Universidad Nacional Autónoma de México

Facultad de Ingeniería

PRÁCTICA 10

REGLAS DE ASOCIACIÓN

Minería de Datos

Profesor:

Dr. Molero Castillo Guillermo Gilberto

Grupo 1

Alumna:

Monroy Velázquez Alejandra Sarahí

No. Cuenta: 314000417

OBJETIVO

Obtener reglas de asociación a partir de datos obtenidos de una plataforma de películas, donde los clientes pueden rentar o comprar este tipo de contenidos.

DESARROLLO

El conjunto de datos con el que se trabajará corresponde a registros varios de datos de películas.

Primero comenzamos la importación de bibliotecas correspondientes que nos ayudarán para la realización del código, las cuales son *pandas* para la manipulación y análisis de datos, *numpy* para crear vectores y matrices, *matplotlib* para la generación de gráficas, así como *seaborn* para la visualización de datos. En esta práctica agregamos *apyori* el cual es un paquete que se instala mediante *pip*. Por último, la biblioteca files para subir el archivo *csv*.

Una vez importadas, el *dataframe* se lee y se despliega en pantalla:

```
[1] !pip install apyori  # pip es un administrador de paquetes de Python. Se instala el paquete Apyori
```

```
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5974 sha256=d766066e57b04394
  Stored in directory: /root/.cache/pip/wheels/cb/f6/e1/57973c631d27efd1a2f375bd6a83b2a616c4021f24aa
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

```
[2] import pandas as pd                # Para la manipulación y análisis de los datos
    import numpy as np                 # Para crear vectores y matrices n dimensionales
    import matplotlib.pyplot as plt    # Para la generación de gráficas a partir de los datos
    from apyori import apriori
```

```
[3] from google.colab import files
    files.upload()
```

Elegir archivos movies.csv

- **movies.csv**(application/vnd.ms-excel) - 461797 bytes, last modified: 26/10/2021 - 100% done

```
Saving movies.csv to movies.csv
{'movies.csv': b"The Revenant,13 Hours,Allied,Zootopia,Jigsaw,Achorman,Grinch,Fast and Furious,Ghostbusters,Wolverine,Mad Max,John
```

```
[4] DatosMovies = pd.read_csv('movies.csv')
    DatosMovies
```

	The Revenant	13 Hours	Allied	Zootopia	Jigsaw	Achorman	Grinch	Fast and Furious	Ghostbusters	Wolverine	Mad Max	John Wick	La La Land	The Good Dunosaur
0	Beirut	Martian	Get Out	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Deadpool	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	X-Men	Allied	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	Ninja Turtles	Moana	Ghost in the Shell	Ralph Breaks the Internet	John Wick	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Ahora que el *dataframe* está cargado, comenzamos con los pasos vistos en clase.

En este paso observamos dos cosas muy importantes:

- Que el encabezado es la primera transacción, lo cual no es ideal.
- 'NaN' indica que esa película no fue rentada o comprada en esa transacción.

Por lo que lo siguiente será remover el encabezado para que los datos de las transacciones sean uniformes:

```
[5] DatosMovies = pd.read_csv('movies.csv', header=None)  #para hacer que el primer renglon no sea el nombre de la columna
    DatosMovies.head(6)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	The Revenant	13 Hours	Allied	Zootopia	Jigsaw	Achorman	Grinch	Fast and Furious	Ghostbusters	Wolverine	Mad Max	John Wick	La La Land	The Good Dunosaur	Ninja Turtles	D
1	Beirut	Martian	Get Out	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	Deadpool	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	X-Men	Allied	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	Ninja Turtles	Moana	Ghost in the Shell	Ralph Breaks the	John Wick	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

Procesamiento de los datos

Antes de ejecutar el algoritmo, es recomendable observar la distribución de la frecuencia de los elementos. Crearemos una sola lista, incluyendo todas las transacciones en ella, luego crearemos una matriz usando la lista y

agregaremos una columna extra de nombre “Frecuencia”, esta será llenada con ceros, y luego agruparemos los elementos de modo que podamos ver en orden ascendente las películas y la frecuencia de estas.

```
[11] #Se incluyen todas las transacciones en una sola lista
     Transacciones = DatosMovies.values.reshape(-1).tolist() #-1 significa 'dimensión desconocida'

     #Se crea una matriz (dataframe) usando la lista y se incluye una columna 'Frecuencia'
     Lista = pd.DataFrame(Transacciones)
     Lista['Frecuencia'] = 1

     #Se agrupa los elementos
     Lista = Lista.groupby(by=[0], as_index=False).count().sort_values(by=['Frecuencia'], ascending=True) #Conteo
     Lista['Porcentaje'] = (Lista['Frecuencia'] / Lista['Frecuencia'].sum()) #Porcentaje
     Lista = Lista.rename(columns={0 : 'Item'})

     #Se muestra la lista
     Lista
```

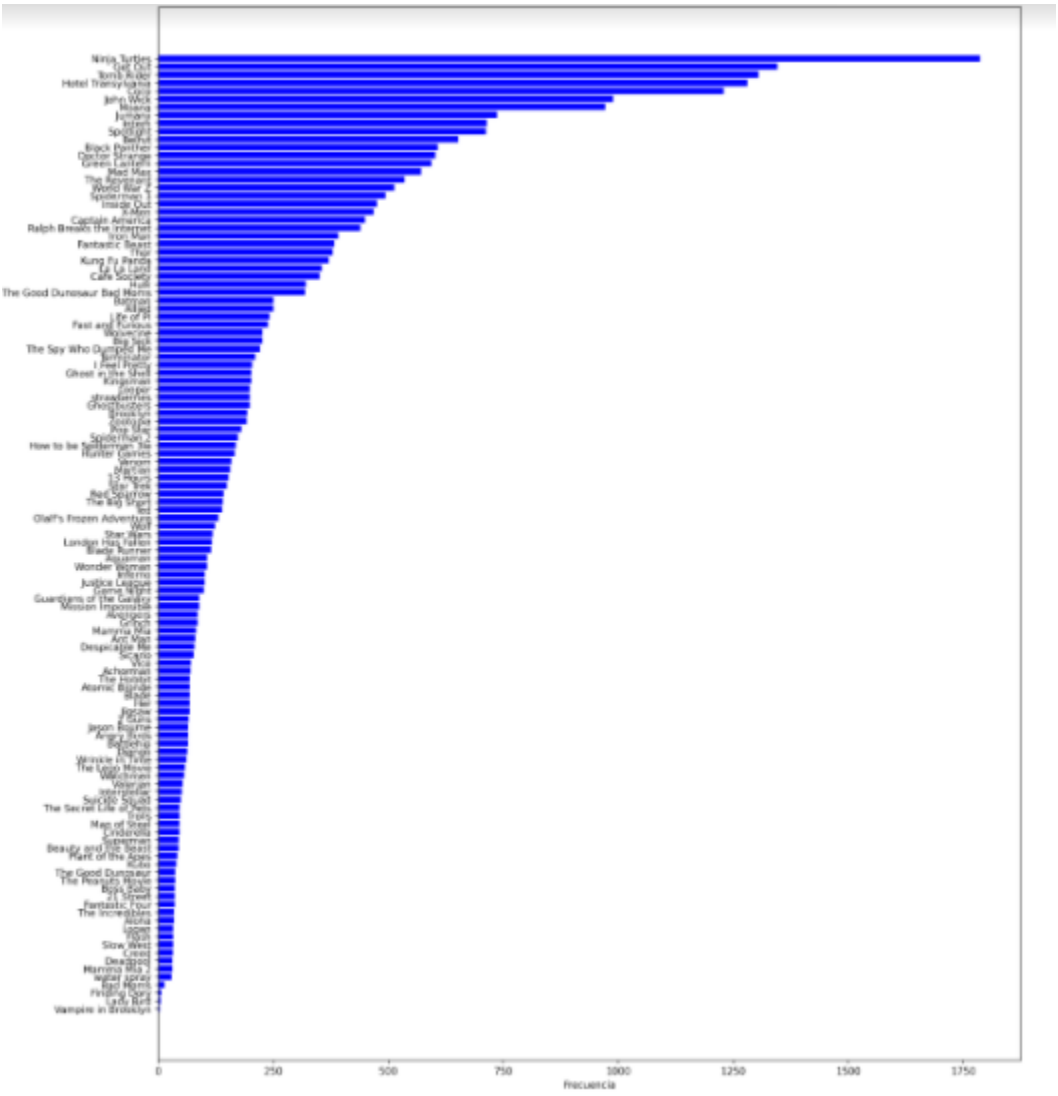
	Item	Frecuencia	Porcentaje
106	Vampire in Brooklyn	3	0.000102
63	Lady Bird	5	0.000171
34	Finding Dory	7	0.000239
11	Bad Moms	14	0.000477
118	water spray	29	0.000989
...	...	...	...
25	Coco	1229	0.041915
44	Hotel Transylvania	1280	0.043655
103	Tomb Rider	1305	0.044507
37	Get Out	1346	0.045906
75	Ninja Turtles	1786	0.060912

119 rows × 3 columns

Como vemos *Vampire in Brooklyn* solo tiene un 3 de frecuencia, lo que quiere decir que solo aparece en tres transacciones, lo que la convierte en la película menos vista de todas, a diferencia de *Ninja Turtles* la cual tiene 1786 de frecuencia, lo que la convierte en la película más vista de todas.

Otra opción para observar las frecuencias de las películas es generar un gráfico de barras con base a la lista anterior:

```
[12] # Se genera un gráfico de barras
     plt.figure(figsize=(16,20), dpi=300)
     plt.ylabel('Item')
     plt.xlabel('Frecuencia')
     plt.barh(ListaM['Item'], width=Lista['Frecuencia'], color='blue')
     plt.show()
```



Dado que los datos actuales están en un *dataframe* de Pandas, los convertiremos a una lista ya que la función de Apriori de Python requiere que el conjunto de datos tenga la forma de una lista de listas, donde cada transacción es una lista interna dentro de una gran lista.

```
[13] MoviesLista = DatosMovies.stack().groupby(level=0).apply(list).tolist()
     MoviesLista
```

```
 'Fast and Furious',
 'Spotlight'],
['Jumanji', 'Ninja Turtles', 'Moana', 'Get Out', "Olalf's Frozen Adventure"],
['The Revenant', 'Avengers', 'John Wick'],
['Inside Out'],
['Kung Fu Panda', 'Intern', 'Ninja Turtles', 'Pop Star', 'Brooklyn'],
['X-Men',
 'World War Z',
 'Tomb Rider',
 'Moana',
 'Despicable Me',
 'Get Out',
 'La La Land',
 'Black Panther',
 'John Wick',
 'Hotel Transylvania',
 'Batman',
 'Wolverine'],
['Kung Fu Panda',
 'Terminator',
 'Tomb Rider',
 'Get Out',
 'Ralph Breaks the Internet',
```

Aplicación del algoritmo Apriori

Configuración 1

Obtenemos las reglas para aquellas películas que se hayan rentado al menos 10 veces en un día (70 veces en la semana):

- El soporte mínimo se calcula de 70/7460 = 0.00938 (1%).
- La confianza mínima para las reglas de 30%
- La elevación de 2.

```
[14] ReglasC1 = apriori(MoviesLista,
                        min_support=0.01,
                        min_confidence=0.3,
                        min_lift=2)
```

Convertimos las reglas a una lista para una mejor observación de estas:

```
[15] ResultadosC1 = list(ReglasC1)
     print(len(ResultadosC1)) #Total de reglas encontradas
```

```
9
```

```
[16] ResultadosC1
```

```
[RelationRecord(items=frozenset({'Kung Fu Panda', 'Jumanji'}), support=0.0160857908847185, ordered_statistics=[OrderedStatistic(items
 RelationRecord(items=frozenset({'Tomb Rider', 'Jumanji'}), support=0.03941018766756032, ordered_statistics=[OrderedStatistic(items_b
 RelationRecord(items=frozenset({'Thor', 'Moana'}), support=0.015281501340482574, ordered_statistics=[OrderedStatistic(items_base=fro
 RelationRecord(items=frozenset({'Tomb Rider', 'Terminator'}), support=0.01032171581769437, ordered_statistics=[OrderedStatistic(item
 RelationRecord(items=frozenset({'Get Out', 'Ninja Turtles', 'Jumanji'}), support=0.010187667560321715, ordered_statistics=[OrderedSt
 RelationRecord(items=frozenset({'Intern', 'Moana', 'Ninja Turtles'}), support=0.011126005361930294, ordered_statistics=[OrderedStati
 RelationRecord(items=frozenset({'Moana', 'Ninja Turtles', 'Jumanji'}), support=0.011126005361930294, ordered_statistics=[OrderedStat
 RelationRecord(items=frozenset({'Tomb Rider', 'Ninja Turtles', 'Jumanji'}), support=0.017158176943699734, ordered_statistics=[Ordere
 RelationRecord(items=frozenset({'Tomb Rider', 'Spiderman 3', 'Ninja Turtles'}), support=0.01032171581769437, ordered_statistics=[Ord
```

En total se encuentran nueve reglas:

```
[17] pd.DataFrame(ResultadosC1)
```

	items	support	ordered_statistics
0	(Kung Fu Panda, Jumanji)	0.016086	[((Kung Fu Panda), (Jumanji), 0.32345013477088...
1	(Tomb Rider, Jumanji)	0.039410	[((Jumanji), (Tomb Rider), 0.3994565217391304,...
2	(Thor, Moana)	0.015282	[((Thor), (Moana), 0.3007915567282322, 2.31092...
3	(Tomb Rider, Terminator)	0.010322	[((Terminator), (Tomb Rider), 0.36492890995260...
4	(Get Out, Ninja Turtles, Jumanji)	0.010188	[((Get Out, Jumanji), (Ninja Turtles), 0.50666...
5	(Intern, Moana, Ninja Turtles)	0.011126	[((Intern, Ninja Turtles), (Moana), 0.30970149...
6	(Moana, Ninja Turtles, Jumanji)	0.011126	[((Moana, Jumanji), (Ninja Turtles), 0.5030303...
7	(Tomb Rider, Ninja Turtles, Jumanji)	0.017158	[((Ninja Turtles, Jumanji), (Tomb Rider), 0.41...
8	(Tomb Rider, Spiderman 3, Ninja Turtles)	0.010322	[((Spiderman 3, Ninja Turtles), (Tomb Rider), ...

Podemos imprimir regla por regla para tener mayor detalle de cada una y poder analizarlas por separado, por ejemplo, imprimiremos la primer regla:

```
[18] print(ResultadosC1[0])

RelationRecord(items=frozenset({'Kung Fu Panda', 'Jumanji'}), support=0.0160857908847185, ordered_statistics=[OrderedStatistic(items_base=frozenset({'Kung Fu
```

La primera regla contiene dos elementos: 'Kung Fu Panda' y 'Jumanji' que se vieron juntos. Esto tiene sentido, ya que las personas que ven películas familiares, en este caso de corte infantil, suelen ver también más películas del mismo tipo, como Kung Fu Panda (2016) y Jumanji (2017).

El soporte es de 0.016 (1.6%), la confianza de 0.32 (32%) y elevación de 3.27, esto representa que existe 3 veces más posibilidades de que los que vean Kung Fu Panda miren también Jumanji, o viceversa.

A pesar de que imprimimos una sola regla, la observación de esta es un poco difícil, por lo que podemos generar código que nos ayude a una mejor visualización de todas las reglas:

```
[20] for item in ResultadosC1:
       #El primer índice de la lista
       Emparejar = item[0]
       items = [x for x in Emparejar]
       print("Regla: " + str(item[0]))

       #El segundo índice de la lista
       print("Soporte: " + str(item[1]))

       #El tercer índice de la lista
       print("Confianza: " + str(item[2][0][2]))
       print("Lift: " + str(item[2][0][3]))
       print("====================================")
```

```
Regla: frozenset({'Kung Fu Panda', 'Jumanji'})
Soporte: 0.0160857908847185
Confianza: 0.3234501347708895
Lift: 3.2784483768897226
====================================
Regla: frozenset({'Tomb Rider', 'Jumanji'})
Soporte: 0.03941018766756032
Confianza: 0.3994565217391304
Lift: 2.283483258370814
====================================
Regla: frozenset({'Thor', 'Moana'})
Soporte: 0.015281501340482574
Confianza: 0.3007915567282322
Lift: 2.3109217437617016
====================================
Regla: frozenset({'Tomb Rider', 'Terminator'})
Soporte: 0.01032171581769437
Confianza: 0.36492890995260663
Lift: 2.0861070254762035
====================================
Regla: frozenset({'Get Out', 'Ninja Turtles', 'Jumanji'})
Soporte: 0.010187667560321715
Confianza: 0.5066666666666666
Lift: 2.1163120567375886
====================================
Regla: frozenset({'Intern', 'Moana', 'Ninja Turtles'})
Soporte: 0.011126005361930294
Confianza: 0.30970149253731344
Lift: 2.37937500960696
====================================
```

```
Regla: frozenset({'Moana', 'Ninja Turtles', 'Jumanji'})
Soporte: 0.011126005361930294
Confianza: 0.5030303030303029
Lift: 2.1011232142251175
====================================
Regla: frozenset({'Tomb Rider', 'Ninja Turtles', 'Jumanji'})
Soporte: 0.017158176943699734
Confianza: 0.4169381107491857
Lift: 2.383416326581552
====================================
Regla: frozenset({'Tomb Rider', 'Spiderman 3', 'Ninja Turtles'})
Soporte: 0.01032171581769437
Confianza: 0.3719806763285024
Lift: 2.1264182723453087
====================================
```

Configuración 2

Obtenemos las reglas para aquellas películas que se hayan visto al menos 210 veces a la semana (30 por día):

- El soporte mínimo se calcula de 210/7460 = 0.028 (2.8%).
- La confianza mínima para las reglas de 30%.
- La elevación mayor a 1.

```
[21] ReglasC2 = apriori(MoviesLista,
                        min_support=0.028,
                        min_confidence=0.3,
                        min_lift = 1.01)
```

Convertimos las reglas a una lista para una mejor observación de estas:

```
[22] ResultadosC2 = list(ReglasC2)
     print(len(ResultadosC2))
```

```
8
```

```
[23] ResultadosC2
```

```
[RelationRecord(items=frozenset({'Beirut', 'Get Out'}), support=0.028954423592493297, ordered_statistics=[OrderedStatistic(item
 RelationRecord(items=frozenset({'Ninja Turtles', 'Coco'}), support=0.05294906166219839, ordered_statistics=[OrderedStatistic(i
 RelationRecord(items=frozenset({'Intern', 'Ninja Turtles'}), support=0.035924932975871314, ordered_statistics=[OrderedStatisti
 RelationRecord(items=frozenset({'Ninja Turtles', 'Jumanji'}), support=0.04115281501340483, ordered_statistics=[OrderedStatisti
 RelationRecord(items=frozenset({'Tomb Rider', 'Jumanji'}), support=0.03941018766756032, ordered_statistics=[OrderedStatistic(i
 RelationRecord(items=frozenset({'Moana', 'Ninja Turtles'}), support=0.04825737265415549, ordered_statistics=[OrderedStatistic(
 RelationRecord(items=frozenset({'Ninja Turtles', 'Spotlight'}), support=0.0339142091152815, ordered_statistics=[OrderedStatist
 RelationRecord(items=frozenset({'Tomb Rider', 'Ninja Turtles'}), support=0.060053619302949064, ordered_statistics=[OrderedStat
```

En total se encuentran ocho reglas:

```
[24] pd.DataFrame(ResultadosC2)
```

	items	support	ordered_statistics
0	(Beirut, Get Out)	0.028954	[((Beirut), (Get Out), 0.3312883435582822, 1.8...
1	(Ninja Turtles, Coco)	0.052949	[((Coco), (Ninja Turtles), 0.32166123778501626...
2	(Intern, Ninja Turtles)	0.035925	[((Intern), (Ninja Turtles), 0.375350140056022...
3	(Ninja Turtles, Jumanji)	0.041153	[((Jumanji), (Ninja Turtles), 0.41711956521739...
4	(Tomb Rider, Jumanji)	0.039410	[((Jumanji), (Tomb Rider), 0.3994565217391304,...
5	(Moana, Ninja Turtles)	0.048257	[((Moana), (Ninja Turtles), 0.3707518022657054...
6	(Ninja Turtles, Spotlight)	0.033914	[((Spotlight), (Ninja Turtles), 0.355337078651...
7	(Tomb Rider, Ninja Turtles)	0.060054	[((Tomb Rider), (Ninja Turtles), 0.34329501915...

Podemos imprimir regla por regla para tener mayor detalle de cada una y poder analizarlas por separado, por ejemplo, imprimiremos la primer regla:

```
[25] print(ResultadosC2[0])

     RelationRecord(items=frozenset({'Beirut', 'Get Out'}), support=0.028954423592493297, ordered_statistics=[Ordered
```

La primera regla contiene dos elementos: 'Get Out' y 'Beirut' que se vieron juntos. Esto también tiene sentido, las personas que ven películas de espionaje, como Beirut (2018), tienen gustos afines con películas de terror, como Get Out (2017).

El soporte es de 0.028 (2.8%), la confianza de 0.33 (33%) y una elevación de 1.83, esto representa que existe casi 2 veces más probabilidades de que los que vean Get Out miren también Beirut, o viceversa.

A pesar de que imprimimos una sola regla, la observación de esta es un poco difícil, por lo que, como en la configuración anterior, podemos generar código que nos ayude a una mejor visualización de todas las reglas:

```
[27] for item in ResultadosC2:
       #El primer índice de la lista
       Emparejar = item[0]
       items = [x for x in Emparejar]
       print("Regla: " + str(item[0]))

       #El segundo índice de la lista
       print("Soporte: " + str(item[1]))

       #El tercer índice de la lista
       print("Confianza: " + str(item[2][0][2]))
       print("Lift: " + str(item[2][0][3]))
       print("====================================")
```

```
Regla: frozenset({'Beirut', 'Get Out'})
Soporte: 0.028954423592493297
Confianza: 0.3312883435582822
Lift: 1.8361151879233173
====================================
Regla: frozenset({'Ninja Turtles', 'Coco'})
Soporte: 0.05294906166219839
Confianza: 0.32166123778501626
Lift: 1.3435570178478284
====================================
Regla: frozenset({'Intern', 'Ninja Turtles'})
Soporte: 0.035924932975871314
Confianza: 0.3753501400560224
Lift: 1.5678118951948081
====================================
Regla: frozenset({'Ninja Turtles', 'Jumanji'})
Soporte: 0.04115281501340483
Confianza: 0.4171195652173913
Lift: 1.742279930863236
====================================
Regla: frozenset({'Tomb Rider', 'Jumanji'})
Soporte: 0.03941018766756032
Confianza: 0.3994565217391304
Lift: 2.283483258370814
====================================
Regla: frozenset({'Moana', 'Ninja Turtles'})
Soporte: 0.04825737265415549
Confianza: 0.3707518022657054
Lift: 1.5486049523528347
====================================
Regla: frozenset({'Ninja Turtles', 'Spotlight'})
Soporte: 0.0339142091152815
Confianza: 0.3553370786516854
Lift: 1.4842187047825157
====================================
Regla: frozenset({'Tomb Rider', 'Ninja Turtles'})
Soporte: 0.060053619302949064
Confianza: 0.3432950191570881
Lift: 1.4339198448554744
====================================
```

CONCLUSIÓN

En esta sesión practica reforzamos los conocimientos acerca de las reglas de asociación aplicando el algoritmo *apriori* sobre el conjunto de datos transaccionales de películas, lo cual nos dio un panorama acerca de cómo se podría utilizar esta herramienta en el mundo actual, por ejemplo, al encontrar películas asociadas, podemos poner estas películas juntas dentro del sistema donde se están rentando, lo cual sería beneficioso para la empresa o negocio, ya que aumentaría las vistas, y/o el dinero sobre esa transacción, además seria también beneficioso para el cliente ya que sería mucho más cómodo ver las películas asociadas a la que acaba de ver o de comprar. Esto pasa muy seguido en plataformas de streaming, como en Netflix, cuando terminamos de ver una película o serie, la propia plataforma te indica que podrías seguir viendo otra película o serie similar.