**Sistema di raccomandazione per un social network musicale**

Exam Specifications

Data Analysis and Classification 2019/2020

Lecturers: M. Papandrea, S. Giordano, L. Luceri

Ti viene richiesto di costruire un sistema che è in grado di consigliare la musica agli utenti appartenenti a un social network musicale.

**The Dataset:**

Il dataset che si riceve contiene informazioni sui social network, sui tag e sull'ascolto di artisti musicali da un sistema musicale online. Per ogni utente nel set di dati contiene un elenco dei suoi artisti più ascoltati, incluso il numero di volte in cui tali artisti sono stati suonati. Include anche i tag applicati dall'utente agli artisti. Il set di dati contiene anche informazioni sul social network delle persone nella rete musicale.

Il set di dati che si riceve è composto dai seguenti file.

**artists.csv**

Colonne:​ <id, name>

Questo file contiene informazioni sugli artisti musicali nel set di dati, che sono stati ascoltati e taggati dagli utenti. Ogni riga del file rappresenta un artista.

* L'ID colonna contiene l'ID dell'artista.
* Il nome della colonna contiene il nome dell'artista.

**tags.csv**

Colonne:​ <tagID, tagValue>

Questo file contiene l'insieme di tag disponibili nel set di dati, che vengono quindi utilizzati dagli utenti per

taggando il tipo di musica degli artisti sulla piattaforma. Ogni riga del file rappresenta un tag:

* Il tagID della colonna contiene l'ID numerico del tag.
* Il tagValue della colonna contiene il valore testuale del tag (es. metal, classico e così via).

**user\_artists.csv**

Colonne: <​userID, artistID, weight​>

Questo file contiene gli artisti ascoltati da ogni utente. Ogni riga del file rappresenta una tupla

user-artist:

* La colonna userID contiene l'ID numerico di un utente.
* La colonna artistID contiene l'ID numerico di un artista.

**user\_taggedartists.csv**

Colonne: ​<userID, artistID, tagID, day, month, year>

**user\_taggedartists\_timestamps.csv**

Colonne: ​<userID, artistID, tagID, timestamp>

Questi file contengono le assegnazioni di tag di artisti forniti da ogni particolare utente. Inoltre, essa

contiene le informazioni sulla data (o rispettivamente il timestamp) in cui l'assegnazione dei tag

stato fatto.

* La colonna userID contiene l'ID numerico di un utente.
* La colonna artistID contiene l'ID numerico di un artista.
* Il tagID di colonna contiene il tag assegnato dall'utente all'artista.
* Le colonne < giorno, mese, anno > contengono la data dell'assegnazione del tag.
* Il timestamp della colonna contiene il timestamp dell'assegnazione di tag

**user\_friends.dat**

Colonne: <​userID, friendID​>

Questo file contiene le relazioni di amicizia tra gli utenti nel database.

* La colonna userID contiene l'ID numerico di un utente.
* La colonna friendID contiene l'ID numerico dell'amico di un utente (un altro utente sulla musica Social Network).

Il set di dati contiene anche una parte di test composta da un singolo file. Questo file sarà

per la generazione del file dei risultati, da presentare a seguito del

algoritmi di raccomandazione (vedere le attività 4 e 6).

**test\_user\_artists.csv**

Colonne: <​userID, artistID​>

Ogni riga del file rappresenta un paio di user-artist:

* La colonna userID contiene l'ID numerico di un utente.
* La colonna artistID contiene l'ID numerico di un artista. Attività

Dato il set di dati, viene richiesto di lavorare sulle attività descritte di seguito.

**Task1: EDA**

Esplorare il set di dati (forma di ogni file, tipi di dati, distribuzione dei valori, ...). Sotto alcuni

suggerimento su un'analisi esplorativa iniziale.

1. Calcolare il numero di utenti e artisti presenti nella rete.

2. Calcolare il numero di coppie relazioni utente-amici bidirezionali.

3. Calcolare le relazioni di amicizia medie per utente.

4. Calcolare il numero di relazioni artisti ascoltate dagli utenti (coppie utente-artista).

5. Tracciare la distribuzione del numero di artisti ascoltati da ciascun utente e calcolare il

media e modalità di distribuzione.

6. Tracciare la distribuzione del numero di assegnazioni di tag per ogni artista e calcolare

media e modalità di distribuzione.

7. Tracciare la distribuzione del numero di assegnazioni di tag per ogni utente e calcolare

media e modalità di distribuzione.

8. Tracciare la distribuzione del numero di assegnazioni di tag distinti per ogni artista, e

calcolare la media e la modalità di distribuzione.

**Task2: Clustering tags**

Clustering tag musicali in base all'associazione artists (content vectors).

1. Tag cluster in base all'associazione con gli artisti . Giustifica le tue scelte e commenti sui risultati. Prendere in considerazione almeno due algoritmi di clustering diversi in base alla somiglianza tra i tag e confrontare il clustering.

2. Associare a ogni cluster un meta-tag, e prendere in considerazione l'utilizzo come tag per il social network musicale.

3. Raggruppa gli artisti musicali in base al metatag e analizza la distribuzione degli artisti rispetto ai metatag correlati.

**Task3: Social network analysis**

1. Calcolare le proprietà di rete: distribuzione dei gradi, densità, diametro.

2. Calcolare le misure di centralità, selezionare quella che si considera più rappresentativa e giustificare la scelta.

3. Estrarre:

- Le comunità basate sul social network

- Le comunità basate sulla somiglianza (ad es. artisti preferiti, tipo di musica)

4. Calcolare il coefficiente di clustering globale di:

: le comunità

(OPZIONALE) confrontarli e discutere il risultato

5. Calcolare i coefficienti di clustering locali ed estrarre la media, entro:

- la rete globale

- le comunità

(OPZIONALE) confrontarli e discutere il risultato

6. Calcolare la somiglianza tra gli utenti in base a:

- I loro amici comuni

- L'artista (s) ascoltato

- Tipo di musica

- (FACOLTATIVO) L'artista/i preferito dai loro amici

- (FACOLTATIVO) Il tipo di musica ascoltata dai loro amici

**Task4: Binary classifier for inferring users's artists interest.**

Costruire un classificatore binario che è in grado di dedurre, dato un tupla <user-artist>, se l'utente

interessato all'artista o no. Per creare il classificatore prendere in considerazione un insieme di

funzionalità previste) che caratterizzano un utente-artista tupla. Alcuni esempi di funzioni sono

elencati di seguito, data una tupla di utente e artista A.

- Caratteristica 1: popolarità dell'artista A.

- Caratteristica 2: popolarità dei cluster N C(i), a cui appartiene l'artista A.

- Caratteristica 3: numero di artisti ascoltati dall'utente U, appartenenti agli stessi cluster C(i) di

l'artista A.

- Caratteristica 4: numero di amici dell'utente U, che ascoltava l'artista A.

- Caratteristica 5: booleano che rappresenta se l'artista Una somiglianza con gli artisti più ascoltati dell'utente è

inferiore a una certa soglia di distanza d .

- Caratteristica 6: le comunità calcolate nell'Attività 3.

**Task5: (OPTIONAL) Scalable classifier**

Analogamente all'Attività 4, creare un classificatore in grado di dedurre, dato un tupla <user-artist> , se l'utente sarebbe interessato all'artista o meno, ma anche considerare che i nuovi utenti possono entrare nella rete. In questo senso, la soluzione proposta dovrebbe essere in grado di ridurre al minimo la fase di allenamento, cioè: dovrebbe essere evitata per dover riavviare la fase di allenamento da zero. Per creare un classificatore dinamico e scalabile di questo tipo considera solo una parte della rete originale (ad esempio 3/4 della rete originale) e quindi simulare per aggiungere gli utenti rimanenti. Mostrare le prestazioni del classificatore proposto e confrontarlo con le prestazioni di quello prodotto in Task4 nel caso di 5/6 della rete originale all'inizio e il restante 1/6 che entra in rete in seguito.

**Task6: Inferring users interest in artist.**

Costruire un sistema che è in grado di raccomandare il peso di una coppia <user-artist> , che è il numero di volte che un utente sta per ascoltare questo artista. La raccomandazione deve essere categorica e i valori possibili sono i seguenti.

- HIGH: peso superiore a 1000 (alto interesse per l'artista).

MEDIA: peso compreso tra 1 e 1000 (interesse medio per l'artista).

LOW: peso uguale a zero (nessun interesse per l'artista).

Per creare il modello di classificazione, prendere in considerazione un set di funzionalità (sono previste almeno 10 funzionalità) che caratterizzano un utente-artista tupla. Alcuni esempi di funzionalità sono elencati nell'attività 4.

**Submission Details**

Il set di dati di test contiene un file test\_user\_artists .csv . Questo file contiene un elenco di tuple

< utente U - artista A > . (Attività4) Per ogni tupla <U,A> stima 1 (True) se si prevede che l'utente sarebbe interessato ad ascoltare l'artista A, prevedere 0 (False) se si prevede che l'utente non sarebbe interessato ad ascoltare l'artista A.

(Attività6) Per ogni tupla <U,A> prevedere HIGH (2) se si prevede che l'utente sarebbe altamente interessato ad ascoltare l'artista A, prevedere MEDIA (1) se si prevede che l'utente avrebbe medio interessato ad ascoltare l'artista A, prevedere LOW (0) se si prevede che l'utente avrebbe non sono interessato ad ascoltare l'artista A. Questo risultato deve essere inviato in un file formattato come segue (un esempio è riportato di seguito).

results.csv

Colonne: ​<userID, artistID, isInterested, weight>

● Le colonne < userID,artistID> corrispondono alle stesse colonne nel file CSV test\_user\_artists e contengono tutte le colonne del file di test (stesso ordine di aspetto).

La colonna isInterested è una colonna booleana, contiene 0 nella corrispondenza di righe il cui

stima è negativa, e 1 nella corrispondenza delle righe la cui stima è positiva.

Questa colonna è correlata all'Attività 4.

Il peso della colonna è una colonna numerica, contiene 0 nella corrispondenza delle righe previste

con interessi LOW, contiene 1 in corrispondenza di interesse MEDIUM e contiene 2 in

corrispondenza di interessi HIGH. Questa colonna è correlata all'Attività 4.

------------------------------------------------------------------------------------------------------------------------

userID artistID isInterested weight

2 52 0 0

2 53 1 2

230 53 0 1

510 120 1 1

714 199 0 0

------------------------------------------------------------------------------------------------------------------------**Submission Dates**

L'utente è richiesto di presentare entro il 23 gennaio:

1. Il file result.csv descritto sopra;

2. Tutto il codice sorgente che hai scritto per generare il file result.csv;

3. Un documento di testo che descrive l'algoritmo, i ragionamenti e le

lavoro per ogni attività presentata sopra. Questo documento non deve superare le 10 pagine e

copia stampata di questo documento può essere portato con voi all'esame scritto di gennaio

27.

**Evaluation**

La valutazione dei risultati della classificazione sarà fornita in termini di precisione, precisione e richiamo, calcolata sulla GT associata al file di test.

È possibile inviare il file dei risultati prima dell'invio finale dell'intero Progetto in qualsiasi momento,

e più volte (massimo 5 volte per studente) prima dell'invio finale, al fine di ottenere la valutazione aggiornata dei risultati.

Per ottenere i risultati della classificazione, inviare un messaggio di posta elettronica:

- per michela.papandrea@supsi.ch

- oggetto: DaCla Final Project, tenta <N> <Cognome> (es. DaCla Final Project, tentativo 1 Papandrea )

- Allegato: result.csv la formattazione del file è specificata sopra