Universidad Nacional
Autónoma de México

Inteligencia Artificial

Reglas de Asociación (Apriori)

Grupo 3

Nombre:
Barreiro Valdez Alejandro

Práctica 3

Profesor: Dr. Guillermo Gilberto Molero Castillo

10 de marzo de 2022

Introducción

En esta práctica se generarán reglas a partir del algoritmo Apriori para generar una base de un sistema de recomendación. En este modelo se analizarán transacciones de productos de comida vendidos en un comercio en un periodo de una semana. A partir de estos datos se generará una frecuencia de cúantas veces se compran y se creará una lista que será analizada por el algoritmo. Se generarán tres configuraciones y se analizará cada una de estas configuraciones.

Objetivo

Analizar las transacciones y obtener reglas significativas (patrones) de los productos vendidos en un comercio minorista en Francia.

Los datos son transacciones de un comercio de un periodo de una semana (7 días).

Desarrollo

Se instalan y cargan los diferentes módulos a utilizar en esta práctica. Se importaron los módulos de apyori, pandas, numpy y matplotlib.

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

```
Requirement already satisfied: apyori in /opt/anaconda3/lib/python3.7/site-packages (1.1.2)
```

```
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
```

Se importan los datos de las transacciones utilizando pandas y la función de lectura de csv. Se tienen 7500 transacciones de 20 productos.

```
DatosTransacciones = pd.read_csv('store_data.csv')
DatosTransacciones
```

	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydar juic
0	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
1	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
2	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
3	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
4	low fat yogurt	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
7495	butter	light mayo	fresh bread	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
7496	burgers	frozen vegetables	eggs	french fries	magazines	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
7497	chicken	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
7498	escalope	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
7499	eggs	frozen smoothie	yogurt cake	low fat yogurt	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

El encabezado tiene de encabezado a la primera transacción que se encuentra en los datos. Todos los NaN indican que esa transacción no fue realizada. Se agrega un encabezado a la tabla para manejarla de mejor manera.

```
DatosTransacciones = pd.read_csv('store_data.csv', header=None)
DatosTransacciones.head(5)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydant juice	frozen smoothie	sp
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

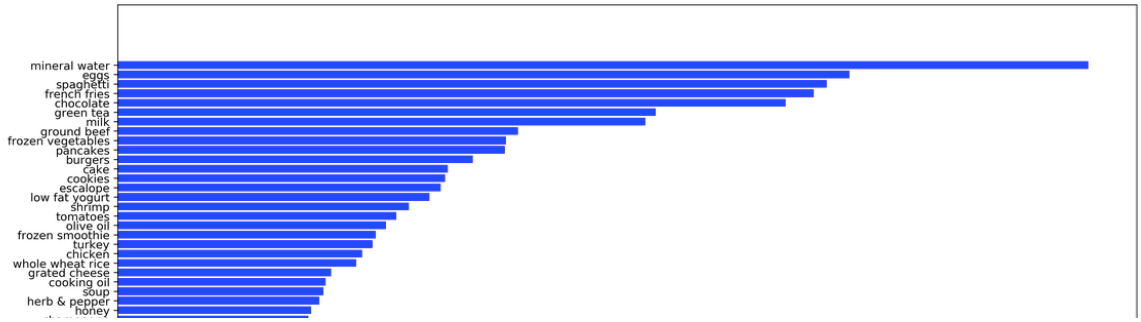
Para el siguiente procedimiento se realiza uno análogo a lo que se realizó la práctica pasada. Se cambia esta tabla a una lista de una dimensión, se le agregan dos columnas de frecuencia y de porcentaje de aparición. Para la columna de frecuencia se contó las veces que aparece cada dato y se calculó la frecuencia utilizando la cuenta total. Por último se le asignó nombre a la columna item.

```
#Se incluyen todas las transacciones en una sola lista
Transacciones = DatosTransacciones.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
0	asparagus	1	0.000034
112	water spray	3	0.000102
77	napkins	5	0.000170
34	cream	7	0.000238
11	bramble	14	0.000477
...	...	...	...
25	chocolate	1230	0.041889
43	french fries	1282	0.043660

Para el siguiente punto se generó un gráfico de barras utilizando matplotlib para poder visualizar la frecuencia de cada item que se encuentra en los datos.

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



A partir del Data Frame se crea una lista de listas y se eliminan todos los datos que son NaN.

```
#Se crea una lista de listas a partir del dataframe y se remueven los 'NaN'
#level=0 especifica desde el primer índice
TransaccionesLista = DatosTransacciones.stack().groupby(level=0).apply(list).tolist()
TransaccionesLista
```

```
[['shrimp',
  'almonds',
  'avocado',
  'vegetables mix',
  'green grapes',
  'whole weat flour',
  'yams',
  'cottage cheese',
  'energy drink',
  'tomato juice',
  'low fat yogurt',
  'green tea',
  'honey',
  'salad',
  'mineral water',
  'salmon',
  'antioxydant juice',
  'frozen smoothie'
```

Se aplica el algoritmo de la misma manera en que se aplicó en la práctica pasada. Se utilizan dos configuraciones. En la primera configuración se generan reglas para artículos que se compran al menos 5 veces al día. El soporte mínimo calculado es de .45%, con una confianza mínima del 20% y una elevación de 3. Se generan las reglas y se imprime el número de reglas que se generaron. También se imprimen algunas reglas para ver su formato.

```
ReglasC1 = apriori(TransaccionesLista,
                   min_support=0.0045,
                   min_confidence=0.2,
                   min_lift=3)
```

Se convierte las reglas encontradas por la clase apriori en una lista, puesto que es más fácil ver los resultados.

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

```
24
```

```
ResultadosC1
```

```
[RelationRecord(items=frozenset({'light cream', 'chicken'}), support=0.004532728969470737, ordered_statistics=[Orde
redStatistic(items_base=frozenset({'light cream'}), items_add=frozenset({'chicken'}), confidence=0.2905982905982905
7, lift=4.84395061728395)]),
 RelationRecord(items=frozenset({'mushroom cream sauce', 'escalope'}), support=0.005732568990801226, ordered_statis
tics=[OrderedStatistic(items_base=frozenset({'mushroom cream sauce'}), items_add=frozenset({'escalope'}), confidenc
e=0.3006993006993007, lift=3.790832696715049)]),
 RelationRecord(items=frozenset({'pasta', 'escalope'}), support=0.005865884548726837, ordered_statistics=[OrderedSt
atistic(items_base=frozenset({'pasta'}), items_add=frozenset({'escalope'}), confidence=0.3728813559322034, lift=4.7
00811850163794)]),
```

Se generaron 24 reglas que servirán para el sistema de recomendación. La primera regla contiene el elemento de pollo y crema ligera. Esta regla sí tiene sentido ya que estos dos productos tienen sentido que se compren juntos. Aún así, dentro de las reglas existen algunas que no tienen tanto sentido. Se tiene una regla que relaciones vegetales congelados, camarón y chocolate. Este tipo de reglas nos permite darnos cuenta sobre las limitaciones de este algoritmo y como no todas las reglas son perfectas. Para mejorar esto se podría reducir las transacciones pocos frecuentes o

categorizar los productos que se tienen. Como en la práctica pasada se imprimieron las 24 reglas con sus principales características numéricas. Se muestran algunas reglas que fueron impresas.

```
Regla: frozenset({'light cream', 'chicken'})
Soporte: 0.004532728969470737
Confianza: 0.29059829059829057
Lift: 4.84395061728395
===================================
Regla: frozenset({'mushroom cream sauce', 'escalope'})
Soporte: 0.005732568990801226
Confianza: 0.3006993006993007
Lift: 3.790832696715049
===================================
Regla: frozenset({'pasta', 'escalope'})
Soporte: 0.005865884548726837
Confianza: 0.3728813559322034
Lift: 4.700811850163794
===================================
Regla: frozenset({'ground beef', 'herb & pepper'})
Soporte: 0.015997866951073192
Confianza: 0.3234501347708895
Lift: 3.2919938411349285
```

Para la segunda configuración se obtuvieron reglas para aquellos productos que se compran mínimo 30 veces al día. Se tiene un valor de soporte calculado de 2.8%, confianza mínima de 25% y elevación de 1. Se realiza la configuración y las reglas utilizando el algoritmo que fue importado del módulo y se imprimió el número de reglas y dichas reglas.

```
ReglasC2 = apriori(TransaccionesLista,
                   min_support=0.028,
                   min_confidence=0.25,
                   min_lift = 1.01)
```

```
ResultadosC2 = list(ReglasC2)
print(len(ResultadosC2))
```

```
10
```

```
ResultadosC2
```

```
[RelationRecord(items=frozenset({'burgers', 'eggs'}), support=0.02879616051193174, ordered_statistics=[OrderedStatistic(items_base=frozenset({'burgers'}), items_add=frozenset({'eggs'}), confidence=0.33027522935779813, lift=1.8378297443715457)]),
 RelationRecord(items=frozenset({'chocolate', 'mineral water'}), support=0.05265964538061592, ordered_statistics=[OrderedStatistic(items_base=frozenset({'chocolate'}), items_add=frozenset({'mineral water'}), confidence=0.3213995117982099, lift=1.3483320682317521)]),
```

La primera regla de estas diez reglas relaciona hambuerguesa y huevo. Esta regla tiene cierto sentido ya que estos productos sí llegan a ser comprados juntos. En estas reglas se puede observar el mismo problema que en la anterior configuración, existen reglas que no tienen mucho sentido. Estas reglas pueden no tener sentido porque se relacionan productos que son muy comprados con otros con los que no tienen relación. Para arreglar esto se podrían quitar aquellos productos que se repiten mucho como el agua mineral. Se imprimen todas las reglas generadas por la configuración.

```
Regla: frozenset({'burgers', 'eggs'})
Soporte: 0.02879616051193174
Confianza: 0.33027522935779813
Lift: 1.8378297443715457
====================================
Regla: frozenset({'chocolate', 'mineral water'})
Soporte: 0.05265964538061592
Confianza: 0.3213995117982099
Lift: 1.3483320682317521
====================================
Regla: frozenset({'mineral water', 'eggs'})
Soporte: 0.05092654312758299
Confianza: 0.28338278931750743
Lift: 1.188844688294532
====================================
Regla: frozenset({'frozen vegetables', 'mineral water'})
Soporte: 0.03572856952406346
Confianza: 0.37482517482517486
Lift: 1.57246288387228
====================================
Regla: frozenset({'ground beef', 'mineral water'})
Soporte: 0.040927876283162246
Confianza: 0.41655359565807326
Lift: 1.7475215442008991
====================================
Regla: frozenset({'ground beef', 'spaghetti'})
Soporte: 0.03919477403012932
Confianza: 0.3989145183175034
Lift: 2.291162176033379
------------------------------------
```

Se creó una tercera configuración donde se eligió una confianza alta y un soporte similar al anterior para observar cómo afecta la confianza y si esto genera reglas más realistas. Se eligió un soporte de 2.6%, una confianza mínima de 40% y una elevación de uno. Para esta configuración solo se generaron dos reglas. Esto indica que aumentar la confianza genera menos reglas ya que no todas las reglas que se generen tienen una confianza tan alta. Ambas reglas relacionan agua mineral con otros dos productos. Con esto se puede concluir que la confianza solo es un parámetro matemático y que no necesariamente va relacionado con lo realista del modelo. Con ambas reglas generadas se tendría un mal sistema de recomendación ya que se relaciona agua mineral con carne molida y aceite de oliva. Tiene sentido que estas reglas se generen porque agua mineral es el producto más vendido de los datos. Se muestran las reglas generadas.

```
Regla: frozenset({'mineral water', 'ground beef'})
Soporte: 0.040927876283162246
Confianza: 0.41655359565807326
Lift: 1.7475215442008991
====================================
Regla: frozenset({'mineral water', 'olive oil'})
Soporte: 0.027596320490601255
Confianza: 0.4190283400809717
Lift: 1.7579035676439423
====================================
```

Conclusión

En esta práctica se generaron reglas para un sistema de recomendación y se reforzaron todos los conceptos sobre el algoritmo vistos en la práctica anterior. Además, se practicó cómo cambiar el formato de los datos que se utilizan para poder manipularlos. Con estas dos prácticas se tiene una buena noción de cómo utilizar los algoritmos de Machine Learning. La primera parte consiste en limpiar los datos que se tienen y ponerlos en un formato en el cual se puedan utilizar. Después, se genera una vista de dichos datos para poder interprtarlos de manera fácil a través de un gráfico. Posteriormente se utiliza el algoritmo en cuestión y se definen los diferentes parámetros que necesita el algoritmo. Para esto se necesitó una noción de lo que hace cada uno de los parámetros. Por último, se interpretan los resultados que arroja el algoritmo. A partir de la lista que se generó se interpreta qué significa cada una de las reglas generadas. Con estas reglas se pueden generar un sistema de recomendación. Gracias a esta práctica se reforzaron los conceptos del algoritmo Apriori y se pudo aplicar este algoritmo a un caso de la vida real.