

PROGETTO – CORSO SISTEMI INTELLIGENTI PER INTERNET 2021/2022

UNIVERSITÀ ROMA TRE

Sperimentazione nella realizzazione di uno user recommendation system per la raccomandazione di film
con approccio collaborative item and user based.

MARIO CUOMO

Sommario

IL PROGETTO	1
IL DATASET	2
NOTEBOOK JUPYTER.....	3
FILM MAGGIORMENTE PIACIUTI	4
FILM CON MAGGIORE ENTROPIA	6
PREDIZIONE USER BASED.....	7
PREDIZIONE ITEM BASED.....	10
L'APPLICAZIONE WEB	13
PREDIZIONE PER UTENZE DEL DATASET.....	14
PREDIZIONE PERSONALIZZATA PER TE.....	16
VALUTAZIONE.....	17

IL PROGETTO

Il progetto è disponibile al seguente indirizzo <https://github.com/mariocuomo/progettoSII>.

È una applicazione web sviluppata in Python con il framework Flask in cui è mostrato come poter raccomandare film a un utente in base ai suoi interessi dimostrati nel tempo. Sono presenti anche diversi notebook jupyter di analisi.

Si utilizza l'approccio *collaborativo - user and item based*: il termine collaborativo indica la situazione in cui si crea una raccomandazione in base ai pareri espressi dai vari utenti del sistema; il termine user and item based indica la ricerca di raccomandazioni identificando gli utenti simili a quello di interesse (user) oppure ricercando i film simili ai film piaciuti (item).

La qualità delle raccomandazioni è sufficiente ma non ottima, veloce ma non efficiente. Lo scopo della applicazione è stato quello di ricreare un piccolo sistema di raccomandazione capendone i principi base di realizzazione e non quello di realizzare un sistema innovativo e competitivo con quelli attuali.

Il progetto è realizzato nell'ambito del corso di Sistemi Intelligenti per Internet dell'università Roma Tre per l'anno accademico 2021/2022.

IL DATASET

MovieLens, il dataset usato, è scaricabile gratuitamente dal sito groupLens al seguente indirizzo <https://grouplens.org/datasets/movielens/>. Tra i vari formati disponibili è stato scelto il ml-latest-small, una versione dalle seguenti caratteristiche:

- 10.000 voti assegnati – compresi da 1 a 5 con step di 0.5
- 600 utenti
- 9.000 film – caratterizzati dal genere di appartenenza

Di seguito qualche esempio di dato nel dataset

RATINGS

userId	movieId	rating	timestamp
1	1	4	964982703
1	3	4	964981247
...

MOVIES

movieId	title	genres
1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
2	Jumanji (1995)	Adventure Children Fantasy
...

NOTEBOOK JUPYTER

Sono state effettuate 2 semplici analisi sul dataset di interesse.

La prima identifica i film maggiormente piaciuti, la seconda identifica quelli che presentano maggiore entropia.

Uno dei principali problemi del collaborative filtering è effettuare delle raccomandazioni a nuovi utenti del sistema di cui non si conoscono gli interessi: proporgli film che sono stati graditi da molti utenti o chiedergli di recensire film con maggiore contenuto informativo potrebbe risolvere tale problema.

FILM MAGGIORMENTE PIACIUTI

Molto semplicemente dal dataset ratings si costruisce una tabella pivot users \times items, una matrice in cui nella cella (i,j) è presente il voto che l'utente i ha assegnato al film j.

```
df = pd.read_csv('ratings.csv')
df.head()
```

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931

```
rating_matrix = df.pivot(index = 'userId', columns = 'movieId',
                           values='rating').fillna(0)
rating_matrix
```

movieId	1	2	3	4	5	6	7	8	9	10	...	193565	193567	193571	193573	193579	193581	193583	193585	193587	193609
userId																					
1	4.0	0.0	4.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...
606	2.5	0.0	0.0	0.0	0.0	0.0	2.5	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
607	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
608	2.5	2.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
609	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
610	5.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

A questo punto è sufficiente creare una lista di coppie (film, #voti_maggiori_di_4) estraendo i primi k elementi con il secondo campo più grande.

```
l=[]
for film in rating_matrix.columns:
    lst = rating_matrix[film].value_counts().items()
    voti=0
    totali=0
    for x in lst:
        if(float(x[0])>=4):
            voti=voti+x[1]
    l.append((film,voti))
l.sort(key = lambda x: x[1], reverse=True)[:20]
```

I dieci film maggiormente piaciuti sono i seguenti

	TITOLO	GENERE
0	Shawshank Redemption, The (1994)	Crime,Drama
1	Forrest Gump (1994)	Comedy,Drama,Romance,War
2	Pulp Fiction (1994)	Comedy,Crime,Drama,Thriller
3	Silence of the Lambs, The (1991)	Crime,Horror,Thriller
4	Matrix, The (1999)	Action,Sci-Fi,Thriller
5	Star Wars: Episode IV - A New Hope (1977)	Action,Adventure,Sci-Fi
6	Fight Club (1999)	Action,Crime,Drama,Thriller
7	Schindler's List (1993)	Drama,War
8	Star Wars: Episode V - The Empire Strikes Back...	Action,Adventure,Sci-Fi
9	Braveheart (1995)	Action,Drama,War
10	Usual Suspects, The (1995)	Crime,Mystery,Thriller

FILM CON MAGGIORE ENTROPIA

Quando non si conoscono i giudizi di un utente una prassi utilizzata è quella di chiedergli di giudicare alcuni film. I film proposti all'utente sono quelli con maggiore entropia, ovvero quelli che hanno avuto dei giudizi contrastanti e quindi – dalla teoria dell'informazione – contengono il maggior contenuto informativo.

Data una sequenza di valori X , l'entropia definita da Shannon si calcola come:

$$H(X) := - \sum_{x \in X} p(x) \log p(x)$$

A partire dataset ratings si costruisce una tabella pivot users \times items – così come avviene nel paragrafo relativo ai film più piaciuti.

Si costruisce infine una lista di coppie coppie (film, valore_entropia) estraendo i primi k elementi con il secondo campo più grande.

```
lst=[]
for (colname,colval) in rating_matrix.iteritems():
    lst.append((colname,scipy.stats.entropy(colval)))
lst.sort(key = lambda x: x[1], reverse=True)[:20]
```

I 10 film con maggiore entropia sono i seguenti

	TITOLO	ENTROPIA	GENERE
0	Forrest Gump (1994)	5.773944	Comedy,Drama,Romance,War
1	Shawshank Redemption, The (1994)	5.744399	Crime,Drama
2	Pulp Fiction (1994)	5.695939	Comedy,Crime,Drama,Thriller
3	Silence of the Lambs, The (1991)	5.606308	Crime,Horror,Thriller
4	Matrix, The (1999)	5.593650	Action,Sci-Fi,Thriller
5	Star Wars: Episode IV - A New Hope (1977)	5.501065	Action,Adventure,Sci-Fi
6	Jurassic Park (1993)	5.444037	Action,Adventure,Sci-Fi,Thriller
7	Braveheart (1995)	5.434879	Action,Drama,War
8	Terminator 2: Judgment Day (1991)	5.379976	Action,Sci-Fi
9	Schindler's List (1993)	5.360749	Drama,War
10	Fight Club (1999)	5.359550	Action,Crime,Drama,Thriller

PREDIZIONE USER BASED

La User-Based Nearest Neighbor Recommendation si basa su 3 passaggi principali

1. Si identificano i k utenti più simili dell'utente target sulla base delle preferenze espresse
2. Per ogni item che l'utente target non ha ancora visionato si stimano i rating – sulla base dei rating espressi da esso e dagli utenti simili
3. Si raccomandano gli item non ancora visionati e con il rate predetto più alto

Per calcolare la similarità tra due utenti si utilizza il Pearson Correlation Coefficient la cui formula, considerando un utente a e un utente b , è

$$\text{sim}(a, b) = \frac{\sum_{p \in P} (r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P} (r_{a,p} - \bar{r}_a)^2} \sqrt{\sum_{p \in P} (r_{b,p} - \bar{r}_b)^2}}$$

dove $r_{i,p}$ è il rating dato dall'utente i all'item p , P è l'insieme degli item valutati da entrambi gli utenti e \bar{r}_i è il rating medio dell'utente i .

Il valore è compreso tra -1 e $+1$ ad indicare una correlazione negativa o una correlazione positiva.

Una volta identificati i k utenti più simili dell'utente target (di seguito indicato come l'insieme N), per ogni film ancora non visto si predice il rating combinando il voto assegnato dagli utenti simili pesato per il valore di similarità.

$$\text{pred}(a, p) = \bar{r}_a + \frac{\sum_{b \in N} \text{sim}(a, b)(r_{b,p} - \bar{r}_b)}{\sum_{b \in N} \text{sim}(a, b)}$$

A partire dataset ratings si costruisce una tabella pivot users \times items – così come avviene nel paragrafo relativo ai film più piaciuti.

Si crea un dizionario con chiave l'id dell'utente e valore la media dei voti assegnati

```
rating_matrix['average'] = rating_matrix.mean(axis=1)
media_utenti = dict()
for index, row in rating_matrix.iterrows():
    media_utenti[index] = row['average']
media_utenti
del rating_matrix['average']
rating_matrix=rating_matrix.fillna(0)
```

Si realizza una tabella pivot users \times user, una matrice in cui nella cella (i, j) è presente il valore di similarità tra l'utente i e l'utente j . È ovviamente una matrice simmetrica con tutti 1 sulla diagonale.

userid	1	2	3	4	5	6	7	8	9	10	...	601	602	603	604	605	606
userid																	
1	1.000000	0.019400	0.053056	0.176920	0.120866	0.104418	0.143793	0.128547	0.055268	-0.000298	...	0.066256	0.149942	0.186978	0.056530	0.134412	0.121981
2	0.019400	1.000000	-0.002594	-0.003804	0.013183	0.016257	0.021567	0.023750	-0.003448	0.061880	...	0.198549	0.010888	-0.004030	-0.005345	-0.007919	0.011299
3	0.053056	-0.002594	1.000000	-0.004556	0.001887	-0.004577	-0.005634	0.001703	-0.003111	-0.005501	...	0.000150	-0.000585	0.011211	-0.004822	0.003678	-0.003246
4	0.176920	-0.003804	-0.004556	1.000000	0.121018	0.065719	0.100602	0.054235	0.002417	0.015615	...	0.072848	0.114287	0.281866	0.039699	0.065493	0.164831
5	0.120866	0.013183	0.001887	0.121018	1.000000	0.294138	0.101725	0.426576	-0.004185	0.023471	...	0.061912	0.414931	0.095394	0.254117	0.141077	0.090158
...
606	0.121981	0.011299	-0.003246	0.164831	0.090158	0.047506	0.172499	0.081913	0.057989	0.054877	...	0.153892	0.084208	0.224637	0.035251	0.106752	1.000000
607	0.254200	0.005813	0.012885	0.115118	0.145764	0.142169	0.173293	0.178133	0.003257	-0.004809	...	0.080034	0.187588	0.173025	0.126267	0.101138	0.115999
608	0.262241	0.032730	0.008096	0.116861	0.122607	0.137954	0.305439	0.175912	0.086229	0.048373	...	0.136316	0.174069	0.164479	0.133734	0.144896	0.188354
609	0.085434	0.024373	-0.002963	0.023930	0.258289	0.207124	0.084494	0.421627	-0.003937	0.014983	...	0.029664	0.331053	0.046000	0.232115	0.089810	0.052385
610	0.098719	0.089329	0.015962	0.062523	0.040372	-0.010400	0.163775	0.058168	0.055265	0.087036	...	0.200487	0.050806	0.064594	0.018502	0.072727	0.093851

Se consideriamo una predizione per l'utente 1, per prima cosa è necessario recuperare la lista dei k utenti più simili.

Di seguito, `rslt_df` è un dataframe di una riga – la riga corrispondente all'utente target nel dataframe di correlazione `users × user`.

```
rslt_df = user_similarity[user_similarity['userId'] == 1]
```

Si selezionano solamente i top k e si inseriscono in una lista di triple (`id_utente`, `similarità`, `lst`), dove `lst` è una lista di coppie (`id_film`, `voto_assegnato`).

```
lst_similarita=[]
for index, value in (rslt_df.iloc[0]).items():
    if index != 'userId':
        lst_similarita.append((index,value, rating_matrix.iloc[index-1]))
lst_similarita.sort(key = lambda x: x[1], reverse=True)

top_k=lst_similarita[1:10]
```

Si selezionano i film non visionati andando a selezionare i film la cui votazione è 0.

```
film_non_visionati = []
for index, value in rating_matrix.iloc[0].items():
    film_non_visionati.append((index,value))

film_non_visionati = list(filter(lambda item: item[1] == 0,
film_non_visionati))
```

Infine si stima il rating come descritto in precedenza.

Per ogni film non visto dall'utente target si combinano i voti dei k utenti simili pesati per il valore di similarità.

```

rating_stimato=[]
for (film,_) in film_non_visionati:
    rate=media_utenti.get(1)
    num=0
    den=0
    for (u,similarita,lista_film_rating) in top_k:
        den=den+similarita
        rate_u_film = lista_film_rating[film] - media_utenti.get(u)
        num=num+(similarita*rate_u_film)
    rating_stimato.append((film,rate+(num/den)))

```

Per l'utente con id 1, i 10 film da suggerire sono i seguenti

	TITOLO	RATING PREDETTO	GENERE
0	Godfather, The (1972)	5.034600	Crime,Drama
1	Aliens (1986)	4.991028	Action,Adventure,Horror,Sci-Fi
2	Terminator 2: Judgment Day (1991)	4.961012	Action,Sci-Fi
3	Sixth Sense, The (1999)	4.809837	Drama,Horror,Mystery
4	Hunt for Red October, The (1990)	4.805610	Action,Adventure,Thriller
5	Godfather: Part II, The (1974)	4.703343	Crime,Drama
6	Die Hard (1988)	4.636191	Action,Crime,Thriller
7	2001: A Space Odyssey (1968)	4.469511	Adventure,Drama,Sci-Fi
8	Blade Runner (1982)	4.414505	Action,Sci-Fi,Thriller
9	Breakfast Club, The (1985)	4.378869	Comedy,Drama
10	Star Trek II: The Wrath of Khan (1982)	4.235676	Action,Adventure,Sci-Fi,Thriller

PREDIZIONE ITEM BASED

La Item-Based Nearest Neighbor Recommendation si basa su 3 passaggi principali

1. Per ogni item non visto dall'utente si identificano i k item più simili di ognuno di essi – in base ai rating assegnati dai vari utenti
2. Per ogni item che l'utente target non ha ancora visionato si stimano i rating – sulla base dei rating espressi da esso e dagli utenti simili
3. Si raccomandano gli item non ancora visionati e con il rate predetto più alto

Per calcolare la similarità tra due item si utilizza la Cosine Similarity la cui formula, considerando un item \vec{a} e un item \vec{b} (vettori n -dimensionali), è

$$\text{sim}(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| \times |\vec{b}|}$$

Il valore è compreso tra 0 e +1 ad indicare una correlazione negativa o una correlazione positiva.

Si può Adjusted Cosine Measure sottraendo la media dell'utente dai rating e ottenere una metrica nell'intervallo $[-1, +1]$.

Per ogni film ancora non visto si predice il rating combinando il voto assegnato dall'utente target pesato per la similarità col film in questione. Di seguito N è l'insieme dei k film più simili a p , visionati dall'utente u .

$$\text{pred}(u, p) = \frac{\sum_{i \in N} \text{sim}(i, p) r_{u,i}}{\sum_{i \in N} \text{sim}(i, p)}$$

A partire dataset ratings si costruisce una tabella pivot users \times items – così come avviene nel paragrafo relativo ai film più piaciuti.

Si realizza una tabella pivot item \times item, una matrice in cui nella cella (i, j) è presente il valore di similarità tra l'item i e l'item j . È ovviamente una matrice simmetrica con tutti 1 sulla diagonale.

movieid	movieid	1	2	3	4	5	6	7	8	9	...	193565	193567	193571	193573	193579	193581
0	1	1.000000	0.353349	0.241703	-0.109716	0.247283	0.337120	0.233089	0.008300	0.141242	...	-0.119254	-0.121518	-0.116790	-0.116790	-0.119254	-0.116790
1	2	0.353349	1.000000	0.231903	0.060885	0.261948	0.250034	0.183685	0.145316	0.035806	...	0.015605	0.015921	0.015262	0.015262	0.015605	0.015262
2	3	0.241703	0.231903	1.000000	0.173316	0.470362	0.246164	0.423017	0.327803	0.338285	...	0.129671	0.132172	0.126954	0.126954	0.129671	0.126954
3	4	-0.109716	0.060885	0.173316	1.000000	0.319023	-0.011095	0.320615	0.653855	0.427895	...	0.774481	0.789323	0.758350	0.758350	0.774481	0.758350
4	5	0.247283	0.261948	0.470362	0.319023	1.000000	0.261599	0.562415	0.406183	0.418025	...	0.261004	0.266014	0.255560	0.255560	0.261004	0.255560
...
9719	193581	-0.116790	0.015262	0.126954	0.758350	0.255560	-0.038217	0.214067	0.630101	0.467547	...	0.998818	0.995079	1.000000	1.000000	0.998818	1.000000
9720	193583	-0.119254	0.015605	0.129671	0.774481	0.261004	-0.039010	0.218630	0.643504	0.477495	...	1.000000	0.998719	0.998818	0.998818	1.000000	0.998818
9721	193585	-0.119254	0.015605	0.129671	0.774481	0.261004	-0.039010	0.218630	0.643504	0.477495	...	1.000000	0.998719	0.998818	0.998818	1.000000	0.998818
9722	193587	-0.119254	0.015605	0.129671	0.774481	0.261004	-0.039010	0.218630	0.643504	0.477495	...	1.000000	0.998719	0.998818	0.998818	1.000000	0.998818
9723	193609	-0.116769	0.015282	0.126974	0.758364	0.255574	-0.038195	0.214082	0.630112	0.467559	...	0.841772	0.857901	0.824242	0.824242	0.841772	0.824242

Se si considera una predizione per l'utente con id 1, si creano due liste di film visionati e non visionati

```
film_rating = []
for index, value in rating_matrix.iloc[user-1].items():
    film_rating.append((index,value))

film_non_visionati = list(filter(lambda item: item[1] == 0, film_rating))
film_rated = list(filter(lambda item: item[1] != 0, film_rating))
```

Per semplicità si crea una lista di triple (film_visionato, rate, lst) – dove lst è una lista di coppie (film, similarita).

```
filmVistiEListaSimili=[]
for (film, rate) in film_rated:
    filmVistiEListaSimili.append((film, rate, dict(list(((dd[dd['movieId'] ==
film])).squeeze()).items()))))
```

Infine si stima il rating come descritto in precedenza.

Per ogni film non visto dall'utente target si combinano i voti assegnati da esso agli altri voti pesati per la similarità col film in questione.

```
rating_stimato=[]
for film,_ in film_non_visionati:
    den=0
    num=0
    for (_film, rate, dictSimili) in filmVistiEListaSimili:
        if(film!=_film):
            sim = dictSimili[_film]
            num=num+(rate*sim)
            den=den+sim
    if(den==0):
        rating_stimato.append((film,0))
    else:
        rating_stimato.append((film,num/den))

#ordina rating
rating_stimato.sort(key = lambda x: x[1], reverse=True)
```


Per l'utente con id 1, i 10 film da suggerire sono i seguenti

	TITOLO	RATING PREDETTO	GENERE
0	Dark Knight, The (2008)	4.487765	Action,Crime,Drama,IMAX
1	Spirited Away (Sen to Chihiro no kamikakushi) ...	4.482222	Adventure,Animation,Fantasy
2	Lord of the Rings: The Two Towers, The (2002)	4.468135	Adventure,Fantasy
3	Shawshank Redemption, The (1994)	4.462798	Crime,Drama
4	Up (2009)	4.462370	Adventure,Animation,Children,Drama
5	Lord of the Rings: The Return of the King, The...	4.461037	Action,Adventure,Drama,Fantasy
6	WALL-E (2008)	4.451850	Adventure,Animation,Children,Romance,Sci-Fi
7	Amelie (Fabuleux destin d'Amélie Poulain, Le) ...	4.449388	Comedy,Romance
8	Lord of the Rings: The Fellowship of the Ring,...	4.445146	Adventure,Fantasy
9	Godfather, The (1972)	4.443358	Crime,Drama
10	Inception (2010)	4.443153	Action,Crime,Drama,Mystery,Sci-Fi,Thriller,IMAX

L'APPLICAZIONE WEB

Per eseguire l'applicazione è sufficiente installare Python (versione 3 e successive) e framework Flask, spostarsi nella cartella progettoSII\sito e lanciare il comando `python run.py`. Sarà avviato un server web in ascolto all'url <http://localhost:5000/>.

BENVENUTO




RECOMMENDER SYSTEM COLLABORATIVE FILTERING

dataset [MovieLens](#) (*ml-latest-small*)

Descrizione dataset
NUMERO DI UTENTI: 600
NUMERO DI FILM: 9.000
NUMERO DI RATING: 100.000

Caso d'uso
Gli utenti sono memorizzati anonimamente nel dataset con un intero: da 1 a 610
Inserisci l'id dell'utente e schiaccia il bottone **ENTRA**.
Una volta entrato nel sistema saranno mostrati tutti i rating dati dall'utente selezionato.
È possibile generare una raccomandazione per l'utente con tre metodi differenti.
È descritto come avviene la predizione e la valutazione del RS.

ID utente:



Vuoi scoprire delle predizioni per te?

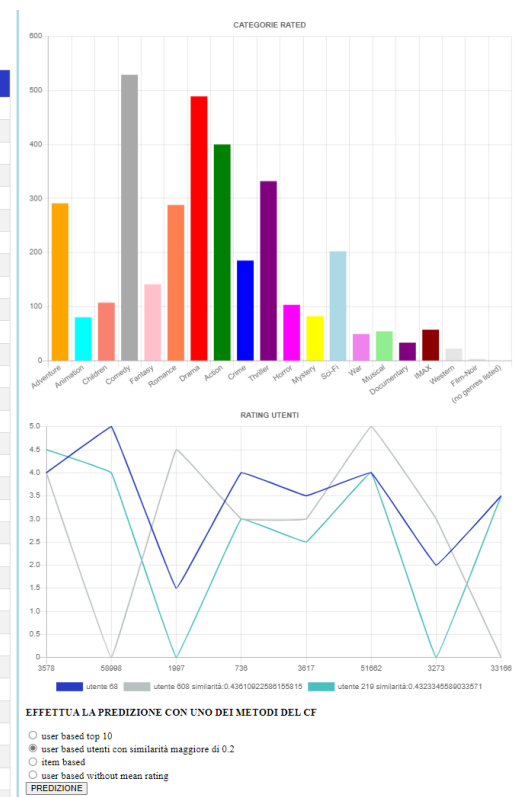
credit: © Mario Cuomo

Nella homepage del sistema si presentano due sezioni principali.

La prima sezione permette di visualizzare le raccomandazioni per un utente del sistema – caratterizzati da un id da 1 a 610 – e l'altra per visualizzare le raccomandazioni realizzate per noi.

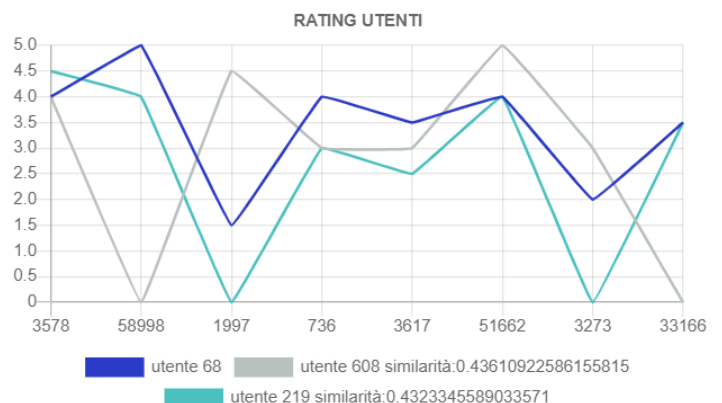
La schermata di presentazione sarà molto simile alla seguente

RATING		
TITOLO	RATING	GENERE
Toy Story (1995)	2.5	Adventure,Animation,Children,Comedy,Fantasy
Jumanji (1995)	2.5	Adventure,Children,Fantasy
Grumpier Old Men (1995)	2.0	Comedy,Romance
Father of the Bride Part II (1995)	2.0	Comedy
Heat (1995)	4.0	Action,Crime,Thriller
Sabrina (1995)	2.0	Comedy,Romance
GoldenEye (1995)	4.5	Action,Adventure,Thriller
American President, The (1995)	4.5	Comedy,Drama,Romance
Casino (1995)	3.5	Crime,Drama
Sense and Sensibility (1995)	3.5	Drama,Romance
Four Rooms (1995)	2.0	Comedy
Ace Ventura: When Nature Calls (1995)	1.5	Comedy
Leaving Las Vegas (1995)	3.5	Drama,Romance
Othello (1995)	3.0	Drama
Dangerous Minds (1995)	3.5	Drama
Babe (1995)	1.0	Children,Drama
Clueless (1995)	4.0	Comedy,Romance
Mortal Kombat (1995)	3.0	Action,Adventure,Fantasy
Seven (a.k.a. Se7en) (1995)	4.0	Mystery,Thriller
Pocahontas (1995)	4.0	Animation,Children,Drama,Musical,Romance
Usual Suspects, The (1995)	3.0	Crime,Mystery,Thriller
Mr. Holland's Opus (1995)	3.5	Drama
Bio-Dome (1996)	1.5	Comedy
From Dusk Till Dawn (1996)	4.0	Action,Comedy,Horror,Thriller
Broken Arrow (1996)	3.5	Action,Adventure,Thriller
Happy Gilmore (1996)	4.5	Comedy
Braveheart (1995)	2.5	Action,Drama,War
Down Periscope (1996)	3.0	Comedy
Up Close and Personal (1996)	3.5	Drama,Romance
Birdcage, The (1996)	3.0	Comedy
Apollo 13 (1995)	3.0	Adventure,Drama,IMAX
Batman Forever (1995)	3.0	Action,Adventure,Comedy,Crime



CATEGORIE RATED

Genre	Count
Adventure	290
Animation	80
Children	105
Comedy	530
Fantasy	140
Romance	285
Drama	485
Action	400
Crime	185
Thriller	330
Horror	100
Mystery	80
Sci-Fi	200
War	50
Musical	55
Documentary	30
IMAX	55
Western	20
Film-Noir	5
(no genres listed)	5



I metodi per effettuare la predizione sono quelli descritti nei paragrafi precedenti

- user based top 10 – sono scelti i 10 utenti più simili all'utente target.
- user based utenti con similarità maggiore di x – sono scelti tutti gli utenti che hanno un valore di similarità maggiore uguale a x.
- item based – non effettua nessun filtraggio dei top k item più simili
- user based without mean rating – variante dello user based in cui non si vuole ipotizzare un vero e proprio rating ma solamente una scala di preferenze.

PREDIZIONE PER L'UTENTE 68 - APPROCCIO USER-BASED WITHOUT MEAN RATING

RATING PREDETTO		
MOVIE	RATING PREDETTO	GENERE
Braveheart (1995)	5.000000	Action,Drama,War
Fight Club (1999)	4.939394	Action,Crime,Drama,Thriller
Schindler's List (1993)	4.726707	Drama,War
Jurassic Park (1993)	4.696636	Action,Adventure,Sci-Fi,Thriller
Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)	4.612621	Action,Adventure
Godfather, The (1972)	4.504313	Crime,Drama
Saving Private Ryan (1998)	4.495479	Action,Drama,War
Seven (a.k.a. Se7en) (1995)	4.405302	Mystery,Thriller
American Beauty (1999)	4.387012	Drama,Romance
Lord of the Rings: The Fellowship of the Ring, The (2001)	4.277367	Adventure,Fantasy
Lord of the Rings: The Return of the King, The (2003)	4.104454	Action,Adventure,Drama,Fantasy
Fargo (1996)	4.008208	Comedy,Crime,Drama,Thriller
Toy Story (1995)	3.979599	Adventure,Animation,Children,Comedy,Fantasy

PREDIZIONE PERSONALIZZATA PER TE

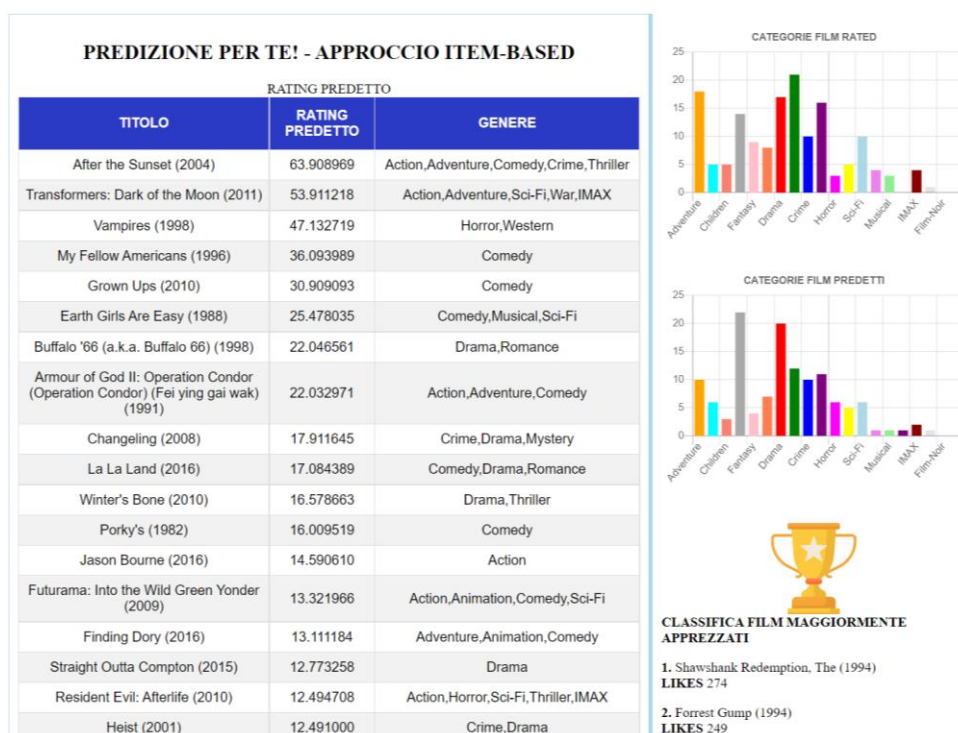
La nostra utenza risulta essere nuova per il sistema: per quanto detto nel paragrafo relativo ai notebook jupyter è buona norma chiedere all'utente di recensire gli item che hanno maggiore entropia, ovvero maggiore contenuto informativo. Questo è ciò che viene mostrato all'utente.

TORNAAL SISTEMA

Purtroppo non conoscendo ancora i tuoi gusti ti chiedo di valutare i seguenti film.
Quelli che vedi sono i film con maggiore *entropia*, ovvero quei film che hanno dei rating molto contrastanti tra loro.
Inserisci una votazione da 1 a 5 - dove 0 indica il fatto di non aver visto quel film

TITOLO Forrest Gump (1994) GENERE Comedy,Drama,Romance,War VOTO: <input type="text"/>
TITOLO Shawshank Redemption, The (1994) GENERE Crime,Drama VOTO: <input type="text"/>
TITOLO Pulp Fiction (1994) GENERE Comedy,Crime,Drama,Thriller VOTO: <input type="text"/>
TITOLO Silence of the Lambs, The (1991) GENERE Crime,Horror,Thriller VOTO: <input type="text"/>
TITOLO Matrix, The (1999) GENERE Action,Sci-Fi,Thriller VOTO: <input type="text"/>

In base ai voti assegnati ai film ai vari film, è effettuata una predizione di classifica con l'approccio item based.



VALUTAZIONE

È possibile valutare il sistema – per semplicità solo l’approccio user based – considerando varie metriche come il Mean Absolute Error (MAE), Root Mean Square Error (RMSE) e il Normalized MAE (NMAE). Il MAE misura la deviazione media – o semplicemente l’errore – nella valutazione prevista rispetto alla valutazione reale.

Sia I un insieme di triple (i_j, r_j, r_j^P) – dove i_j è un film e r_j è il voto assegnato e r_j^P è il voto predetto.

Il MAE si calcola come

$$MAE = \frac{\sum_{(i_j, r_j, r_j^P)} |r_j - r_j^P|}{|I|}$$

L’RMSE si calcola come

$$RMSE = \frac{\sum_{(i_j, r_j, r_j^P)} (r_j - r_j^P)^2}{|I|}$$

L’NMAE si calcola come

$$NMAE = \frac{MAE}{\max r_j - \min r_j} = \frac{MAE}{4.5}$$

VALUTAZIONE APPROCCIO USER BASED WITHOUT MEAN RATING

RATING DELL'UTENTE 68

FILM	RATING UTENTE	RATING USER BASED NO MEANS	RATING USER BASED	RATING ITEM BASED
Toy Story (1995)	2.5	3	3	3
Jumanji (1995)	2.5	2	3	3
Grumpier Old Men (1995)	2.0	1	1	3
Father of the Bride Part II (1995)	2.0	1	1	3
Heat (1995)	4.0	2	1	3
Sabrina (1995)	2.0	1	0	3
GoldenEye (1995)	4.5	2	3	3
American President, The (1995)	4.5	1	1	3
Casino (1995)	3.5	1	2	3
Sense and Sensibility (1995)	3.5	1	0	3
Four Rooms (1995)	2.0	0	0	3
Ace Ventura: When Nature Calls (1995)	1.5	1	2	3
Leaving Las Vegas (1995)	3.5	1	0	3
Othello (1995)	3.0	0	0	3
Dangerous Minds (1995)	3.5	0	1	3

Mean Absolute Error (MAE): 1.4187545767623309

Root Mean Square Error (RMSE): 3.2054605192500674

Normalized MAE (NMAE): 0.3152787948360735

TORNA AL SISTEMA