Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Sistemas de recomendación: Métodos de filtrado colaborativo

Óscar Navarro Mesa alu0101504094@ull.edu.es
Alejandro Javier Aguiar Pérez alu0101487168@ull.edu.es
Leandro Samuel Armas Morales alu0101464305@ull.edu.es

Índice

Índice **1**
Introducción **2**
Formato de entrada **3**
Ejecución **3**
Resultados **3**
Comprobación de la fiabilidad **5**
Otros ejemplos **6**
utility-matrix-10-25-1.txt (pearson-media-9 vecinos) 6
utility-matrix-10-25-1.txt (coseno-media-9 vecinos) 7
Observaciones 8
utility-matrix-50-250-1.txt (coseno-simple–40 vecinos) 8
utility-matrix-50-250-1.txt (euclidean-simple–40 vecinos) 9
Observaciones 10
Conclusiones **11**

Introducción

Este proyecto tiene como objetivo desarrollar un sistema de recomendación utilizando técnicas de filtrado colaborativo, un enfoque ampliamente utilizado para ayudar a los usuarios a descubrir contenido que les pueda interesar. Con el volumen de contenido disponible, los sistemas de recomendación se han vuelto esenciales para mejorar la experiencia del usuario al personalizar sugerencias basadas en sus preferencias individuales y en las calificaciones de otros usuarios.

El sistema se basa en el análisis de una matriz de utilidad, que recopila las calificaciones que los usuarios asignan a diversos temas como películas y series. A través de esta matriz, se pueden identificar patrones de calificación y similitudes entre usuarios. Para calcular estas similitudes, se implementan varias métricas, como la correlación de Pearson, la distancia coseno y la distancia euclidiana.

- **Correlación de Pearson**: Evalúa la relación lineal entre dos usuarios, permitiendo identificar aquellos con patrones de calificación similares, sin tener en cuenta sus promedios individuales.
- **Distancia coseno**: Mide la orientación entre dos vectores de calificación, lo que es útil para encontrar usuarios con gustos comparables, sin importar la magnitud de sus calificaciones.
- **Distancia euclidiana**: Calcula la distancia real entre dos conjuntos de calificaciones, proporcionando una forma directa de ver cuán similares son los usuarios en sus preferencias.

A través de estas métricas, el sistema no solo busca predecir calificaciones faltantes en la matriz, sino que también permite a los usuarios recibir recomendaciones personalizadas. Además, este proyecto representa una oportunidad para que los participantes apliquen conceptos de análisis de datos y aprendizaje automático en un contexto práctico, contribuyendo a su formación en el campo de la ingeniería informática y la gestión del conocimiento en las organizaciones.

Formato de entrada

El sistema requiere un archivo de texto (.txt) que contenga la matriz de utilidad, con el siguiente formato:

1. Primera línea: Valor mínimo de calificación.
2. Segunda línea: Valor máximo de calificación.
3. Filas subsiguientes: Calificaciones de los usuarios, donde un guion (-) indica una calificación faltante.

Ejemplo de matriz de utilidad:

matriz

```
0.000
5.000
3.142  2.648  1.649  -  1.116  0.883  0.423  3.976    -    3.143
3.412  0.314  3.796     4.233  2.159  4.513  2.392  0.868  2.473
4.408  4.495  2.052  -  0.051    -    3.355  3.739  4.085    -
```

Ejecución

Para ejecutar el programa, se debe utilizar la línea de comandos de la siguiente manera:

bash

```
python sistema_recomendacion.py ../data/matrix/utility-matrix-test.txt --metrica pearson --
vecinos 2 --prediccion simple --salida resultado.txt
```

Resultados

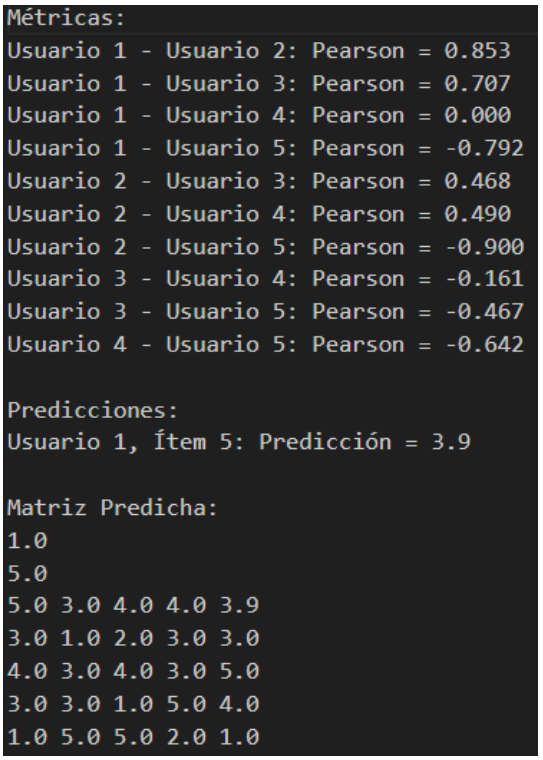
El programa genera un fichero de salida que contiene varias secciones. Primero, se incluyen las métricas calculadas entre los pares de usuarios, indicando las similitudes entre ellos. Luego, se presenta una lista de predicciones realizadas para los ítems faltantes, con el usuario, el ítem correspondiente y el valor de la predicción. Finalmente, se muestra la matriz de utilidad predicha, donde las calificaciones originales se han completado con las predicciones calculadas. El fichero también guarda los valores mínimos y máximos de la escala de puntuación utilizados en la matriz.

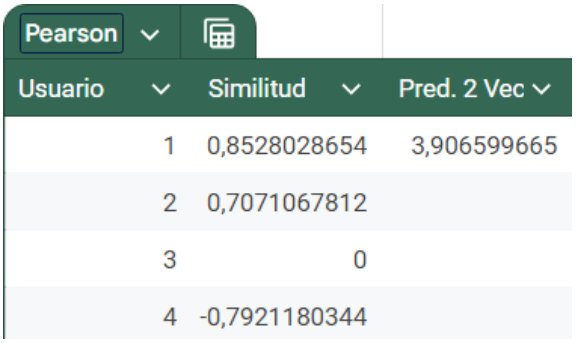
resultado.txt

```
Métricas:
Usuario 1 - Usuario 2: Pearson = 0.853
Usuario 1 - Usuario 3: Pearson = 0.707
Usuario 1 - Usuario 4: Pearson = 0.000
Usuario 1 - Usuario 5: Pearson = -0.792
Usuario 2 - Usuario 3: Pearson = 0.468
Usuario 2 - Usuario 4: Pearson = 0.490
Usuario 2 - Usuario 5: Pearson = -0.900
Usuario 3 - Usuario 4: Pearson = -0.161
Usuario 3 - Usuario 5: Pearson = -0.467
Usuario 4 - Usuario 5: Pearson = -0.642

Predicciones:
Usuario 1, Ítem 5: Predicción = 3.9

Matriz Predicha:
1.0
5.0
5.0 3.0 4.0 4.0 3.9
3.0 1.0 2.0 3.0 3.0
4.0 3.0 4.0 3.0 5.0
3.0 3.0 1.0 5.0 4.0
1.0 5.0 5.0 2.0 1.0
```

Comprobación de la fiabilidad

El programa funciona correctamente con cualquier tamaño de matriz de utilidad siempre que se use el formato correcto, además hemos podido demostrar haciendo los cálculos en las hojas de cálculo de Google que la respuesta del programa es la esperada usando cualquiera de los tres tipos de métricas. En las siguientes imágenes se muestran los resultados del programa con una matriz de utilidad que se encuentra en el repositorio del trabajo comparado con la salida esperada.





Se ha de mencionar que en “data/comparisons” en el repositorio se encuentran todas las salidas comprobadas en la hoja de cálculo.

Otros ejemplos

utility-matrix-10-25-1.txt (pearson-media-9 vecinos)

python3 src/sistema_recomendacion.py data/matrix/utility-matrix-10-25-1.txt --metrica pearson --vecinos 9 --prediccion media --salida prueba.txt

Las predicciones son:

```
Predicciones:
Usuario 1, Ítem 12: Predicción = 2.6
Usuario 1, Ítem 19: Predicción = 2.0
Usuario 1, Ítem 21: Predicción = 2.1
Usuario 2, Ítem 5: Predicción = 2.6
Usuario 2, Ítem 22: Predicción = 2.3
Usuario 3, Ítem 23: Predicción = 3.4
Usuario 4, Ítem 1: Predicción = 2.6
Usuario 4, Ítem 9: Predicción = 2.7
Usuario 5, Ítem 10: Predicción = 0.6
Usuario 5, Ítem 23: Predicción = 2.0
Usuario 6, Ítem 13: Predicción = 1.7
Usuario 6, Ítem 16: Predicción = 2.1
Usuario 7, Ítem 4: Predicción = 2.7
Usuario 7, Ítem 9: Predicción = 4.4
Usuario 7, Ítem 21: Predicción = 2.7
Usuario 7, Ítem 24: Predicción = 2.7
Usuario 8, Ítem 3: Predicción = 2.6
Usuario 8, Ítem 14: Predicción = 1.6
Usuario 8, Ítem 18: Predicción = 2.7
Usuario 8, Ítem 23: Predicción = 1.7
Usuario 8, Ítem 24: Predicción = 2.5
```

La matriz predicha es:

```
Matriz Predicha:
0.0
5.0
0.8 4.6 4.5 1.3 3.0 4.5 0.0 4.4 3.5 1.3 1.1 2.6 0.3 3.4 4.2 2.8 3.3 0.3 2.0 4.2 2.1 0.3 1.6 0.3 1.3
3.3 4.7 1.2 1.0 2.6 4.4 1.5 3.3 3.1 2.7 3.5 4.7 2.8 0.3 2.7 2.3 2.4 3.3 0.4 3.3 3.2 2.3 4.0 1.3 2.8
1.8 1.3 3.6 3.1 4.3 4.4 4.4 0.6 0.6 4.9 2.4 3.1 3.2 2.4 2.2 2.4 4.9 0.9 2.1 3.1 4.6 0.6 3.4 1.2 4.8
2.6 4.4 3.8 3.2 4.2 2.0 4.0 4.8 2.7 4.8 4.3 0.3 0.0 1.2 2.5 4.5 1.4 0.8 0.4 2.0 2.9 3.9 1.5 1.3 0.5
0.7 3.4 3.2 2.2 0.1 2.7 1.6 2.7 4.1 0.6 0.7 1.7 2.7 1.8 1.9 3.2 0.5 3.1 4.5 4.9 1.1 3.1 2.0 2.5 3.4
1.0 4.0 3.4 3.1 1.9 0.9 0.3 0.4 4.0 1.5 3.2 1.6 1.7 4.6 2.1 2.1 2.3 2.8 1.0 4.1 2.0 0.3 4.7 0.6 0.1
3.4 4.7 4.0 2.7 2.8 3.9 0.1 3.0 4.4 0.6 4.1 4.4 2.9 3.9 5.0 2.8 2.0 2.7 5.0 3.6 2.7 1.5 2.7 2.7 3.1
1.0 4.3 2.6 4.5 3.7 1.8 2.2 1.2 1.9 4.8 1.3 4.4 0.5 1.6 4.7 4.4 2.2 2.7 0.1 0.4 0.4 4.1 1.7 2.5 4.0
1.4 0.1 1.4 2.5 4.2 3.6 2.3 3.4 1.7 4.7 3.3 4.5 4.6 1.8 2.6 1.1 4.2 1.5 4.9 1.3 0.2 0.1 4.7 2.9 2.0
3.7 1.2 3.0 2.6 4.0 2.1 4.2 0.5 4.0 3.8 4.3 0.5 3.6 3.9 2.4 1.5 1.1 1.2 4.7 2.6 4.5 2.5 4.0 1.8 0.3
```

utility-matrix-10-25-1.txt (coseno-media-9 vecinos)

python3 src/sistema_recomendacion.py data/matrix/utility-matrix-10-25-1.txt --metrica coseno --vecinos 9 --prediccion media --salida prueba.txt

Las predicciones son:

```
Predicciones:
Usuario 1, Ítem 12: Predicción = 2.5
Usuario 1, Ítem 19: Predicción = 2.2
Usuario 1, Ítem 21: Predicción = 2.1
Usuario 2, Ítem 5: Predicción = 3.2
Usuario 2, Ítem 22: Predicción = 1.9
Usuario 3, Ítem 23: Predicción = 3.5
Usuario 4, Ítem 1: Predicción = 1.9
Usuario 4, Ítem 9: Predicción = 2.9
Usuario 5, Ítem 10: Predicción = 2.9
Usuario 5, Ítem 23: Predicción = 3.1
Usuario 6, Ítem 13: Predicción = 1.8
Usuario 6, Ítem 16: Predicción = 2.3
Usuario 7, Ítem 4: Predicción = 3.2
Usuario 7, Ítem 9: Predicción = 3.5
Usuario 7, Ítem 21: Predicción = 2.9
Usuario 7, Ítem 24: Predicción = 2.1
Usuario 8, Ítem 3: Predicción = 3.1
Usuario 8, Ítem 14: Predicción = 2.5
Usuario 8, Ítem 18: Predicción = 1.8
Usuario 8, Ítem 23: Predicción = 3.2
Usuario 8, Ítem 24: Predicción = 1.5
```

La matriz predicha es:

```
Matriz Predicha:
0.0
5.0
0.8 4.6 4.5 1.3 3.0 4.5 0.0 4.4 3.5 1.3 1.1 2.5 0.3 3.4 4.2 2.8 3.3 0.3 2.2 4.2 2.1 0.3 1.6 0.3 1.3
3.3 4.7 1.2 1.0 3.2 4.4 1.5 3.3 3.1 2.7 3.5 4.7 2.8 0.3 2.7 2.3 2.4 3.3 0.4 3.3 3.2 1.9 4.0 1.3 2.8
1.8 1.3 3.6 3.1 4.3 4.4 4.4 0.6 0.6 4.9 2.4 3.1 3.2 2.4 2.2 2.4 4.9 0.9 2.1 3.1 4.6 0.6 3.5 1.2 4.8
1.9 4.4 3.8 3.2 4.2 2.0 4.0 4.8 2.9 4.8 4.3 0.3 0.0 1.2 2.5 4.5 1.4 0.8 0.4 2.0 2.9 3.9 1.5 1.3 0.5
0.7 3.4 3.2 2.2 0.1 2.7 1.6 2.7 4.1 2.9 0.7 1.7 2.7 1.8 1.9 3.2 0.5 3.1 4.5 4.9 1.1 3.1 3.1 2.5 3.4
1.0 4.0 3.4 3.1 1.9 0.9 0.3 0.4 4.0 1.5 3.2 1.6 1.8 4.6 2.1 2.3 2.3 2.8 1.0 4.1 2.0 0.3 4.7 0.6 0.1
3.4 4.7 4.0 3.2 2.8 3.9 0.1 3.0 3.5 0.6 4.1 4.4 2.9 3.9 5.0 2.8 2.0 2.7 5.0 3.6 2.9 1.5 2.7 2.1 3.1
1.0 4.3 3.1 4.5 3.7 1.8 2.2 1.2 1.9 4.8 1.3 4.4 0.5 2.5 4.7 4.4 2.2 1.8 0.1 0.4 0.4 4.1 3.2 1.5 4.0
1.4 0.1 1.4 2.5 4.2 3.6 2.3 3.4 1.7 4.7 3.3 4.5 4.6 1.8 2.6 1.1 4.2 1.5 4.9 1.3 0.2 0.1 4.7 2.9 2.0
3.7 1.2 3.0 2.6 4.0 2.1 4.2 0.5 4.0 3.8 4.3 0.5 3.6 3.9 2.4 1.5 1.1 1.2 4.7 2.6 4.5 2.5 4.0 1.8 0.3
```

Observaciones

Podemos observar que las predicciones no varían tanto usando 9 vecinos sin importar la métrica que usemos y aproximando por media ya que al nutrir de más datos (vecinos) la ecuación se vuelve más exacta.

utility-matrix-50-250-1.txt (coseno-simple–40 vecinos)

python3 src/sistema_recomendacion.py data/matrix/utility-matrix-50-250-1.txt --metrica coseno --vecinos 40 --prediccion simple --salida prueba.txt

Obviamente en una imagen no cabe el resultado del programa, pero los cálculos se hacen rápidamente y podemos observar el resultado como cabe esperar. En las siguientes imágenes se mostraran algunas métricas y predicciones

Métricas.

```
Métricas:
Usuario 1 - Usuario 2: Coseno = 0.721
Usuario 1 - Usuario 3: Coseno = 0.749
Usuario 1 - Usuario 4: Coseno = 0.760
Usuario 1 - Usuario 5: Coseno = 0.752
Usuario 1 - Usuario 6: Coseno = 0.743
Usuario 1 - Usuario 7: Coseno = 0.771
Usuario 1 - Usuario 8: Coseno = 0.751
Usuario 1 - Usuario 9: Coseno = 0.750
Usuario 1 - Usuario 10: Coseno = 0.782
Usuario 1 - Usuario 11: Coseno = 0.752
Usuario 1 - Usuario 12: Coseno = 0.760
```

Predicciones.

```
1228    Predicciones:
1229    Usuario 1, Ítem 84: Predicción = 2.5
1230    Usuario 1, Ítem 119: Predicción = 2.5
1231    Usuario 1, Ítem 121: Predicción = 2.4
1232    Usuario 2, Ítem 192: Predicción = 2.4
1233    Usuario 2, Ítem 228: Predicción = 2.4
1234    Usuario 4, Ítem 20: Predicción = 2.5
1235    Usuario 4, Ítem 130: Predicción = 2.9
1236    Usuario 4, Ítem 133: Predicción = 2.5
1237    Usuario 4, Ítem 149: Predicción = 2.3
1238    Usuario 4, Ítem 239: Predicción = 2.7
1239    Usuario 5, Ítem 152: Predicción = 2.3
1240    Usuario 6, Ítem 180: Predicción = 2.2
1241    Usuario 7, Ítem 3: Predicción = 2.5
1242    Usuario 7, Ítem 147: Predicción = 2.4
1243    Usuario 7, Ítem 242: Predicción = 2.9
1244    Usuario 9, Ítem 106: Predicción = 2.7
1245    Usuario 9, Ítem 180: Predicción = 2.3
1246    Usuario 9, Ítem 182: Predicción = 2.8
1247    Usuario 10, Ítem 97: Predicción = 2.8
1248    Usuario 10, Ítem 117: Predicción = 2.4
1249    Usuario 10, Ítem 178: Predicción = 2.6
1250    Usuario 10, Ítem 216: Predicción = 2.4
1251    Usuario 11, Ítem 29: Predicción = 2.5
1252    Usuario 11, Ítem 134: Predicción = 2.7
1253    Usuario 11, Ítem 233: Predicción = 2.3
```

utility-matrix-50-250-1.txt (euclidean-simple–40 vecinos)

python3 src/sistema_recomendacion.py data/matrix/utility-matrix-50-250-1.txt --metrica euclidean --vecinos 40 --prediccion simple --salida prueba.txt

Predicciones.

```
Predicciones:
Usuario 1, Ítem 84: Predicción = 2.4
Usuario 1, Ítem 119: Predicción = 2.5
Usuario 1, Ítem 121: Predicción = 2.4
Usuario 2, Ítem 192: Predicción = 2.3
Usuario 2, Ítem 228: Predicción = 2.3
Usuario 4, Ítem 20: Predicción = 2.5
Usuario 4, Ítem 130: Predicción = 2.8
Usuario 4, Ítem 133: Predicción = 2.5
Usuario 4, Ítem 149: Predicción = 2.4
Usuario 4, Ítem 239: Predicción = 2.7
Usuario 5, Ítem 152: Predicción = 2.4
Usuario 6, Ítem 180: Predicción = 2.3
Usuario 7, Ítem 3: Predicción = 2.5
Usuario 7, Ítem 147: Predicción = 2.4
Usuario 7, Ítem 242: Predicción = 2.9
Usuario 9, Ítem 106: Predicción = 2.7
Usuario 9, Ítem 180: Predicción = 2.2
Usuario 9, Ítem 182: Predicción = 2.8
Usuario 10, Ítem 97: Predicción = 2.9
Usuario 10, Ítem 117: Predicción = 2.3
Usuario 10, Ítem 178: Predicción = 2.5
Usuario 10, Ítem 216: Predicción = 2.4
Usuario 11, Ítem 29: Predicción = 2.6
Usuario 11, Ítem 134: Predicción = 2.7
Usuario 11, Ítem 233: Predicción = 2.5
```

Observaciones

Podemos observar de nuevo que nuestro programa funciona para matrices mucho más grandes correctamente y que las predicciones entre las diferentes métricas son casi idénticas lo cual es lo que debería de pasar si usamos un tamaño de vecinos tan grande incluso si no aproximamos por media.

Conclusiones

Este sistema de recomendación demuestra la efectividad del filtrado colaborativo basado en similitudes entre usuarios. La implementación de métricas como la correlación de Pearson, la similitud coseno y la distancia euclidiana permite captar diferentes aspectos de las relaciones en las calificaciones. Por ejemplo, la correlación de Pearson identifica patrones comunes, la similitud coseno evalúa la orientación de los vectores de calificación, y la distancia euclidiana se centra en las diferencias absolutas.

El desarrollo manual de estas métricas favorece un mejor entendimiento de los algoritmos del filtrado colaborativo, lo cual es valioso para estudiantes y desarrolladores. Sin embargo, en aplicaciones a gran escala, se sugiere usar bibliotecas optimizadas que pueden manejar grandes volúmenes de datos y ofrecer técnicas avanzadas.

Además, la selección del número de vecinos y el tipo de predicción afecta directamente la precisión de las recomendaciones. Por último, es crucial considerar la ética en el manejo de datos de los usuarios y garantizar la privacidad y la transparencia en la generación de recomendaciones. En resumen, este trabajo no solo refuerza el conocimiento técnico sobre sistemas de recomendación, sino que también plantea importantes consideraciones éticas en su implementación.