
**Facultad de Ingeniería**
Ingeniería Civil Informática

Informe Proyecto Ciencia de Datos
Cuarta Entrega

# Retail Online

Autores:
Marcelo Guzmán Escobar
Sebastián Herrera Liberona
Pedro Hurtado Lagos
Cristian Meza Cesped
Diego Muñoz Cabrera

Profesor:
Billy Peralta Márquez.

15 de Julio del 2020

July 16, 2020

# Contents

# 1 Introducción

## Definición de proyecto

Recordando los hitos anteriores del proyecto, se tenía como objetivo principal encontrar los patrones de compra dentro de un conjunto de datos de ventas online de retail y las distintas relaciones entre los datos de esta. En donde se tenía la siguiente distibución dentro del dataset como se puede apreciar en la siguiente tabla:

| | |
|---|---|
| invoiceNo | Número de transacción. Si comienza con una C, significa que es una transacción cancelada. |
| StockCode | Código de identificación para cada producto. |
| Description | Nombre del producto. |
| Quantity | Cantidad del producto por transacción. |
| InvoiceDate | Día y hora en la que la transacción fue efectuada. |
| UnitPrice | Precio por unidad del producto. |
| CustomerID | Código de identificación del cliente. |
| Country | País donde reside el cliente. |

Table 1: Distribución dataset

Como primer paso se realizaron procesos de análisis y preprocesamientos entre los cuales se incluyen: Limpieza de datos/Reducción de datos y eliminación de columnas irrelevantes. Luego se hicieron análisis 1D y 2D utilizando las herramientas como los mapas de calor y gráficos para la distribución de datos.
También se crearon nuevas columnas para un mejor desarrollo de los experimentos, quedando el dataset de la siguiente forma:

| | |
|---|---|
| invoiceNo | Número de transacción. Si comienza con una C, significa que es una transacción cancelada. |
| StockCode | Código de identificación para cada producto. |
| Description | Nombre del producto. |
| Quantity | Cantidad del producto por transacción. |
| UnitPrice | Precio por unidad del producto. |
| CustomerID | Código de identificación del cliente. |
| Country | País donde reside el cliente. |
| Fecha | Día en que la transacción fue efectuada. |
| Hora | Hora en que la transacción fue efectuada. |
| Monto | valor asigando en libras a cada compra total, obtenido al multiplicar las columnas UnitPrice y Quantity. |

Table 2: Distribución dataset final

En la segunda fase, los experimentos realizados fueron:

- Encontrar las fechas con mayores ganancias
- Recomendaciones de productos
- Encontrar el promedio de compra de cada transacción en productos
- Encontrar el promedio de Libras gastadas en transacciones
- Dividir grupos de clientes según dinero gastado(premium y normal)
- Horarios de mayor fluctuación de datos (mayor cantidad de compras)
- Intervalos en hora de menor y mayor numero de transacciones.

Además, dentro de este hito se implementó el algoritmo Apriori y se realizó una revisión de trabajos existentes que sirvieron de guía para el proyecto.

## 2 Correcciones

Con respecto a la entrega del hito 3 del presente proyecto, se presentaron una serie de errrores con respecto a lo solicitado, las cuales fueron distintas tareas asignadas que no se cumplieron en un 100%. Las tareas contemplaban lo siguiente:

- Aplicar A-priori, considerando hallar diferentes soportes y confianzas considerando al menos 5*5 combinaciones, midiendo tiempo
- Considerar A-priori usando K sets random, usando estrategia particional. Considerar de K=2 a 10. Medir tiempo. Medir coincidencia con resultados a A priori original considerando 500, 1000, 2000 y 5000 itemsets.
- Lo mismo que punto anterior pero usando Kmeans de K=2 a 10. Medir tiempo (incluyendo tiempo de K-Means)
- Lo mismo que punto anterior pero usando algoritmo ROCK (opcional)
- En Medición de coincidencia de itemsets frecuentes usar promedio de Jackard index de media de los top 500,1000 y 2000 itemsets.

De las tareas asignadas no se logró el resultado esperado por parte del profesor, presentando los siguientes errores:

- En primer punto deben probar de forma sistemática. El rango deben uds proponerlo. La idea es que muestren una tabla midiendo el tiempo.
- En versión particional tienen que adaptar soporte naturalmente debe ser menor que el original.
- No es prioritario análisis por particion. Enfocarse en proceso.
- No se entiende que devolvieron en parte de K-Means.
- Análisis final de K-Means no es necesario.

Por último, se propusieron las siguientes correcciones al proyecto con el fin de obtener correctos resultados y un aporte en el aprendizaje relacionada a la parte práctica del curso:

- 1. Aplicar Apriori en 5*5 combinaciones. Devolver tabla de 5x5. Elegir la confianza-soporte más interesante a su criterio, usar el soporte mas interesante.
- 2. Aplicar enfoque particional usando K sets random con K=2,4,6,8,10. Para definir particiones use centroides como puntos random y luego asocie cada punto al centro mas cercano

- 3. Medir tiempo y número de reglas encontradas. Obtener porcentaje respecto a número de reglas con A-Priori general. Guardar tabla con ambos datos.
- 4. Repetir 2 y 3 pero usando centroides dados por K-Means.
- 5. Analizar y comparar resultados de tiempos y número de reglas de random vs Kmeans.

# 3 Ejecución de correcciones

Para el tercer hito de este proyecto se utilizarán algoritmos sugeridos por el profesor Billy Peralta. En primera instancia, se sugieren variaciones para la ejecución del algoritmo Apriori, considerando distintas combinaciones de confianza y soporte, llegando a un total de 25 distintas ejecuciones.

Luego, seleccionar una de estas combinaciones para realizar un enfoque particional y poder realizar desde 2 particiones hasta un máximo de 10, todo esto junto a sus tiempos de ejecución y cantidad de reglas obtenidas, comparandolas con las reglas del Apriori general.

Por útlimo, se aplica el algoritmo K-means para lograr distintos grupos de asociaciones, estás también con particiones (en este caso K) desde 2 hasta 10. Los ejercicios presentado más adelante se realizaron con el fin de obtener asociaciones favorables para el estudio del dataset, al igual que en el ejercicio anterior se agregan los tiempos de ejecución y reglas obtenidas, junto a la comparación correspondiente.

## 3.1 Combinaciones Soporte-Confianza Apriori

Como error del hito 3, se indicó que las combinaciones planteadas no seguían alguna relación, por lo cual eran rangos no equidistantes, así que se propuso trabajar con rangos de soporte-confianza que siguieran los mismos patrones, llegando así a la definición de soportes cada 0.004 y confianzas cada 0.1.

Considerando las restricciones mencionadas anteriormente se escogieron 5 confianzas diferentes y para cada confianza, 5 soportes diferentes haciendo un total de 25 experimentos.
Las confianzas escogidas fueron las siguientes:

- Soporte: 0.01 con confianzas de 0.1 hasta 0.5
- Soporte: 0.014 con confianzas de 0.1 hasta 0.5
- Soporte: 0.018 con confianzas de 0.1 hasta 0.5
- Soporte: 0.022 con confianzas de 0.1 hasta 0.5
- Soporte: 0.026 con confianzas de 0.1 hasta 0.5

Además de esto, se evaluó el tiempo de ejecución en segundos de cada combinación y la cantidad de reglas de asociación que entregaba cada una, lo cual queda representado de manera gráfica en la siguiente imagen.

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | Posición | Soporte | Confianza | Tiempo | Reglas |
| 1 | 0 | 0.01 | 0.1 | 129.447 | 678 |
| 2 | 0 | 0.01 | 0.2 | 132.333 | 678 |
| 3 | 0 | 0.01 | 0.3 | 130.936 | 630 |
| 4 | 0 | 0.01 | 0.4 | 131.307 | 494 |
| 5 | 0 | 0.01 | 0.5 | 131.74 | 336 |
| 6 | 1 | 0.014 | 0.1 | 22.5039 | 226 |
| 7 | 1 | 0.014 | 0.2 | 22.5208 | 226 |
| 8 | 1 | 0.014 | 0.3 | 22.2635 | 224 |
| 9 | 1 | 0.014 | 0.4 | 22.3922 | 182 |
| 10 | 1 | 0.014 | 0.5 | 22.1988 | 122 |
| 11 | 2 | 0.018 | 0.1 | 7.03717 | 94 |
| 12 | 2 | 0.018 | 0.2 | 7.01126 | 94 |
| 13 | 2 | 0.018 | 0.3 | 6.94843 | 94 |
| 14 | 2 | 0.018 | 0.4 | 6.96653 | 86 |
| 15 | 2 | 0.018 | 0.5 | 7.00328 | 50 |
| 16 | 3 | 0.022 | 0.1 | 3.90855 | 50 |
| 17 | 3 | 0.022 | 0.2 | 3.9664 | 50 |
| 18 | 3 | 0.022 | 0.3 | 3.93668 | 50 |
| 19 | 3 | 0.022 | 0.4 | 3.93152 | 48 |
| 20 | 3 | 0.022 | 0.5 | 3.94845 | 32 |
| 21 | 4 | 0.026 | 0.1 | 3.06581 | 10 |
| 22 | 4 | 0.026 | 0.2 | 3.11867 | 10 |
| 23 | 4 | 0.026 | 0.3 | 3.05484 | 10 |
| 24 | 4 | 0.026 | 0.4 | 3.0658 | 10 |
| 25 | 4 | 0.026 | 0.5 | 3.08874 | 8 |

Figure 1: Apriori, combinaciones Soporte-Confianza

Como resultados se puede observar una notable disminución en la cantidad de reglas de asociación a medida que tanto el soporte como la confianza aumentaban, llevando así por consecuencia a una disminución en los tiempos de ejecución, ya que no existe mayor cantidad de combinaciones que cumplan con

los parametros mínimos establecidos, bajando desde 678 reglas obtenidas con un soporte = 0.01 y una confianza de 0.1 hasta solo 8 reglas con un soporte = 0.026 y una confianza = 0.5. Como se muestra en la imagen anterior, existe una tendencia de disminución de las reglas de asociación a mayor soporte y confianza, siendo las de la última combinación las que se podrían considerar como las más solidas en cuanto a dichos parametros, pudiendo así utilizarse para los fines que estime conveniente.

Para el punto 2, se seleccionó como soporte a 0.01, ya que es este el que permite la mayor cantidad de reglas de asociación a lo largo de sus distintas combinaciones de confianza, por lo tanto, se comienza el enfoque particional sobre un gran número de reglas encontradas, para así facilitar su análisis.

## 3.2 Enfoque particional

Para está parte del experimento se busca dividir el dataset original en k particiones, las cuales son 2, 4, 6, 8 y 10. Esto asociandolos a centroides dados por puntos randoms. Luego de ejecutar el algoritmo Apriori en cada partición se buscó compararlos con la ejecución del Apriori general ( única partición) para así poder definir una relación porcentual con el número de reglas de asociación encontradas.
Las comparaciones obtenidas se centran en el tiempo de ejecución de cada partición del dataset original junto a las reglas obtenidas, comprobando que se encuentren dentro del Apriori general y no sea solo una regla que cumple con las confianza y soporte mínimo de manera local a la partición correspondiente. Los resultados se pueden apreciar en la tabla que se muestra a continuación.

| Particiones | Soporte | Confianza | Tiempo de ejecución | Número de reglas | Coincidencias con general |
|---|---|---|---|---|---|
| 1 | 0.01 | 0.1 | 179 segundos | 678 | 678 |
| 2 | 0.005 | 0.1 | 103 segundos | 100 | 100 |
| | | | 103 segundos | 98 | 98 |
| 4 | 0.0025 | 0.1 | 148 segundos | 16 | 16 |
| | | | 143 segundos | 28 | 20 |
| | | | 170 segundos | 16 | 16 |
| | | | 143 segundos | 18 | 16 |
| 6 | 0.001667 | 0.1 | 193 segundos | 20 | 14 |
| | | | 205 segundos | 20 | 14 |
| | | | 193 segundos | 18 | 14 |
| | | | 204 segundos | 28 | 16 |
| | | | 208 segundos | 20 | 16 |
| | | | 193 segundos | 20 | 14 |
| 8 | 0.00125 | 0.1 | 242 segundos | 26 | 14 |
| | | | 259 segundos | 24 | 14 |
| | | | 235 segundos | 20 | 14 |
| | | | 241 segundos | 16 | 14 |
| | | | 240 segundos | 24 | 14 |
| | | | 231 segundos | 20 | 14 |
| | | | 244 segundos | 22 | 14 |
| | | | 255 segundos | 28 | 14 |
| 10 | 0.001 | 0.1 | 336 segundos | 30 | 14 |
| | | | 347 segundos | 22 | 14 |
| | | | 355 segundos | 26 | 14 |
| | | | 354 segundos | 30 | 14 |
| | | | 371 segundos | 24 | 14 |
| | | | 366 segundos | 30 | 14 |
| | | | 374 segundos | 34 | 14 |
| | | | 364 segundos | 30 | 14 |
| | | | 373 segundos | 22 | 14 |
| | | | 392 segundos | 34 | 14 |

Table 3: Enfoque particional, tiempos de ejecución, números de reglas y coninincidencias con general

Como se puede apreciar en la tabla 3, existe una tendencia en cuanto a los tiempos de ejecución, los cuales van en aumento mientras más pequeña sea la partición, no siendo proporcionales al número de reglas obtenidas por cada una. También cabe recordar que el soporte de cada partición va disminuyendo proporcionalmente a la cantidad de datos analizados en la partición

Con respecto a la ejecución del Apriori en el total de los datos (general) el cual obtuvo 678 reglas de asociación, ninguna partición pudo acercarse a esto, no llegando nisiquiera a un tercio de las reglas encontradas. No obstante, si se pudo observar una tendencia, ya que a medida que era más pequeña la partición, era mayor el número de reglas encontradas, pero, estas se consideraban como "reglas locales", ya que luego de eso se comparaba la coincidencia con las reglas del Apriori general, y las que calzaran con ellas, se consideraban como aptas para ser una regla de asociación global.

En cifras, se pueden calcular los siguiente promedios asociados a las particiones:

| Particiones | Tiempo de ejecución | # de reglas | Coincidencias con general |
|---|---|---|---|
| 1 | 179 segundos | 678 | 678 |
| 2 | 103 segundos | 99 | 99 |
| 4 | 151 segundos | 19.5 | 17 |
| 6 | 199 segundos | 21 | 14.6 |
| 8 | 243 segundos | 22.5 | 14 |
| 10 | 363 segundos | 28.2 | 14 |

Table 4: Enfoque particional, promedio de particiones

Como se puede ver en la tabla 4, salvo la partición 2, el resto de ellas tiene una tendencia a aumentar el número de reglas de asociación locales encontradas, no obstante, no coinciden con alguna regla del Apriori general, por lo que no son validas para esto. En resumen, mientras más pequeña sea la partición, mayor será el tiempo de análisis gastada en cada una y menor el beneficio obtenido, ya que solo se generan reglas locales, no aptas para definirlas como globales.

También, cabe la posibilidad de que el promedio no fuese tan representativo, ya que es cierto que al utilizar más particiones se pueden obtener mayor cantidad de reglas, esto queda demostrado de la siguiente forma:

| Particiones | Tiempo de ejecución total | # de reglas totales | Coincidencias totales |
|---|---|---|---|
| 1 | 179 segundos | 678 | 678 |
| 2 | 206 segundos | 198 | 198 |
| 4 | 604 segundos | 78 | 68 |
| 6 | 1.196 segundos | 126 | 88 |
| 8 | 1.947 segundos | 180 | 112 |
| 10 | 3630 segundos | 282 | 140 |

Table 5: Enfoque particional, cifras totales

Como se aprecia, en un principio tanto reglas locales como globales mantienen una proporción 1 a 1, la cual va disminuyendo con la cantidad de particiones, llegando en el final de las ejecuciones con 10 particiones a una efectividad menor al 50%. Para cada fracción de este experimento, la efectividad en cuanto a coincidencia local-global es:

| Particiones | Efectividad |
|---|---|
| 1 | 100% |
| 2 | 100% |
| 4 | 87% |
| 6 | 69.9% |
| 8 | 62% |
| 10 | 49.6% |

Table 6: Enfoque particional, efectividad global-local

A raíz de los expuesto anteriormente, creemos que no es recomendable utilizar un enfoque particional en un algoritmo Apriori, ya que a medida que aumenta la cantidad de particiones su consumo en tiempo se volverá muy elevado y la efectividad de su respuesta irá bajando cada vez más. Por lo tanto se recomienda su uso solo de manera general, o sea, con el total de los datos sin ninguna partición.

## 3.3 K-means

Para la ejecución del algoritmo K-means se considera la busqueda de nuevas relaciones antes la imposibilidad de encontrar reglas de asociación, por lo cual se propone buscar relaciones usando las variables 'Quantity', 'UnitPrice' y 'CustomerID', lo obtenido será mostrado a continuación.

```
In [87]: wcss = []

for i in range (1,11):
    kmeans = KMeans(n_clusters = i, max_iter=300)
    kmeans.fit(X_train)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,11),wcss)
plt.title("Codo de Jambu")
plt.xlabel("numero de Clusters")
plt.ylabel("WCSS")
plt.show()
```

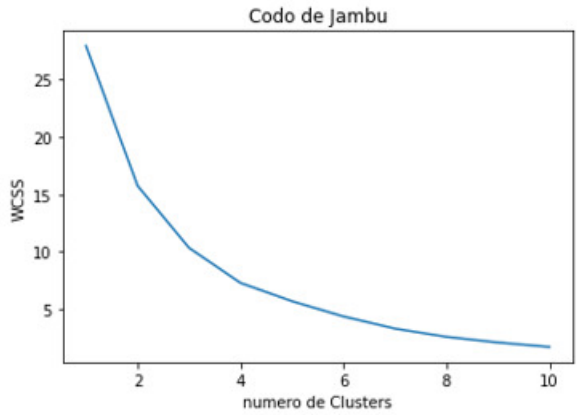



Figure 2: Codo de Jambu.

Primero, utilizamos la función para el cálculo de WCSS y poder así determinar el valor óptimo de clusters a utilizar, y como podemos ver, existe una curva sobre la cual se demuestra un punto de quiebre entre 2 y 4 cluster, por lo cual se considera una partición óptima en un punto medio, en este caso 3

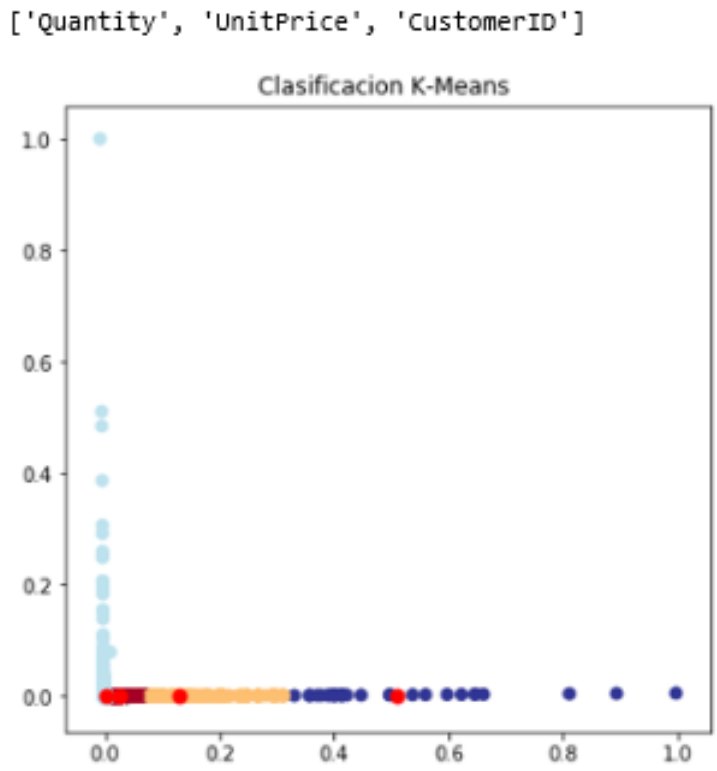
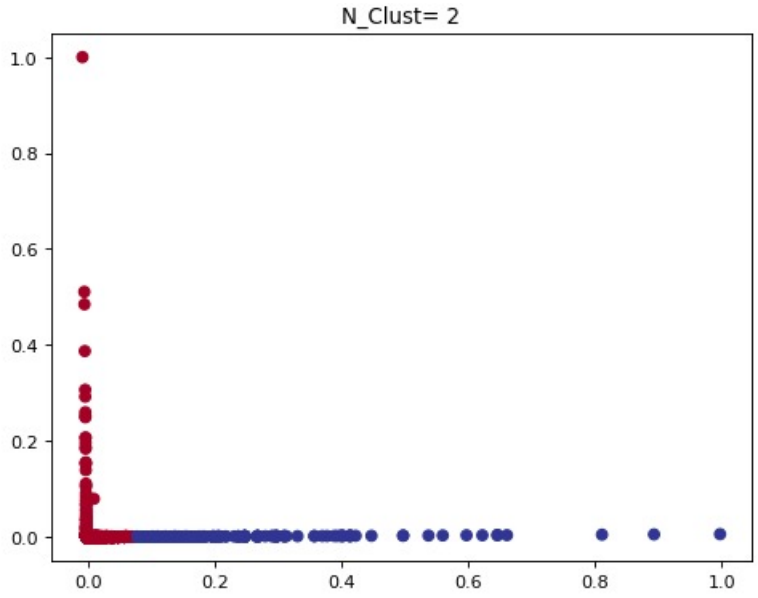



Figure 3: Kmeans, k=3

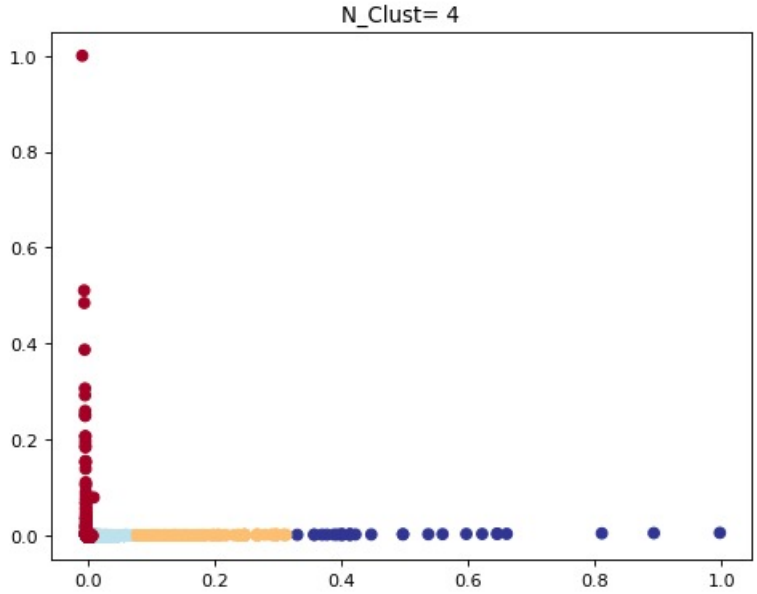
Luego, se procedio a hacer particiones de 2,4,6,8 y 10 para demostrar la efectividad del cálculo de WCSS, donde como se verá a continuación, queda demostrado que 3 es el valor óptimo en cuanto a cantidad de clusters, siendo incluso desde 4 hacia adelante poco diferenciador con respecto a los datos, volviendose innecesario para esta ocasión.




```
tiempo: 0.8153574467 segundos.

N_Centroides:
[[2.17238592e-03 3.83901600e-04]
 [1.50680546e-01 1.46507140e-04]]
```

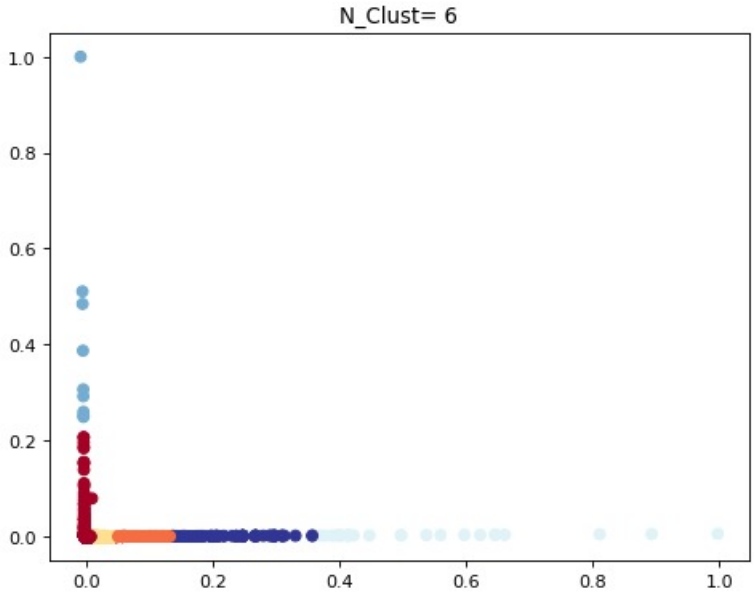
Figure 4: Kmeans, k=2




```
tiempo: 1.2681474686 segundos.

N_Centroides:
[[1.56356028e-03 3.89400070e-04]
 [1.30847448e-01 1.47971488e-04]
 [2.57861398e-02 1.70639348e-04]
 [5.08360219e-01 1.20098374e-04]]
```

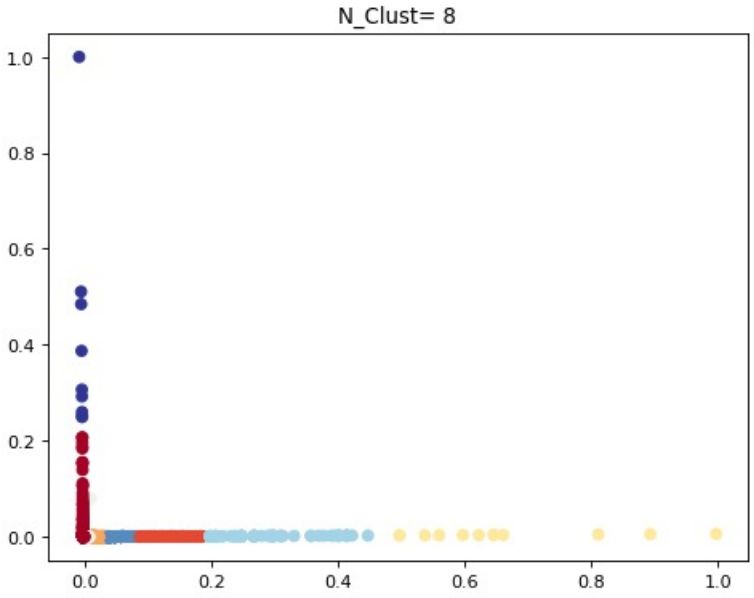
Figure 5: Kmeans, k=4




```
tiempo: 1.5612149239 segundos.

N_Centroides:
[[1.53566466e-03 3.77520891e-04]
 [7.35823342e-02 1.61256271e-04]
 [2.18490081e-02 1.77420991e-04]
 [5.26519948e-01 1.28571192e-04]
 [0.00000000e+00 4.16142649e-01]
 [2.03874463e-01 1.31738010e-04]]
```

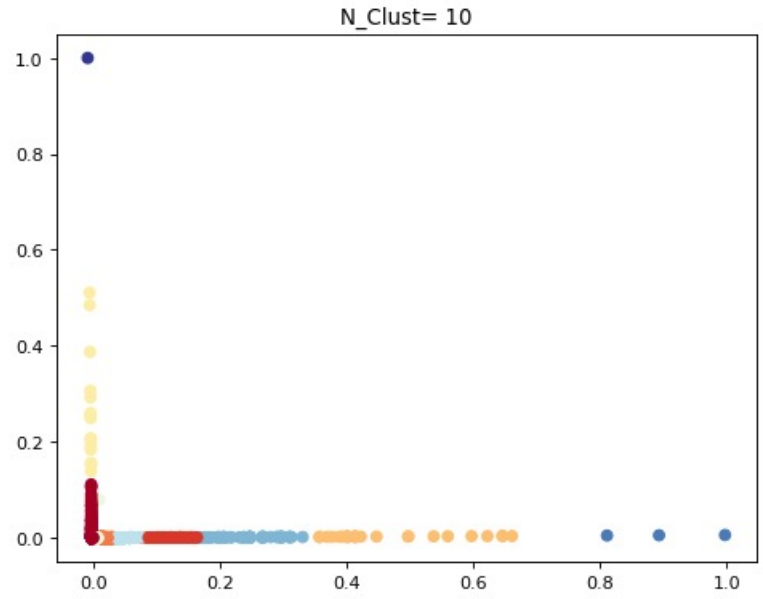
Figure 6: Kmeans, k=6




```
tiempo: 2.2651233673 segundos.

N_Centroides:
[[9.10575939e-04 4.09913340e-04]
 [1.14830113e-01 1.51887120e-04]
 [2.06848197e-02 1.70262886e-04]
 [6.66458290e-01 1.05308403e-04]
 [5.86207227e-03 1.61544027e-04]
 [2.74397444e-01 1.20461624e-04]
 [4.95701320e-02 1.72317913e-04]
 [0.00000000e+00 4.16142649e-01]]
```

Figure 7: Kmeans, k=8




```
tiempo: 2.6861701012 segundos.

N_Centroides:
[[9.10606816e-04 4.03957358e-04]
 [1.10963790e-01 1.54636018e-04]
 [2.04304295e-02 1.73461683e-04]
 [4.67989164e-01 1.29489423e-04]
 [0.00000000e+00 2.54691886e-01]
 [5.82129467e-03 1.60961774e-04]
 [4.95488501e-02 1.72271913e-04]
 [2.31074773e-01 1.14959009e-04]
 [9.03174272e-01 7.16383695e-05]
 [0.00000000e+00 1.00000000e+00]]
```

Figure 8: Kmeans, k=10

Finalmente, se calcularon los tiempos de ejecución para cada partición, siendo bastante bajos incluso para la tarea básica asignada ahora. Suponemos que de poder haber utilizado su correcta implementación para comparar con el Apriori con enfoque particional su resultados hubiesen sido mucho más rápidos pero menos precisos, ya que incluso la mayor partición del algoritmo Apriori generó una mayor cantidad de reglas que clusters encontrados por kmeans.

| Particiones | Tiempo de ejecución |
|---|---|
| 2 | 2.9 segundos |
| 4 | 3.5 segundos |
| 6 | 6.65 segundos |
| 8 | 5.31 segundos |
| 10 | 6.77 segundos |

Table 7: Ejecución K-means

# 4 Conclusión

Al no poder haber evaluado lo mismo bajo el algoritmo Apriori y el Kmeans no se pudo comparar fortalezas y debilidades de ambos de manera empírica. No obstante, si se puede sacar conclusiones bajo lo que se logro.

Es así que podemos demostrar que el algoritmo Apriori tiene un funcionamiento más solido cuando se evalua la totalidad de los datos de una vez y no de forma particionada, manteniendo un tiempo estable y proporcional a las reglas entregadas al final de la ejecución. En cambio, su contraprarte particional aumentaba los tiempos de ejecución y empeoraba los resultados entregados, logrando solo reglas de asociación catalogadas como locales, no siendo beneficiosas para un resultado global, lo cual bajaba la efectividad de la ejecución del algoritmo.

En conclusión, y bajo lo visto en este experimento el algoritmo Apriori entrega un mejor resultado cuando la cantidad de datos a analizar es mayor, tanto en tiempo de ejecución como calidad de las respuestas recibidad.

## References