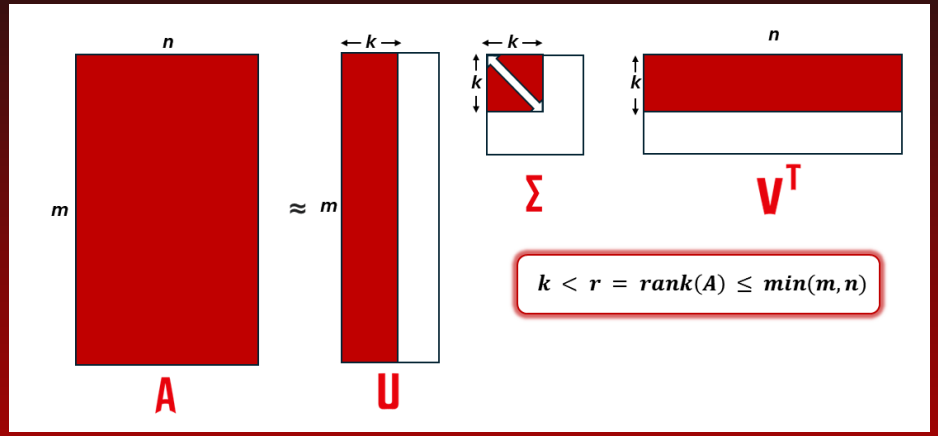
V^T Matrice ortogonale le cui righe rappresentano i vettori singolari destri, dunque gli autovettori associati alle righe di A.



TRUNCATED SVD

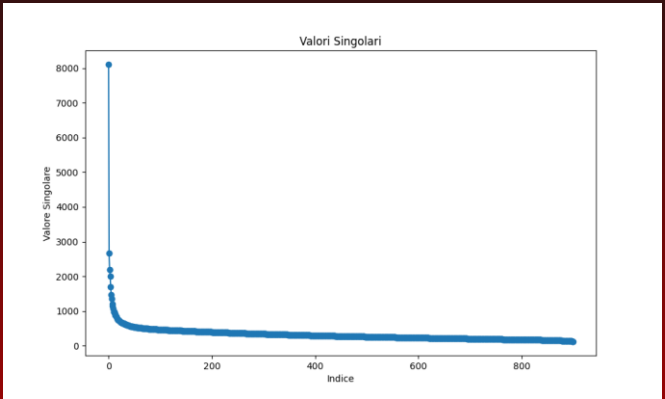
La Truncated SVD (Singular Value Decomposition Troncata) è una versione ridotta della decomposizione ai valori singolari (SVD) in cui si mantengono solo i primi k valori singolari più grandi, eliminando i restanti. Questa tecnica riduce la dimensionalità della matrice originale, conservando le componenti più rilevanti e scartando quelle meno significative.

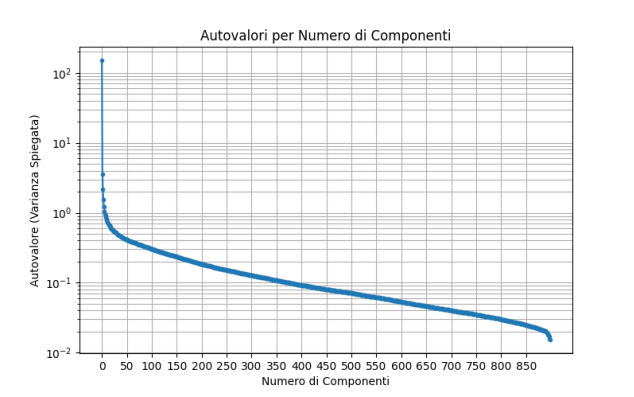
$$A_{m\times n} \approx U_{m\times k} \cdot \Sigma_{k\times k} \cdot V^T{}_{k\times n}$$



ANALISI DEGLI AUTOVALORI

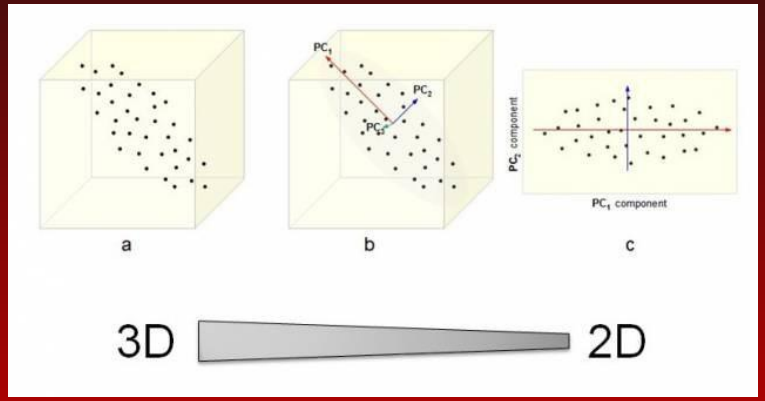
Il grafico mostra i valori singolari ottenuti dalla decomposizione SVD di una matrice. Osservando il grafico, si può notare che i valori singolari iniziali sono generalmente molto più alti rispetto a quelli successivi. Questo indica che le prime k componenti catturano la maggior parte dell'informazione contenuta nella matrice originale.





PCA

La Principal Component Analysis (PCA) è una tecnica di riduzione della dimensionalità utilizzata per trasformare un dataset con molte variabili correlate in un insieme di variabili non correlate, chiamate componenti principali. Le componenti principali catturano la maggior parte della varianza presente nel dataset originale, ordinandole in base alla loro importanza.
La PCA riduce la complessità dei dati mantenendo solo le informazioni più rilevanti, facilitando l'elaborazione dei dati, senza perdere troppe informazioni critiche.



PCA

- **Normalizzazione**: 'centrare' colonne della matrice X in modo che la media di ogni colonna sia 0

$$X = X - mean(X)$$

- **Matrice di Covarianza**: calcolare matrice di covarianza

$$Var(X) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} \qquad Cov(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n} \qquad S = XX^T$$

- **Decomposizione tramite SVD**:

$$X_{m\times n} = U_{m\times r} \cdot \Sigma_{r\times r} \cdot V^T_{r\times n} \qquad T_{m\times r} = U\Sigma \longrightarrow T = XV \qquad T^TT = \Sigma$$

dove T è la *score matrix* le cui colonne rappresentano le PC di X e le righe sono combinazioni lineari delle righe di X che permettono la migliore rappresentazione della varianza di X

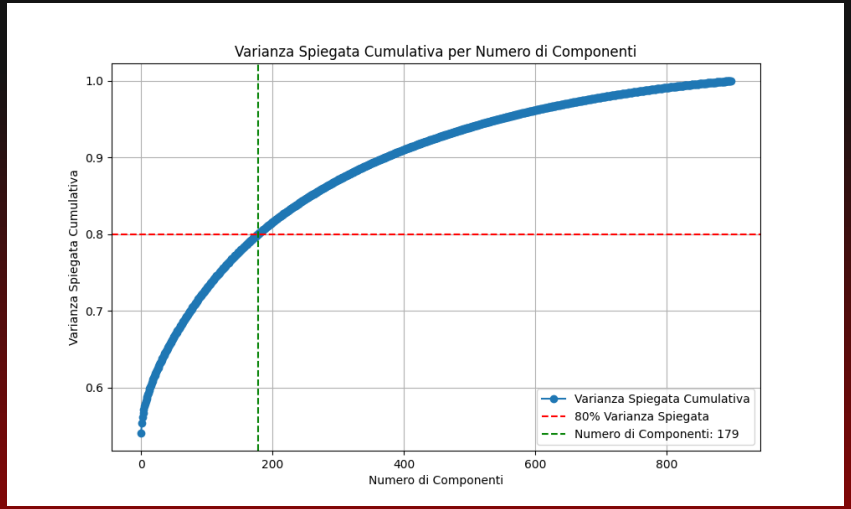
- **Riduzione della dimensionalità**:

$$X = TV^T$$
$$X^{(k)} = T^{(k)} \cdot \left(V^{(k)}\right)^T$$
$$con \quad T^{(k)} = T(1{:}m, 1{:}k) \;\; e \;\; V^{(k)} = V(1{:}n, 1{:}k) \quad dove \; k < r$$

VARIANZA SPIEGATA

La varianza spiegata è il rapporto tra la varianza cumulativa delle prime k componenti principali e la varianza totale di tutte le componenti principali. Essa misura quanto della variabilità complessiva dei dati è rappresentata dalle prime k componenti selezionate rispetto alla varianza totale disponibile.

$$EV = \frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{L} \lambda_i} \cdot 100$$



NMF

La **Non-negative Matrix Factorization (NMF)** è una tecnica di fattorizzazione di matrici che scompone una matrice V, dove ogni elemento è ≥0, in due matrici a valori non negativi, W e H, tali che il prodotto di W e H approssimi V. Questa fattorizzazione è utile per ridurre la dimensionalità e scoprire strutture nascoste o latenti nei dati, mantenendo la non-negatività, il che rende i risultati più interpretabili in molti contesti applicativi.

$$V_{m\times n} \approx W_{m\times k} \cdot H_{k\times n}$$

k è il numero di fattori latenti o componenti, ed è un parametro che deve essere scelto.

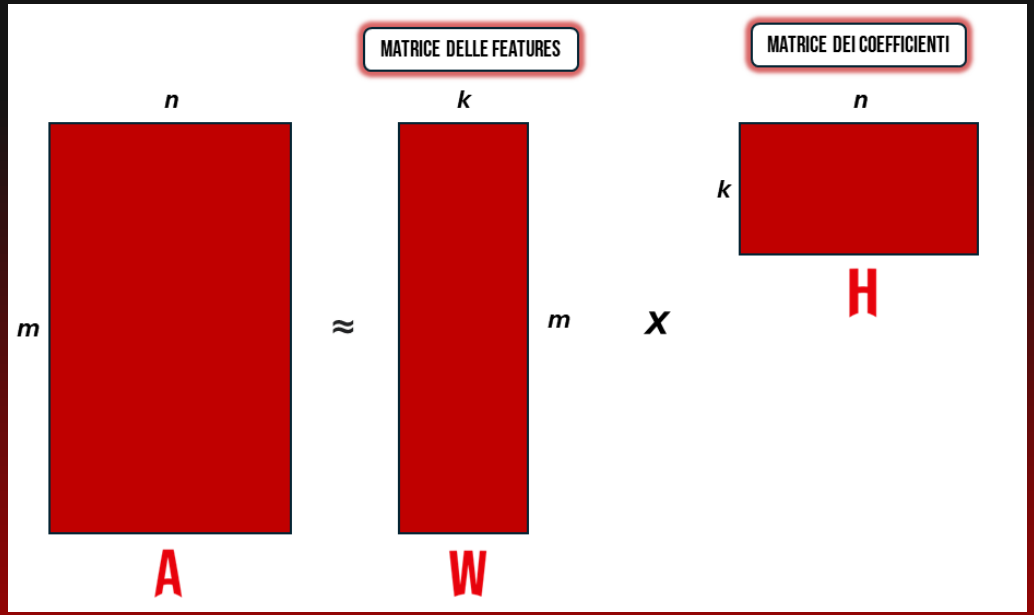
W matrice delle componenti latenti o dei fattori base

H matrice dei coefficienti

NMF

L'obiettivo è minimizzare una funzione di errore, che misura la differenza tra V e W×H. L'errore più comunemente utilizzato è la norma di Frobenius:

$$\min_{W,H} \|V - WH\|_F^2$$



04

ALGORITMI

APPROCCI UTILIZZATI

4.1 SURPRISE SVD

4.2 SURPRISE NMF

4.3 RETE NEURALE

4.4 AUTOENCODER

Home Serie TV Film Nuovi e Popolari La mia lista

LIBRERIA SURPRISE

Surprise è una libreria Python progettata per semplificare la costruzione e la valutazione di sistemi di raccomandazione. La libreria supporta una varietà di algoritmi di raccomandazione, tra cui il filtraggio collaborativo, che si basa su dati di interazioni tra utenti e articoli (nel caso della Netflix Prize Competition si parla di film).

SURPRISE SVD

Nella Singular Value Decomposition (SVD) classica, una matrice A è scomposta come:

$$A = U \cdot \Sigma \cdot V^T$$

Nel contesto delle raccomandazioni, la matrice dei rating A è tipicamente molto sparsa, rendendo impossibile applicare la classica SVD. L'idea è quella di approssimare A calcolando due matrici più piccole, che possiamo chiamare P e Q, basandosi sugli elementi non nulli della matrice sparsa e scegliendo un fattore k per la riduzione della dimensionalità.

$$A'_{m \times n} \approx P_{m \times k} \cdot Q^T_{k \times n}$$

- A': approssimazione della matrice dei rating
- k: numero fattori
- P: matrice in cui ogni riga rappresenta un utente nello spazio dei fattori latenti.
- Q: matrice in cui ogni riga rappresenta un film nello spazio dei fattori latenti.

SURPRISE SVD

Il modello prevede che il rating r_{ui} dato dall'utente u al film i venga stimato come:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

L'addestramento dell'SVD in Surprise avviene tramite un approccio di ottimizzazione basato sul gradiente stocastico. L'obiettivo è minimizzare l'errore tra i rating reali r_{ui} e quelli stimati $\hat{r}_{ui}$, che si traduce nel minimizzare il Regularized Squared Error (RSE):

- μ: media globale dei rating.
- b_u: bias dell'utente u. Rappresenta quanto l'utente tende a dare rating superiori o inferiori alla media.
- b_i: bias del film i. Rappresenta quanto l'item tende a ricevere rating superiori o inferiori alla media.
- p_u: vettore di fattori latenti per l'utente u
- q_i: vettore di fattori latenti per il film i.
- $q_i^T p_u$: prodotto scalare tra i fattori latenti dell'utente e del film. Rappresenta l'interazione latente tra l'utente e il film.
- λ: parametro di regolarizzazione ottenuto tramite cross-validation.

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

SSE **Regularization Term**

SURPRISE SVD

Il processo di calcolo e aggiornamento dei parametri viene ripetuto iterativamente per un numero prefissato di passi (epoche) o fino a quando la funzione di perdita converge a un valore soddisfacente. Durante le iterazioni, il modello migliora gradualmente la qualità delle predizioni dei rating.

$$
\begin{aligned}
b_u &\longleftarrow b_u + \gamma(e_{ui} - \lambda b_u) \\
b_i &\longleftarrow b_i + \gamma(e_{ui} - \lambda b_i) \\
p_u &\longleftarrow p_u + \gamma(e_{ui} \cdot q_i - \lambda p_u) \\
q_i &\longleftarrow q_i + \gamma(e_{ui} \cdot p_u - \lambda q_i)
\end{aligned}
\qquad e_{ui} = r_{ui} - \hat{r}_{ui}
$$

dove γ è il learning rate

SURPRISE NMF

Nella Non-Negative Matrix Factorization, una matrice R è scomposta come:

$$R \approx PQ \qquad con\ P, Q \geq 0$$

Nel contesto delle raccomandazioni, R è una matrice che contiene valutazioni esplicite di alcuni utenti per alcuni film. Questa matrice è in gran parte sparsa perché non tutti gli utenti hanno valutato tutti gli oggetti. L'algoritmo NMF parte da due matrici casuali, P e Q, inizializzate con valori non negativi. La dimensione di queste matrici dipende dal numero di utenti, film, e dal numero di fattori latenti k, che è un parametro scelto dall'utente.

- **R** : approssimazione della matrice dei rating
- **P**: matrice delle componenti latenti o dei fattori base
- **Q**: matrice dei coefficienti

SURPRISE NMF

NMF utilizza metodi iterativi per migliorare progressivamente l'approssimazione $R \approx PQ$. I rating previsti vengono calcolati attraverso la seguente equazione:

$$\hat{r}_{ui} = p_u q_i$$

L'addestramento dell'NMF in Surprise avviene tramite un approccio di ottimizzazione basato sul gradiente stocastico. L'obiettivo è minimizzare l'errore tra i rating reali r_{ui} e quelli stimati $\hat{r}_{ui}$, che si traduce nel minimizzare il Regularized Squared Error (RSE):

- p_u: vettore di fattori latenti per l'utente u
- q_i: vettore di fattori latenti per il film i.
- λ_i : parametro di regolarizzazione per i film ottenuto tramite cross-validation.
- λ_u: parametro di regolarizzazione per gli utenti ottenuto tramite cross-validation.

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda_i \|q_i\|^2 + \lambda_u \|p_u\|^2$$

SSE **Regularization Term**

SURPRISE NMF

Il processo di calcolo e aggiornamento dei parametri viene ripetuto iterativamente per un numero prefissato di passi (epoche) o fino a quando la funzione di perdita converge a un valore soddisfacente. Durante le iterazioni, il modello migliora gradualmente la qualità delle predizioni dei rating.

$$p_{uk} \leftarrow p_{uk} \cdot \frac{\sum_{i \in I_u} q_{ik} \cdot r_{ui}}{\sum_{i \in I_u} q_{ik} \cdot \hat{r}_{ui} + \lambda_u |I_u| p_{uk}}$$

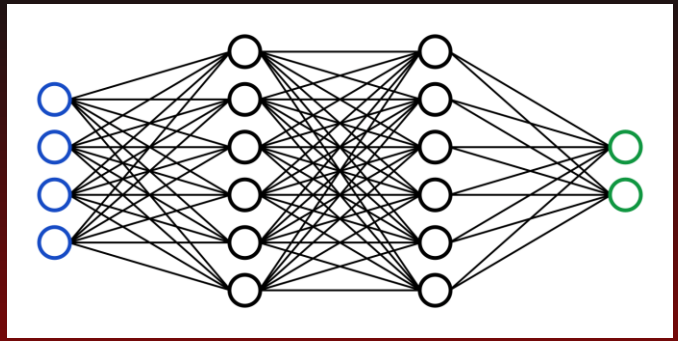
$$q_{ik} \leftarrow q_{ik} \cdot \frac{\sum_{u \in U_i} p_{uk} \cdot r_{ui}}{\sum_{u \in U_i} p_{uk} \cdot \hat{r}_{ui} + \lambda_i |U_i| q_{ik}}$$

- $|U_i|$: cardinalità dell'insieme degli utenti che hanno recensito l'i-esimo film.
- $|I_u|$: cardinalità dell'insieme dei film che sono stati recensiti dall'u-esimo utente.
- k: numero di fattori latenti

RETE NEURALE

Una rete neurale è un modello ispirato al cervello umano, progettato per riconoscere pattern e fare previsioni. È composta da una serie di layer, ognuno formato da neuroni collegati tra loro.

I dati vengono inseriti nell'input layer, dove ogni neurone rappresenta una caratteristica specifica. L'informazione passa poi agli hidden layer, dove i neuroni ricevono input dai nodi precedenti, li moltiplicano per pesi assegnati, sommano i risultati e aggiungono un bias per migliorare la flessibilità del modello.



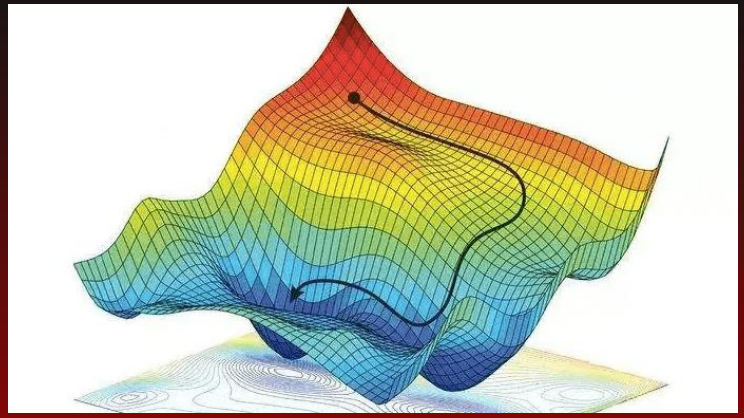
Una funzione di attivazione viene applicata a questa somma per introdurre non-linearità, permettendo alla rete di riconoscere pattern complessi. Infine, l'informazione raggiunge l'output layer, dove viene generata la previsione finale.

Durante l'addestramento, la rete confronta la previsione con il valore reale usando una loss function e, attraverso il processo di backpropagation, modifica i pesi per migliorare le prestazioni. L'ottimizzazione, tramite metodi come il gradient descent, aiuta a ridurre l'errore e affinare le previsioni della rete.

GRADIENT DESCENT

Il Gradient Descent è un algoritmo di ottimizzazione fondamentale utilizzato per minimizzare una funzione di costo. Il suo scopo principale è trovare i valori ottimali dei parametri del modello, come i pesi e i bias in una rete neurale, che riducano al minimo l'errore, ovvero la differenza tra le previsioni del modello e i valori reali dei dati.

L'algoritmo procede iterativamente, aggiornando i parametri nella direzione opposta al gradiente della funzione di costo, che indica la pendenza locale. In questo modo, il Gradient Descent guida il modello verso una configurazione di parametri che produce le previsioni più accurate possibili, minimizzando l'errore complessivo.



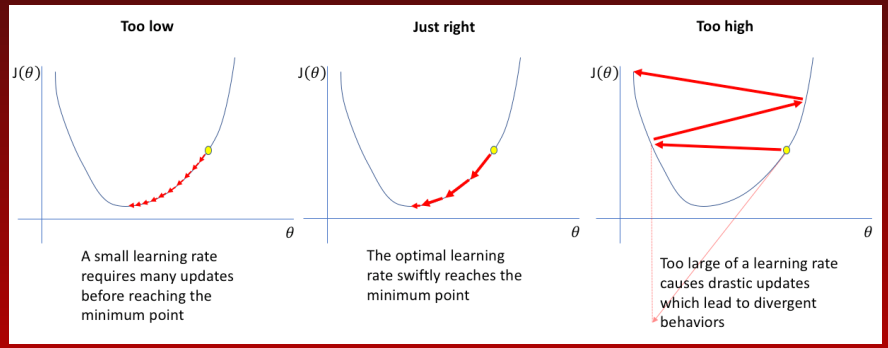
GRADIENT DESCENT

1. **Inizializzazione: Si parte con valori casuali per i parametri (pesi) del modello.**
2. **Calcolo del gradiente: Si calcola il gradiente della funzione di costo rispetto ai parametri, che indica la direzione e la velocità del cambiamento della funzione di costo.**
3. **Aggiornamento dei pesi: I pesi vengono aggiornati nella direzione opposta al gradiente. Questo perché il gradiente indica la direzione di massima crescita, ma noi vogliamo ridurre la funzione di costo. L'aggiornamento avviene secondo la formula:**

$$\theta_{k+1} = \theta_k - \alpha \cdot \nabla J(\theta_k)$$

4. **Ripetizione: Si ripete il processo di calcolo del gradiente e aggiornamento dei pesi finché il modello non raggiunge un minimo della funzione di costo (o si soddisfa una condizione di arresto, come il numero di iterazioni o una convergenza).**

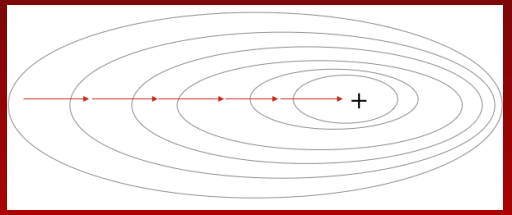
- θ_k **: parametri (o pesi) al passo k.**
- α**: learning rate, che controlla la dimensione del passo che facciamo lungo il gradiente. Se è troppo grande, l'algoritmo può oscillare e non convergere mai, saltando sopra il minimo. Se è troppo piccolo, l'algoritmo impiegherà molto tempo a convergere, poiché i passi sono troppo piccoli.**
- $\nabla J(\theta_k)$**: gradiente della funzione di costo** $J(\theta)$ **rispetto ai parametri** θ**.**



BATCH GD

Calcola il gradiente della funzione di costo utilizzando tutti i dati del dataset in ogni iterazione. Dopo aver calcolato il gradiente sull'intero dataset, i pesi vengono aggiornati.

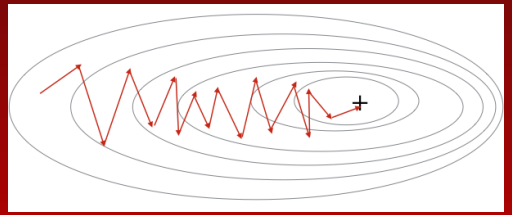
- Aggiornamento dei pesi molto accurato basandosi su tutti i punti del dataset
- Convergenza stabile
- Computazionalmente pesante



STOCHASTIC GD

Aggiorna i pesi dopo aver calcolato il gradiente su un singolo sample del dataset per ogni iterazione. L'aggiornamento dei pesi avviene dopo ogni sample.

- Più veloce perché aggiorna i pesi molto più frequentemente
- Convergenza lenta e meno precisa
- Computazionalmente poco pesante

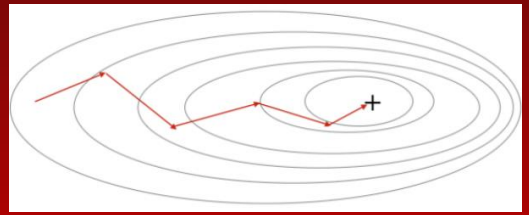


MINI-BATCH GD

Divide il dataset in piccoli lotti (batch) di dimensione b definita (1<b<<m). Il gradiente viene calcolato su ogni mini-batch e i pesi vengono aggiornati dopo ogni mini-batch.

Combina i vantaggi e gli svantaggi di SGD e BGD:

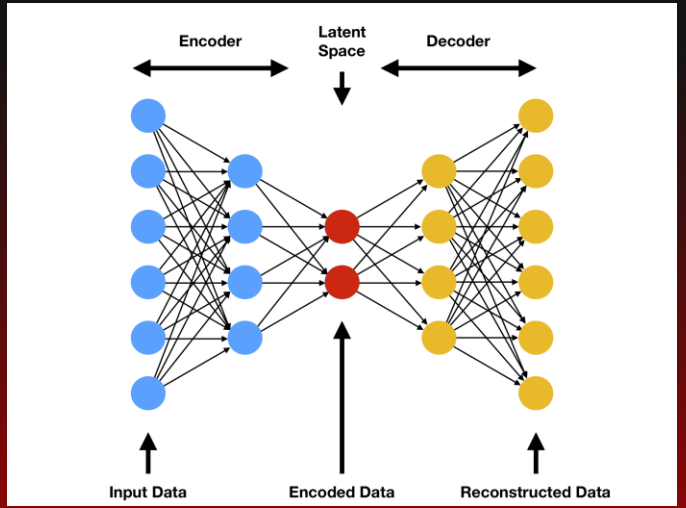
- Più veloce di BGD e più stabile di SGD
- Convergenza più veloce di BGD ma meno di SGD



AUTOENCODER

Un autoencoder è un tipo di rete neurale utilizzato principalmente per l'apprendimento non supervisionato e la riduzione dimensionale. È progettato per apprendere una rappresentazione più compatta (compressa) dei dati in ingresso. Un autoencoder è costituito da due parti principali:

- **Encoder: la prima metà della rete che riduce progressivamente la dimensione dei dati di input, comprimendoli in una rappresentazione più semplice chiamata codifica o latent space.**
- **Latent space: rappresentazione compatta dei dati. In questo spazio, i dati contengono una versione ridotta e spesso più significativa rispetto alla loro forma originale.**
- **Decoder: la seconda metà della rete che ricostruisce i dati originali a partire dalla codifica ridotta.**



VALUTAZIONI

05

SVD SURPRISE

INPUT: Dataframe (ridotto ad 1.000.000 di entries)

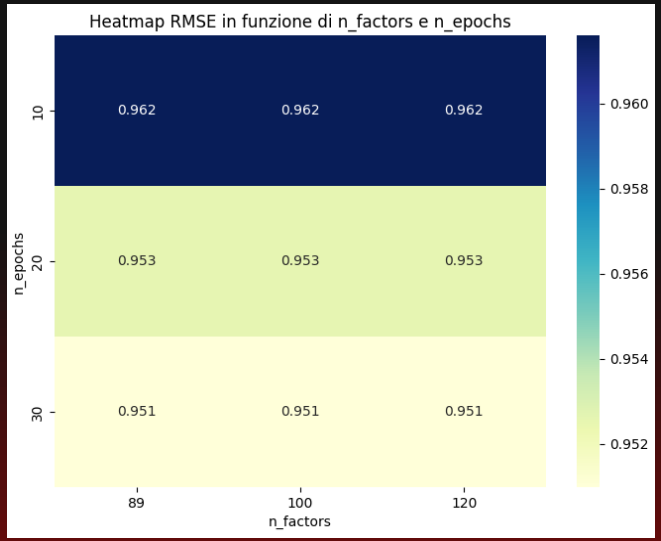
Grid Search

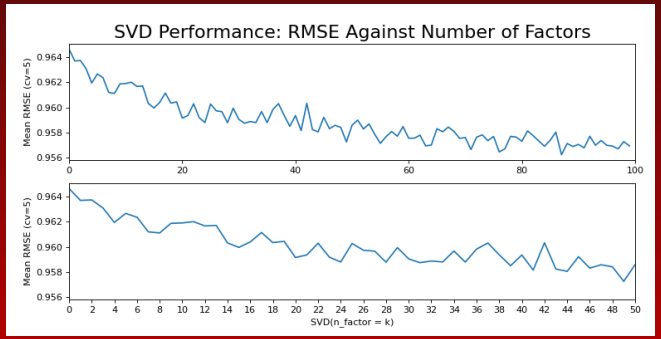
```
'n_factors': [89,100,120],
'n_epochs': [10,20,30],
'lr_all': [0.003,0.005, 0.007],
'reg_all': [0.2, 0.1]
```

Cross Validation

```
n_folds = 5
measures = 'rmse'
```

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}$$





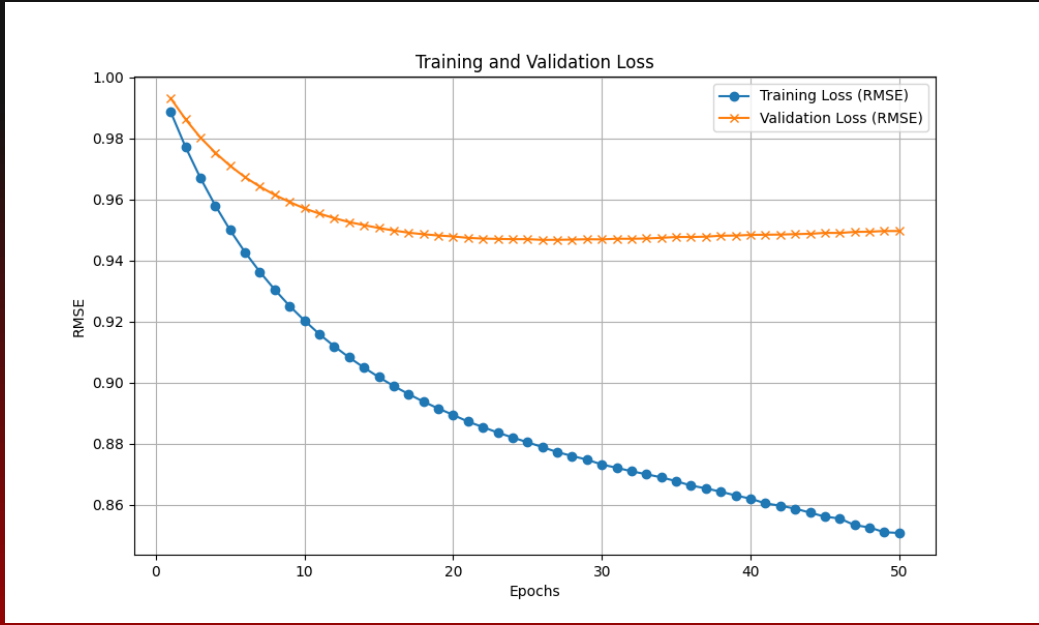
SVD SURPRISE

Migliori Parametri

```
'n_factors': 89,
'n_epochs': 30,
'lr_all': 0.005,
'reg_all': 0.1
```

Loss Values

```
'loss': 0.8764,
'val_loss': 0.9451
```



NMF SURPRISE

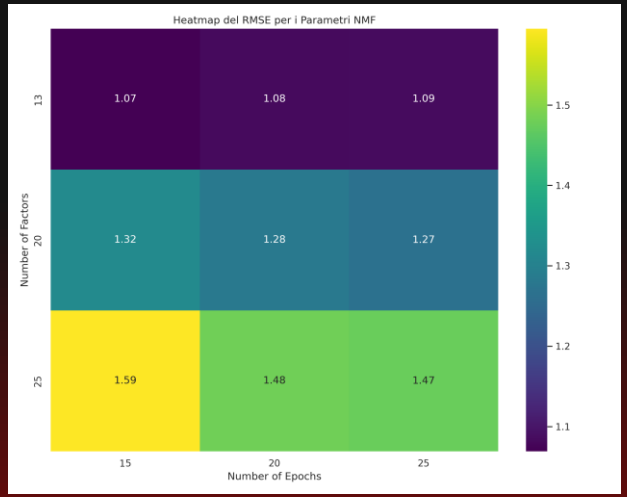
INPUT: Dataframe (ridotto ad 1.000.000 di entries)

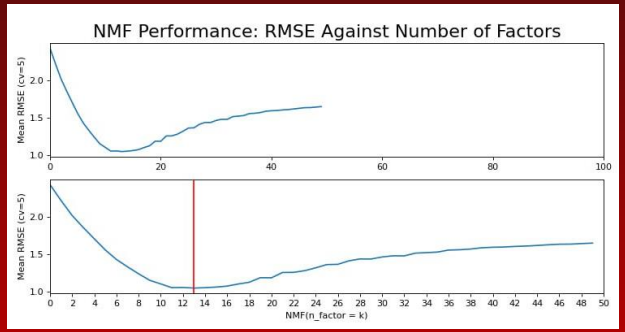
Grid Search

```
'n_factors': [13,20,25],
'n_epochs': [15,20,25],
'reg_qi': [0.01,0.05],
'reg_pu': [0.01,0.05]
```

Cross Validation

```
n_folds = 5
measures = 'rmse'
```



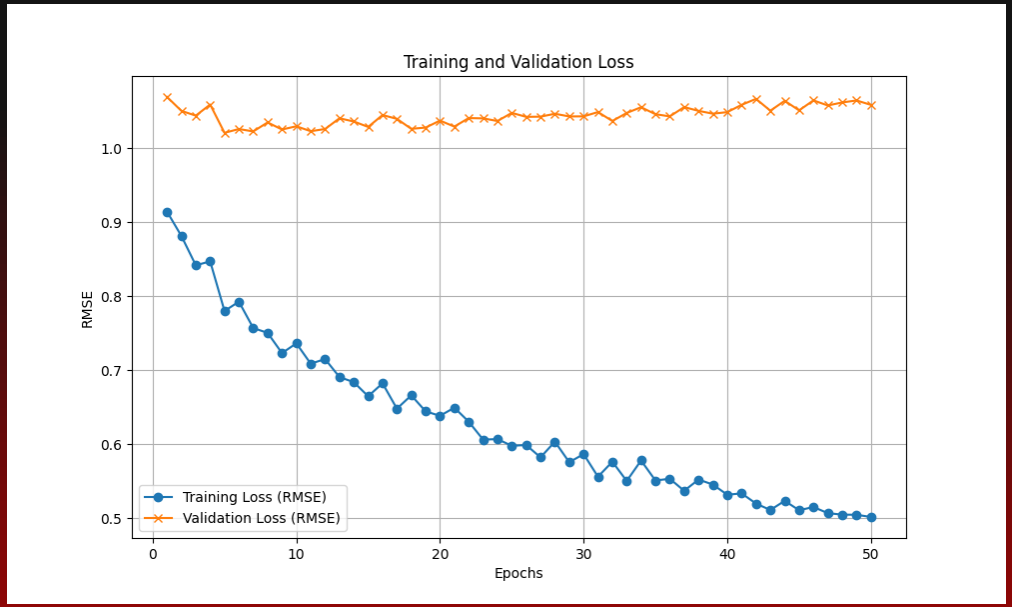


NMF SURPRISE

Migliori Parametri

```
'n_factors’: 13,
'n_epochs’: 15,
‘reg_qi': 0.05,
'reg_pu': 0.05
```

```
'loss’: 0.6789,
'val_loss’:1.07
```



Home Serie TV Film Nuovi e Popolari La mia lista

RETE NEURALE

TASK 1: Predizione di tutti i rating mancanti

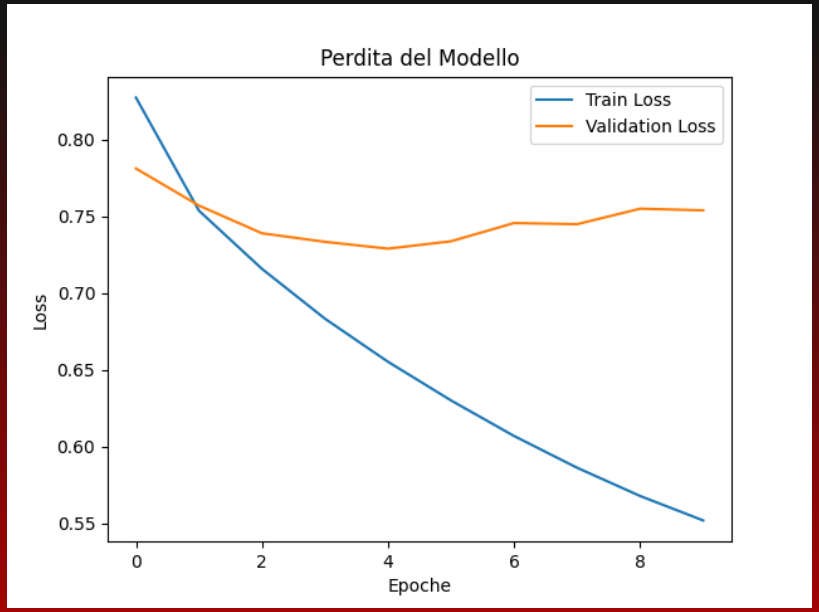
INPUT: Dataframe (ridotto ad 1.000.000 entries)

LAYERS:

- Input per gli utenti
- Input per i film
- Embedding degli utenti
- Embedding dei film
- Flatten degli utenti
- Flatten dei film
- Concatenate
- Dense(128 neuroni, activation='relu')
- Dense(1 neurone)

OPTIMIZER: SGD
BATCH SIZE: 32
CALLBACK: Early Stopping
COST FUNCTION: MSE

TASK 1



RETE NEURALE

TASK 2: Predizione dei rating del film con più recensioni(movie_id=1905)

INPUT: Pivot Table con PCA applicata e NaN sostituiti con row_avg

LAYERS:

- Input
- Dense(128 neuroni, activation='relu', regularizers.l2(0.01))
- Dropout(0.4)
- Dense(64 neuroni, activation='relu')
- Dense(1 neurone, activation='linear')

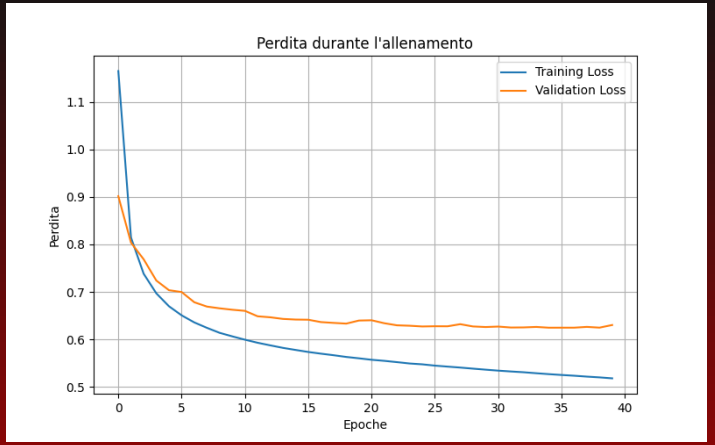
OPTIMIZER: Adam

BATCH SIZE: 64

CALLBACK: Early Stopping (patience=10)

COST FUNCTION: MSE

TASK 2



AUTOENCODER

INPUT: Pivot Table con NaN sostituiti con row_avg

LAYERS:

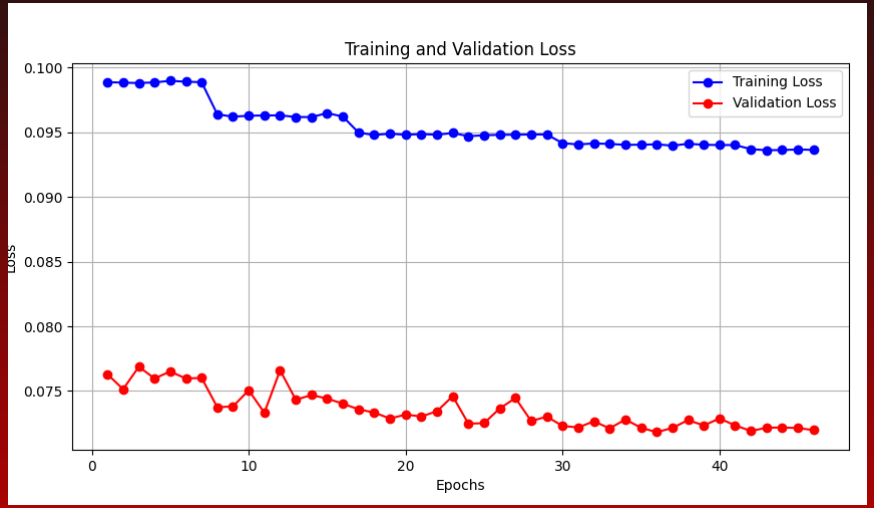
- Input
- Dense(512 neuroni, activation='relu', regularizer=l2(0.001))
- Dropout(0.2)
- Dense(256, activation='relu')
- Dropout(0.2)
- Dense(encoding_dim = 64, activation='relu')
- Dense(256 neuroni, activation='relu')
- Dense(512 neuroni, activation='relu')
- Dense(num_movies, activation='linear')

OPTIMIZER: Adam

BATCH SIZE: 32

CALLBACK: Early Stopping (patience=10)

COST FUNCTION: MSE

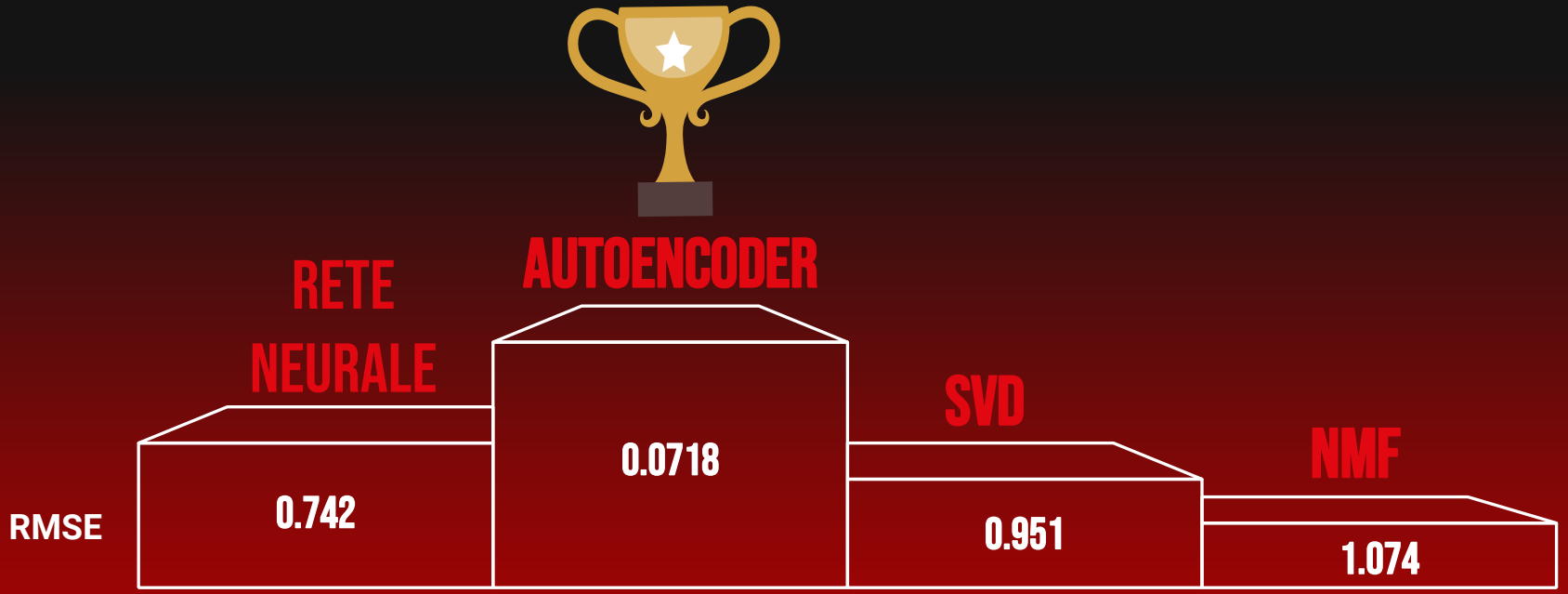


06

CONCLUSIONI

Home Serie TV Film Nuovi e Popolari La mia lista

RISULTATI



BIBLIOGRAFIA

- Ren, YiBo, and SongJie Gong. "A collaborative filtering recommendation algorithm based on SVD smoothing." 2009 Third International Symposium on Intelligent Information Technology Application. Vol. 2. IEEE, 2009.
- Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.
- Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. Recommender Systems Handbook. 1st edition, 2010.
- Luo, Xin, et al. "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems." IEEE Transactions on Industrial Informatics 10.2 (2014): 1273-1284.

GRAZIE PER L'ATTENZIONE