**Tema del Proyecto**

Recommender System: Songs

**Integrantes**

20114297 Diego Pavel Vargas Flores

20140880 Ronie Paolo Arauco Alarcón

20142145 Enrique André Pando Robles

**Calificación**

|  |  |  |
| --- | --- | --- |
| **Criterios** | **Puntaje Máx.** | **Nota** |
| **Formato, estructura y extensión del informe:**  (1) Todas las secciones están incluidas (Resumen, Introducción, Descripción del Dataset, Experimentación, Resultados y Discusión, Conclusiones, Referencias)  (2) La extensión es válida (**4-5 páginas o 2000 palabras máximo**). Si desean añadir gráficos o tablas adicionales que sobrepasan la extensión, pueden ponerlas como **anexo** al final del documento. | +2p |  |
| **Legibilidad del contenido:**  El informe presenta buena **redacción** y correcta **ortografía**. | +2p |  |
| **Evidencia del experimento:**  Se adjuntan **archivos** y **código** que evidencie la obtención de los resultados presentados. | +4p |  |
| **Contenido: Resumen, Introducción**  (1) El resumen es conciso y explica brevemente el contenido de todo el informe (desde el tema hasta los resultados).  (2) La introducción presenta el tema que se ha desarrollado en el trabajo. | +2p |  |
| **Contenido: Descripción del Dataset**  Se explica brevemente el dataset analizado y/o generado para el trabajo, y se destacan sus aspectos más importantes. | +3p |  |
| **Contenido: Experimentación**  Se explica el enfoque propuesto y los pasos que se llevaron a cabo en el trabajo | +3p |  |
| **Contenido: Resultados y Discusión**  Se presentan todos los resultados obtenidos a partir de sus experimentos y se interpretan los resultados.  Se colocan **tablas** y/o **gráficos** que ayudan a la explicación de estos resultados. | +3p |  |
| **Contenido: Conclusiones, Referencias**  (1) Se presentan conclusiones concisas en base a los experimentos y resultados obtenidos  (2) Se colocan referencias de las librerías o programas utilizados (con el formato respectivo), así como de trabajos similares en los que haya basado su experimentación. | +1p |  |
|  | **NOTA FINAL** |  |

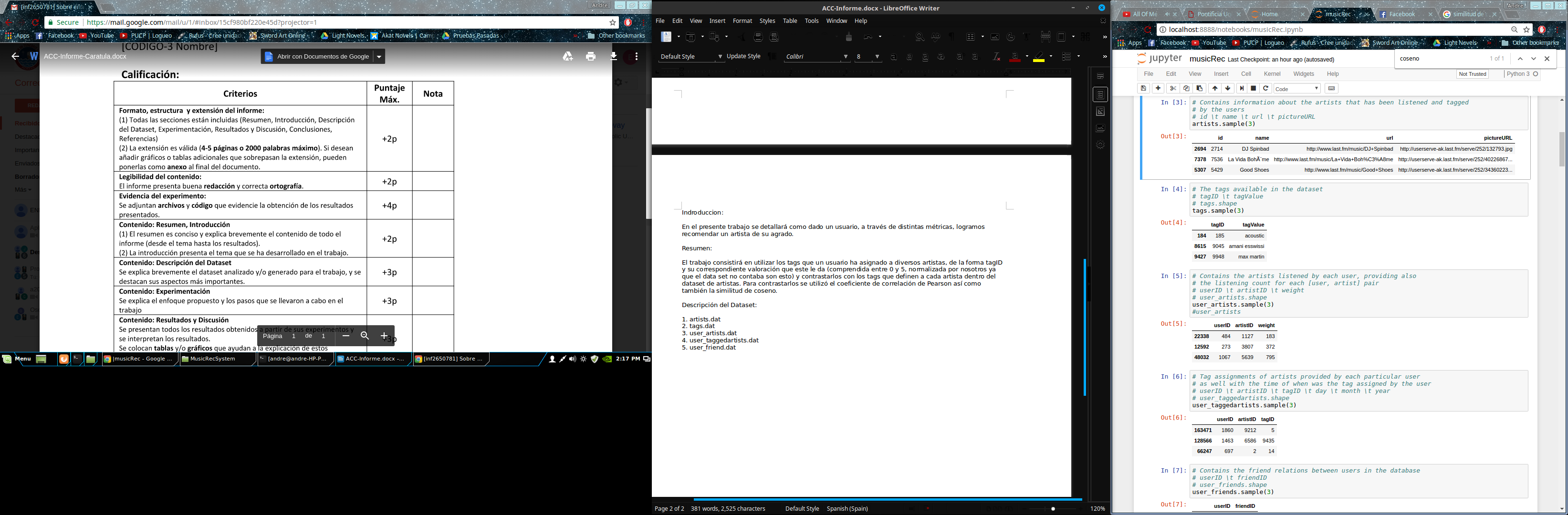
**Introducción**

En el presente trabajo se detallará como dado un usuario, a través de distintas métricas, logramos recomendar un artista de su agrado.

**Resumen**

El trabajo consistirá en utilizar los tags que un usuario ha asignado a diversos artistas, de la forma tagID y su correspondiente valoración que este le da (comprendida entre 0 y 5, normalizada por nosotros ya que el data set no contaba con esto) y contrastarlos con los tags que definen a cada artista dentro del data set de artistas. Para contrastarlos se utilizó el coeficiente de correlación de Pearson, así como también la similitud de coseno.

**Descripción del data set**



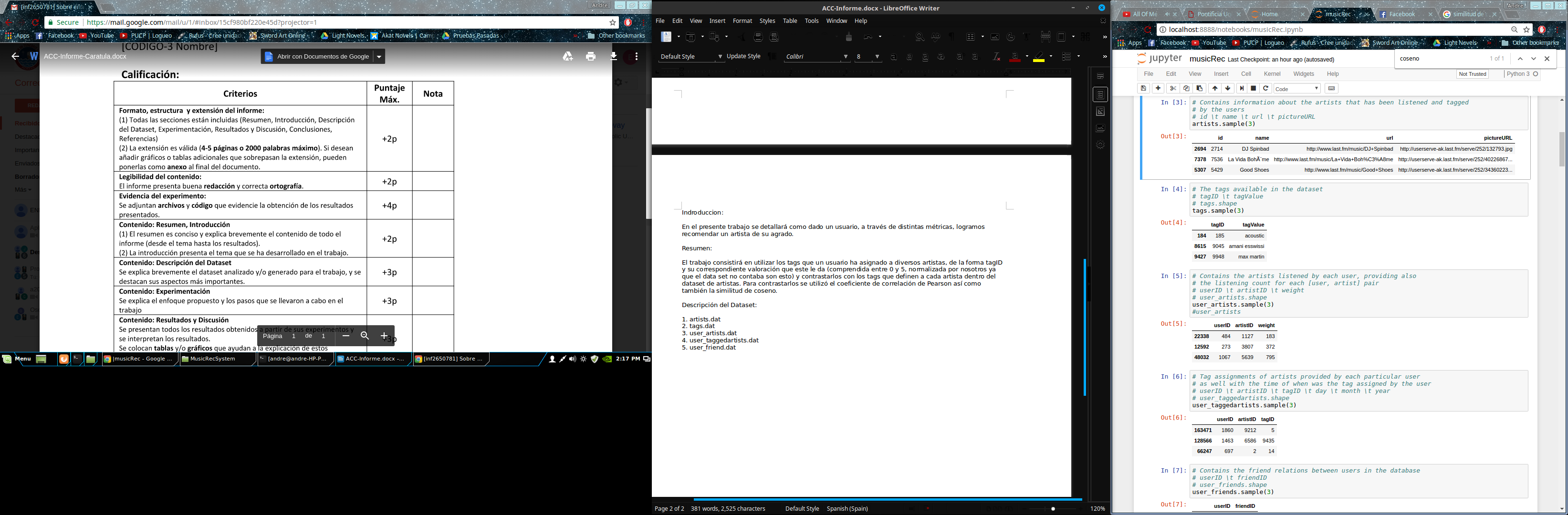
Imagen 1. Data set de los artistas.

Imagen 2. Data set de los tags.

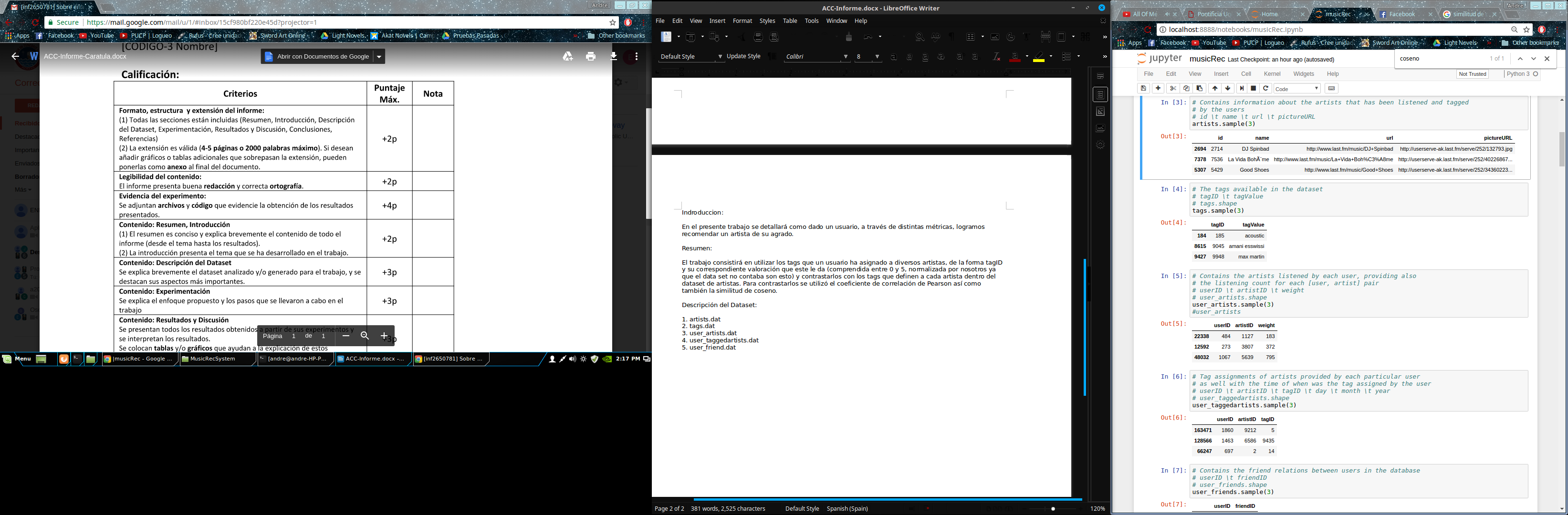


Imagen 3. Data set de los usuarios. Contiene la cantidad de veces que un usuario a reproducido a cierto artista.

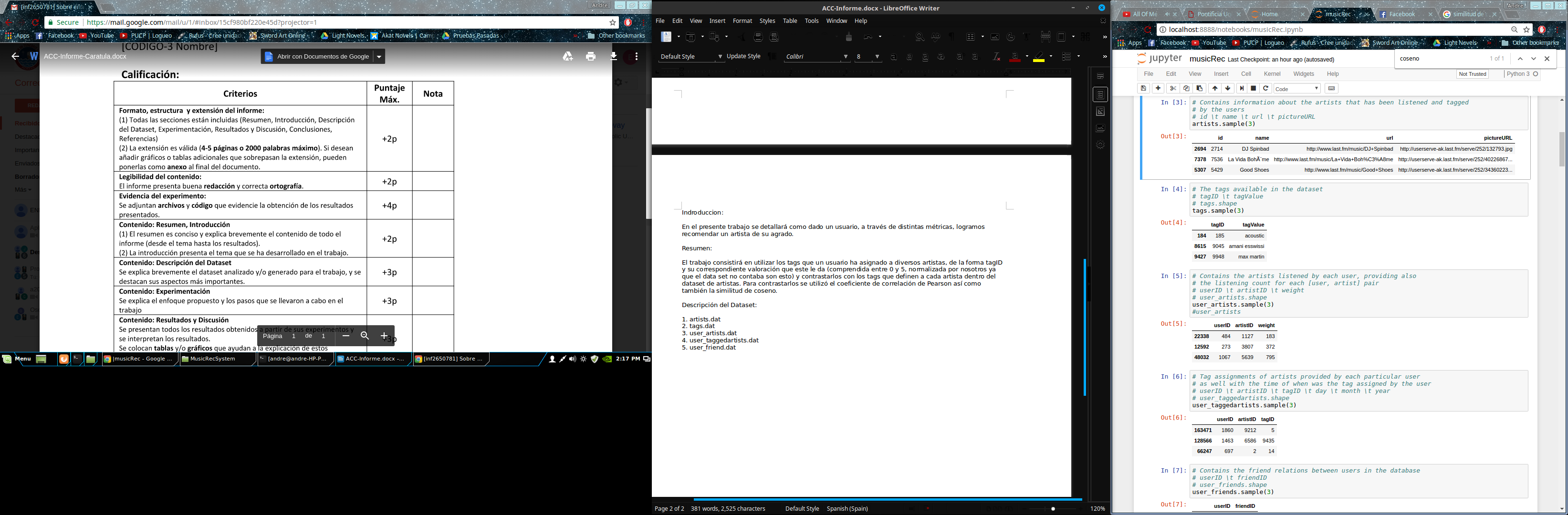


Imagen 4. Contiene los tags asignados por cada usuario a un artista

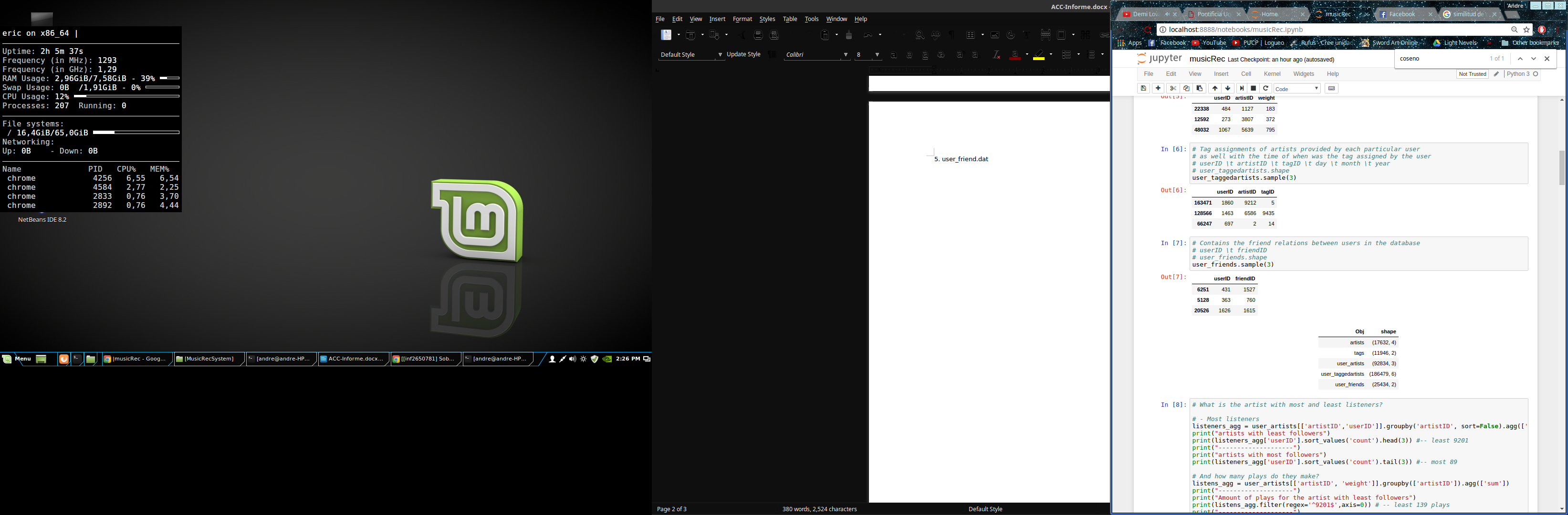
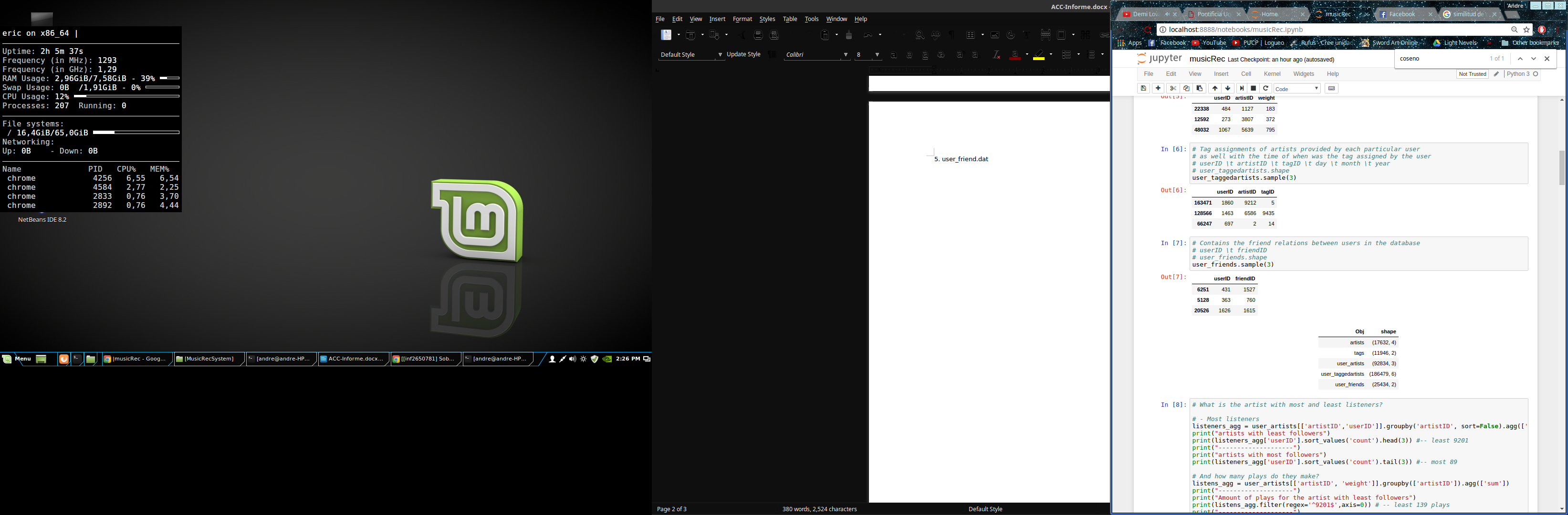


Imagen 5. Contiene los amigos de cada usuario.

Finalmente, un *shape* de la data que tenemos disponible:



**El problema**

El *data set* no contenía ninguna columna de calificación a un artista, es decir, el usuario no califica a un artista con un puntaje, por lo que se tuvo que lidiar con este problema. Lo que sí había era una columna de “peso”, que significaba cuántas veces un usuario había escuchado un artista y cuántas veces los usuarios le habían asignado un *tag* a un artista. Por lo tanto, se optó por ordenar los datos como se presentan en la tabla 1 y plantear dos fórmulas para generar un interés de un usuario hacia un artista y así normalizar los datos.

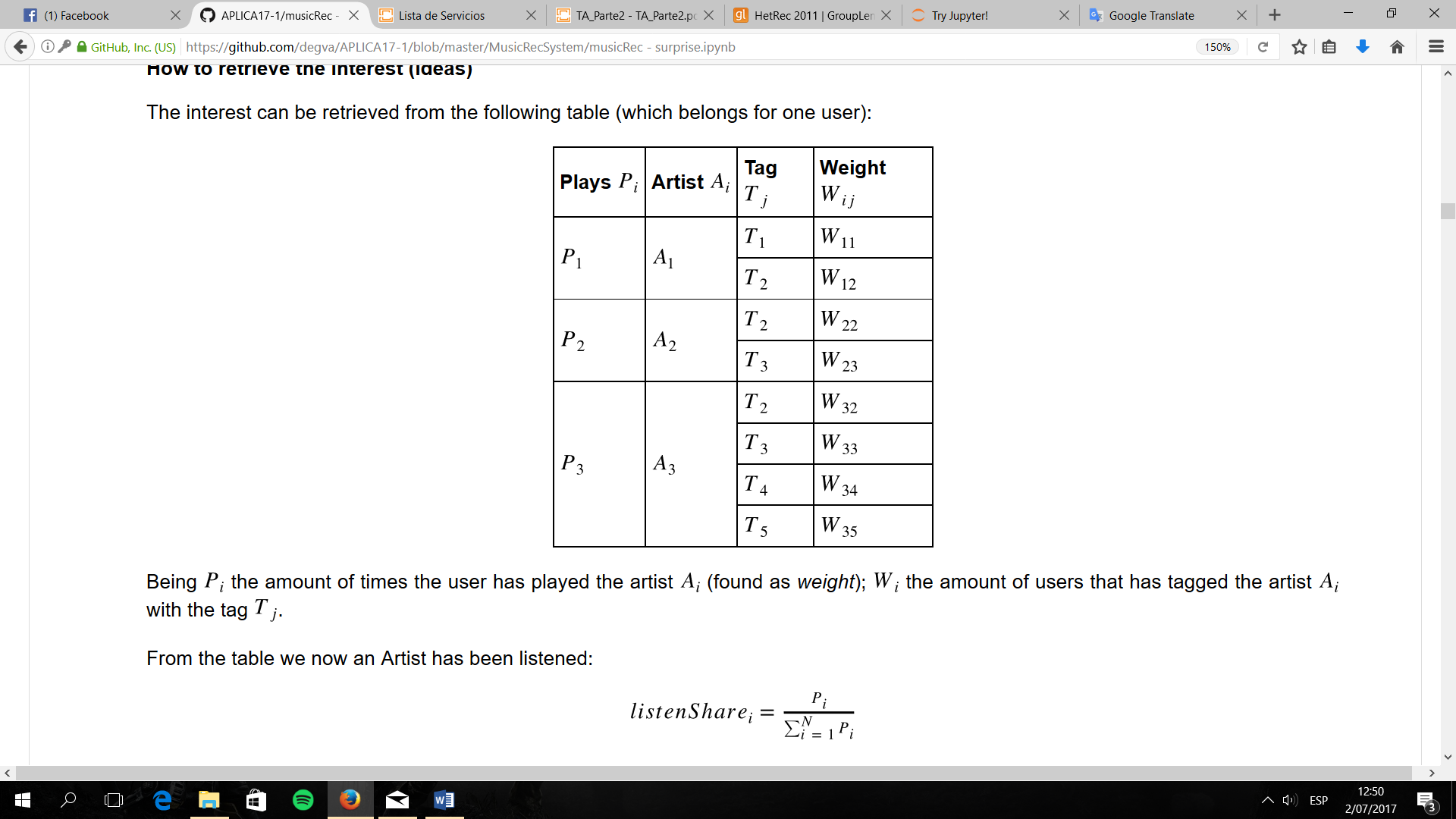
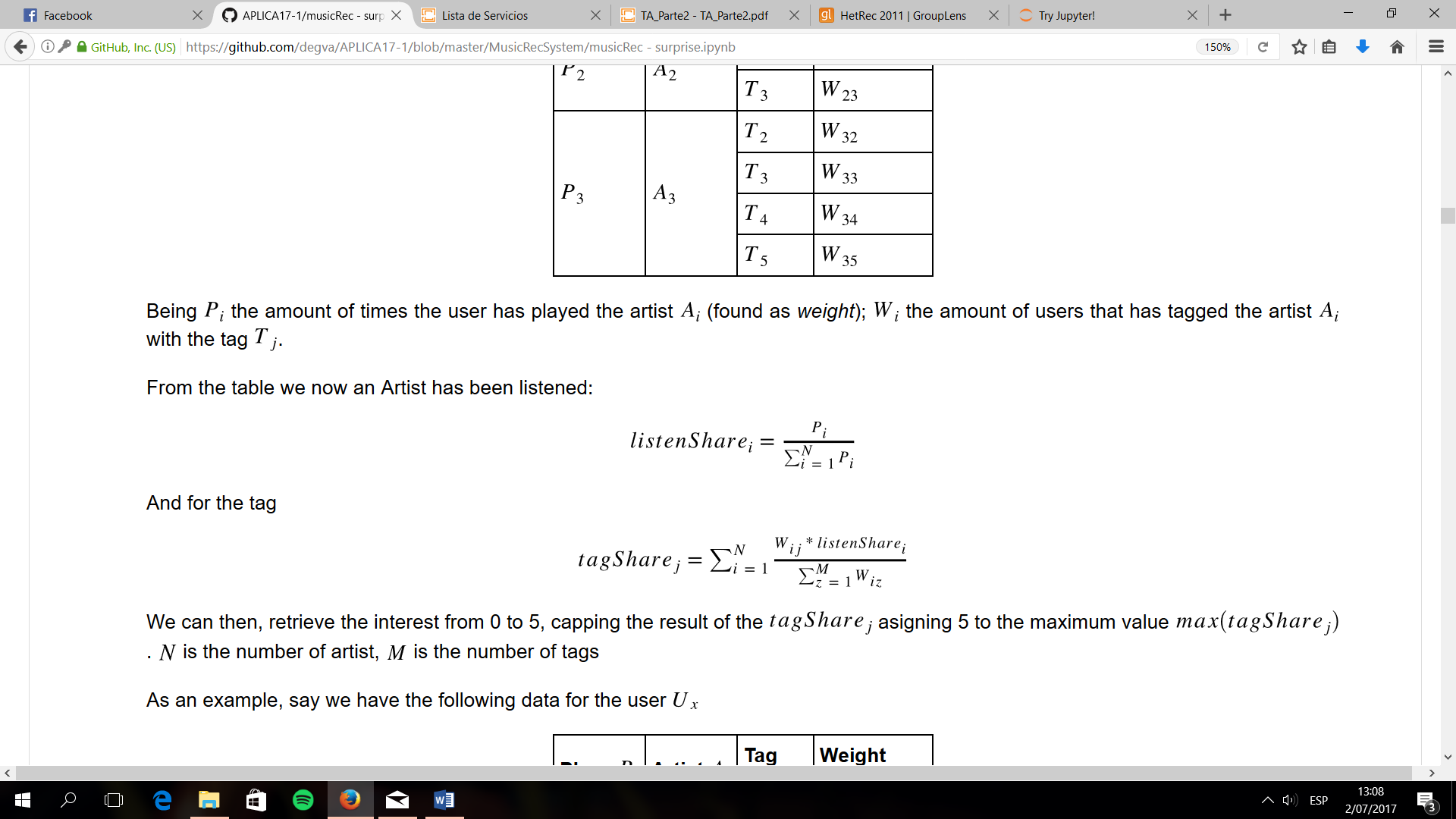


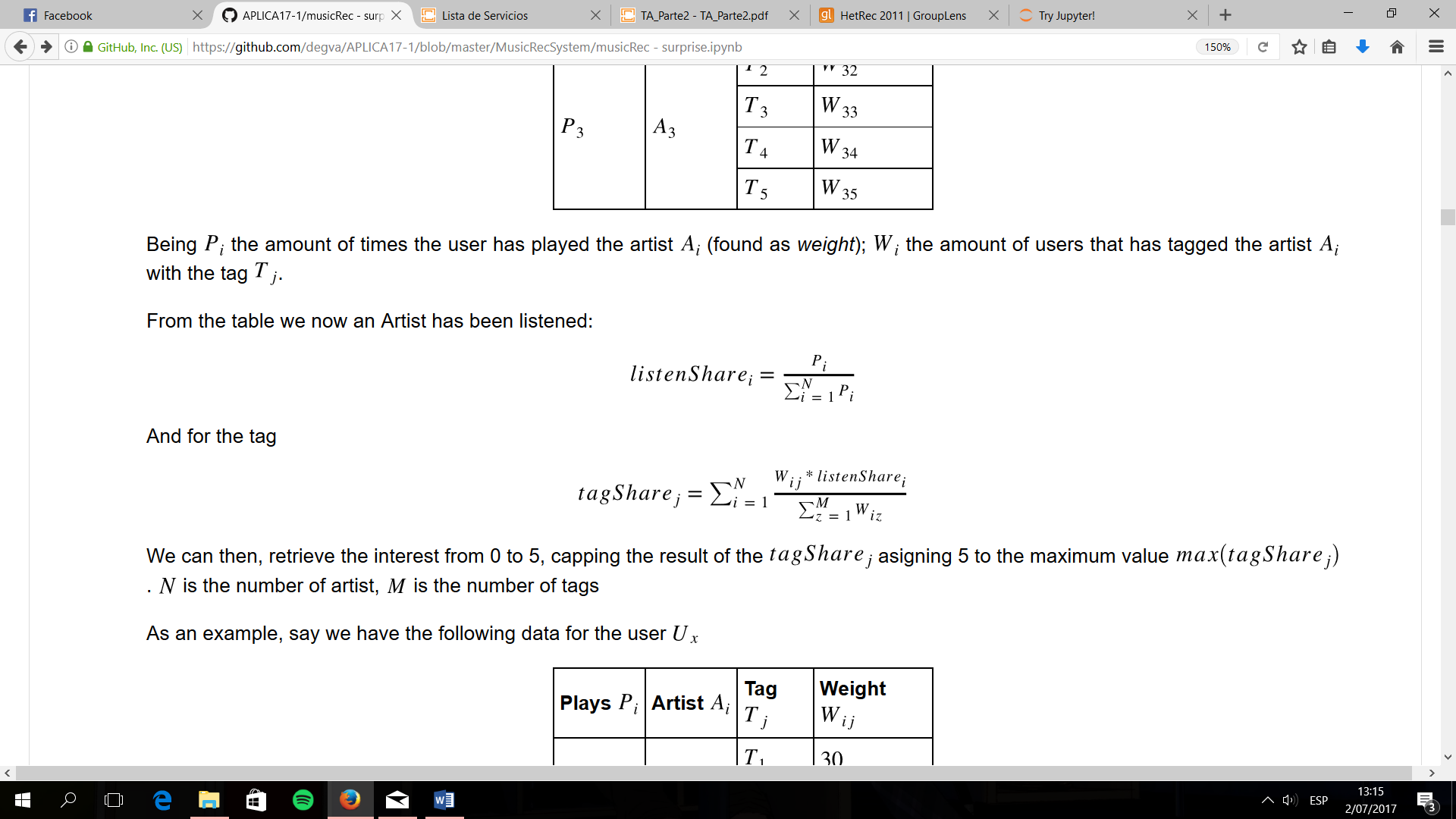
Tabla 1. Información de un usuario. Se muestra la cantidad de reproducciones por artista, los artistas y sus tags con sus respectivos pesos.

**Obteniendo el interés**

Siendo *Pi*, la cantidad de veces que un usuario ha reproducido un artista *Ai* (encontrado como “Peso” en el data set) y siendo *Wij* la cantidad de veces que los usuarios le han asignado el tag *Tj* al artista *Ai*, donde N es el número de artistas y M el número de tags, se obtiene:

****

Fórmula 1. La cantidad de veces que el artista *Ai* ha sido escuchado por el usuario.

****

Fórmula 2. El interés de un usuario *Ux* hacia un artista *Ai* respecto a cada tag *Tj*.

Ahora, se puede establecer el *tagSharej* en un rango del 0 al 5 para un usuario *Ux*:

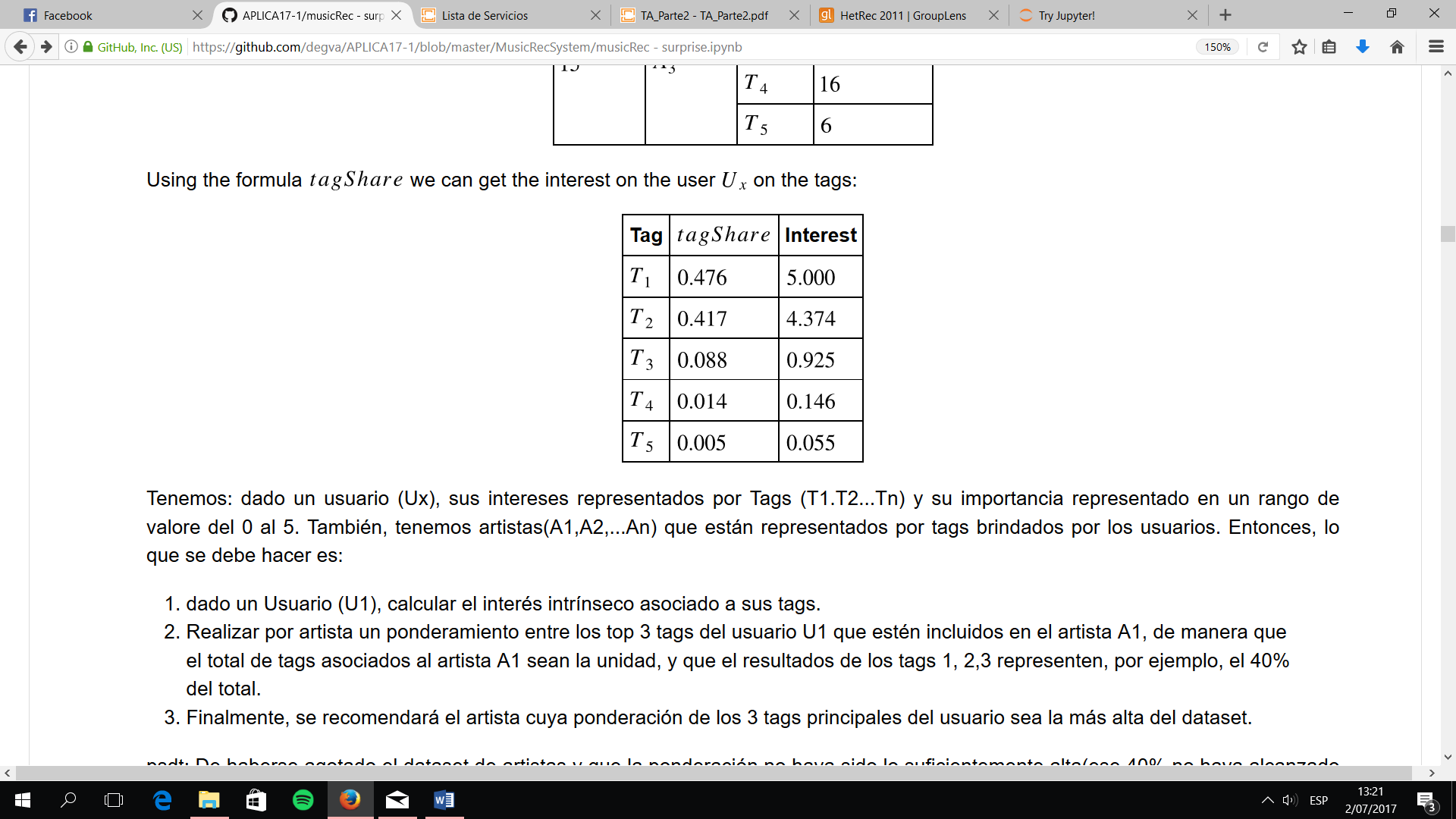


Tabla 2. Se muestra cada tag con su respectivo *tagShare*. Se le asigna 5 al máximo *tagShare*: *max(tagSharej)*.

**Experimentación**

Primero, tuvimos que normalizar los datos ya que solo contábamos con los tags que un usuario asignaba a determinado artista y la cantidad de veces que el usuario escuchaba a tal artista, por ello usamos el *listenShare* que básicamente era dividir el peso (cantidad de veces que un usuario U escucha a un artista A) entre la cantidad total que el usuario ha escuchado a todos los artistas. Con ellos teníamos un indicador que, dado del total de veces que el usuario U ha escuchado a diversos artistas, qué porcentaje de esto le corresponde al artista A.

Después, teníamos que sacar una métrica que clasificara el valor que un usuario le daba a determinado tag (de 0 a 5). Para ello se usó el tagShare, el cual procederé a describir. El tagShare se calcula una vez obtenido el listenShare por artista, ya que se multiplicará este por el número de veces que un artista ha sido *taggeado* con el tag T y todo esto será dividido entre el total de veces que los artistas que escucha el usuario han sido taggeados con el tag T.

Al final de este proceso tendremos un tagShare por cada tag del usuario. Al mayor tagShare se le asignará un puntaje de 5 y se calculará el valor del resto con respecto al mayor.

Una vez claro esto, procedimos a hallar la correlación entre artistas dado sus tags con Pearson, utilizando la siguiente línea de código:

pM1 = user\_taggedartists.pivot\_table(index = ['tagID'], columns = ['artistID'], aggfunc = 'count')

Esto nos dará una matriz donde las filas sean los tags, las columnas los artistas y el valor que almacena cada celda será el número de veces que un artista ha sido taggeado.

Con la matriz ya hecha, se normalizó la cantidad de veces entre 0 a 5 para suavizar los datos, transformando pM1 en M1. Luego se convirtió el diccionario de user\_tag\_interest (que contenía todos los tags que a un usuario le gustaban y su respectiva valoración comprendida entre 0 y 5) en un dataFrame.

Finalmente, a través de recs3 = get\_recs2(f[238], M1, 10) tendremos en recs3 una lista de los 10 mejores artistas recomendados para un usuario (en este caso el usuario con ID = 238), los cuales serán seleccionados, basado en sus tags, a través de Pearson. Esto es, compararlos con los tags de los artistas comprendidos en M1.

**Conclusiones**

Intentamos realizar esto solamente con el número de veces que el usuario había escuchado a los artistas sin considerar los tags, resultándonos en una relación muy variable (entre 60% a 80%). Sin embargo, con los tags nos encontramos con una relación mayor al 80% en los 10 primeros artistas recomendados. No solo eso, sino que también nos percatamos que la lista de artistas recomendados que brindábamos era muy similar a los recomendados en www.last.fm.

**Bibliografía**

1. Robillard, M., Maalej, W., Walker, R. J., & Zimmermann, T. (Eds.). (2014). Recommendation Systems in Software Engineering. Springer Berlin Heidelberg. Cap. 2 p. 20-21 <https://doi.org/10.1007/978-3-642-45135-5>

2. Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2011). Recommender systems: an introduction. Cambridge University Press (Vol. 40). <https://doi.org/10.1017/CBO9780511763113>