

## Da dove Parto:

- URM: mi serve per generare il modello
- Profilo Utente: (una riga dell'URM), mi serve per avere la classificazione dei fileri per un utente.  
Il P.U. cambia in base al tempo.

## A cosa voglio ~~Arrivare~~ Arrivare

- Modello di un determinato URM (dipende solo dall'URM)
- Classifica (Top 10) da restituire all'utente (dipende dal modello e dal P.U.). Può cambiare sia x' cambia il P.U., sia per il Reflussing.

## Pseudo Codice:

Param.algoritmo = Algoritmo1; // l'algo che voglio usare nelle varie computazioni.

Param.param = ... // i parametri delle varie funzioni di model e online

Model = createModel(URM, Param);

Voti = onlineRecom (Profilo, URM, Param);

[N, Classifica] = sort (Voti);

→ Classifica i l'elenco ordinato delle chiavi, da quella con meno voti, a quella con più voti.

Passo avere + classifiche + tempi +. Per evitare il reshuffling:

ParamAntiReshuffling. listaVecchia = VotiVecchi;

... listaNuova = VotiNuovi;

... altri parametri

VotiFinali = antiReshuffling (ParamAntiReshuffling);

[N, ClassificaFinale] = sort (VotiFinali);

## Test:

12

Devo passargli l'URL e l'URLprobe.

Quindi gli passo il solito Param come la tecnica di test (leave1out, topk...) e l'algoritmo da utilizzare (cosine, Knn...)

Il risultato passo:  
- stamparlo a video come grafico  
- esportarlo come csv

## Pseudo Codice:

Param.Tecnica = ... // la tecnica da utilizzare per il test (leave1out, ...)

Param.Algoritmo = ... // l'algoritmo per il modello (cosine, ...)

Param. ... = ... // altri parametri delle varie funzioni

Risultato = test(URL, URLprobe, Param);

plot(Risultato);

esporta2csv(Risultato);

# Interfaccia Grafica

[Genera Raccomandazione]

[test]

## - Genera Raccomandazione

1. URH

- Carica URH:

Definizioni:

→ carica la matrice URH

→ " il vettore di definizioni dei film  
(c'è un film per ogni record) FACOLTATIVA

- Usa URH default

2. Profilo Utente

- Inserisci le preferenze dell'utente:

Film 1	3
Film 2	4
:	

- Usa una specifica riga dell'URH:

- Usa una riga random dell'URH

3. Algoritmo

Seleziona l'algoritmo da utilizzare:  come

Imposta i parametri per l'algoritmo:  :   
 :   
 :

4. Antireshuffling

- Seleziona le nuove preferenze:

Film 1	3
Film 2	6
Film 3	2
:	

Parametri:  :

:

:

- Aggiungi  preferenze random

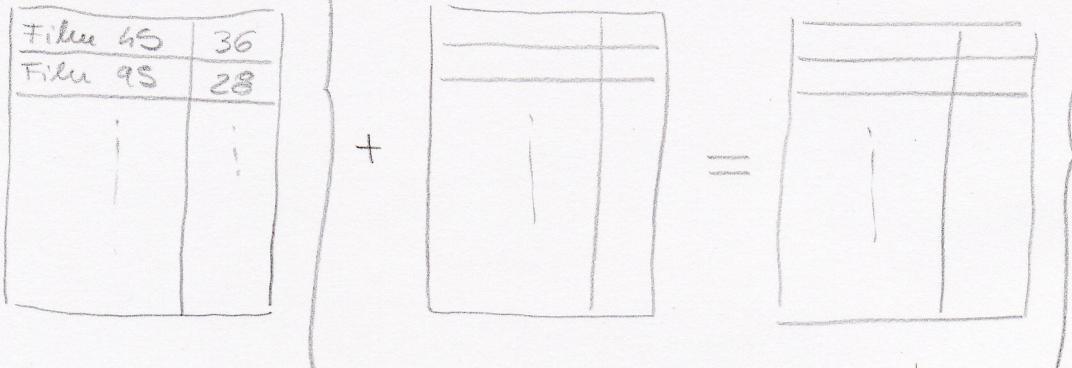
- Aggiungi random preferenze random

- Non effettuare antireshuffling

## 5. Classifica

ANTIRESHUFFLING

L5



Penso voler esportare qualcosa?

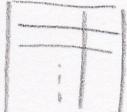
### - Test

1. URH (come l'altro, Carica URH e definizioni, oppure usa default)

(

### 2. Probe

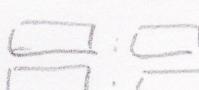
o Carica la matrice Probe

o Modifica le varie righe 

( o Usa Probe di default

### 3. Algoritmo

Seleziona l'algo 

Imposta i parametri: 

(

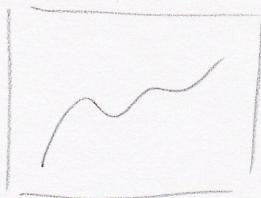
### 4. Metodo di test

Seleziona il metodo di test da utilizzare: 

Imposta i parametri:   


(

### 5. Risultati



[Esporta i dati in csv]

# Documento

- Lo scenario, perché abbiamo bisogno dei sistemi di raccomandazione.
- Cosa sono i SDR e come funzionano.
  - Panoramica dei vari algoritmi di raccomandazione (abbastanza veloce)
  - Panoramica delle tipologie di test (abbastanza veloce)
- Stato del sistema, analisi di cosa (e soprattutto come) è stato fatto fin'ora.
- Reruggerizzazione: studio di come si potra arrivare a un sistema + omogeneo e ordinato, analizzando per cosa vengono usati i vari algoritmi e soprattutto da chi; creazione di "classi di chiamata" facili per l'intente, spostando il livello di parametrizzazione dalla scelta della cartella di file, alla scelta degli attributi da passare alle classi.
- Interfaccia grafica: dove si vuole arrivare con quest'applicazione e perché può servire un'interfaccia grafica da semplifici e visualizza il tutto, sia lato raccomandazione, sia lato test, con relative esportazioni
- Sviluppi futuri: analizzare brevemente le "pseudo-API" introdotte per rendere il sistema espandibile con nuovi algoritmi / classi di test. Analizzare poi i possibili (?) sviluppi futuri che possono avere un'applicazione del genere.