



**Post Data**  
SMART SOLUTIONS

## Retail a la Medida

Juan Pablo de la Huerta  
Félix González  
Felipe López  
Marcelo Lozano  
Eduardo Vega

14 de marzo de 2022

# Índice

<b>1. Presentación y objetivos</b>	<b>3</b>
<b>2. Planificación de investigación</b>	<b>4</b>
2.1. Detalle de recursos . . . . .	4
2.2. Librerías utilizadas . . . . .	5
2.3. Modelación vector objetivo . . . . .	6
2.4. Preprocesamiento . . . . .	6
<b>3. Análisis exploratorio</b>	<b>9</b>
3.1. Recodificación de los datos . . . . .	13
<b>4. Entrenamiento</b>	<b>14</b>
4.1. K=3 y K= 4 . . . . .	16
4.2. Caracterización de los cluster . . . . .	18
<b>5. Implementación Comercial</b>	<b>20</b>
5.1. Conclusión . . . . .	21

# 1. Presentación y objetivos

En el presente documento se presenta un modelo para segmentar los clientes de retail, con el fin de entregar productos financieros personalizados y disminuir la incertidumbre de una empresa, buscando replantear la estrategia de préstamo de dinero a sus clientes. La necesidad de esta herramienta, surge desde distintos antecedentes que podemos rescatar del mercado financiero en Chile, como, por ejemplo:

- Según la CMF, la cartera de morosidad en Chile es de 2.503 millones de pesos respecto a no pagos de 90 o más días.
- Esta situación refleja la necesidad imperante de que las instituciones financieras busquen formas de disminuir este indicador.
- El último trimestre de 2021 se incrementó la deuda morosa en un 0,1 personas. El 77,1 directamente en pérdidas para las empresas que facilitan estos medios de pago, asociados al endeudamiento de sus cliente

Tener una buena segmentación de clientes, puede traer múltiples beneficios a la compañía de retail, como, por ejemplo:

- Mejorar la imagen de la empresa (en cuanto a confiabilidad y transparencia), haciéndola más atractiva para los clientes potenciales.
- Segmentar de mejor manera los clientes, basándonos en diferentes factores y comportamientos de pago
- Disminuir la incertidumbre del retorno del préstamo, generando ahorro al no prestar dinero a los clientes caracterizados como morosos

## 2. Planificación de investigación

### 2.1. Detalle de recursos

Se dispone de una base de datos de una empresa de retail la cual contiene información de los EEEF de sus clientes durante un año. La base de datos contiene 35.799.227 y 19 columnas, las cuales se presentan a continuación:

- ID\_CLIENTE: Identificación única del cliente. (“String”)
- TIPO\_PRODUCTO: Tipo de producto (C o R) (nominal)
- FECHA\_EMISION: Fecha de emisión del EECC. (objeto)
- MORA\_1: Monto en mora por 1 mes. (numérico)
- MORA\_2: Monto en mora por 2 meses. (numérico)
- MORA\_3: Monto en mora por 3 meses. (numérico)
- MORA\_4: Monto en mora por 4 meses. (numérico)
- SALDO\_FAVOR: Saldo a favor del cliente al momento de emitir EECC. (numérico)
- SALDO\_PENDIENTE: Saldo de la facturación anterior. (numérico)
- PAGOS: Suma de pagos dentro del mes (antes de la facturación). (numérico)
- CARGO: Suma de cargos correspondientes al mes de facturación. (numérico)
- TOTAL\_MES: Monto total del mes. (numérico)
- INTERESES: Intereses a pagar. (numérico)
- MINIMO: Pago mínimo. (numérico)
- MONTO\_ATRASADO: Monto atrasado. (numérico)
- MONTO\_CANCELAR: Monto total a cancelar. (numérico)

## 2.2. Librerías utilizadas

Dado el data set a analizar, se utilizarán las siguientes librerías:

- Numpy Lectura y manipulación de los datos
- PANDAS Para generar gráficos y análisis
- Matplotlib Para generar gráficos y visualización
- Seaborn Para generar gráficos y visualización
- Sckit-learn Generar modelos ML
- Apache Spark Preprocesamiento de BigData
- Google Colab Trabajo colaborativo del proyecto

```
1 !pip install pyspark
2 # Librerías para manejo y exploración de datos
3 import pandas as pd
4 import numpy as np
5 import matplotlib.pyplot as plt
6 import seaborn as sns
7 from google.colab import drive
8 import warnings
9 warnings.filterwarnings('ignore')

1 # Librerías para el trabajo con pyspark
2 from pyspark.sql import SparkSession
3 from pyspark.sql import functions as f
4 from pyspark.ml.feature import VectorAssembler
5 from pyspark.ml.clustering import KMeans
6 from pyspark.ml.evaluation import ClusteringEvaluator
7 from pyspark.ml.linalg import DenseVector, SparseVector, Vectors, VectorUDT

1 #Iniciación de trabajo con spark
2 spark = SparkSession\
3     .builder\
4     .appName('proyecto')\
5     .enableHiveSupport()\
6     .getOrCreate()
7 spark
8
9
10 plt.style.use('seaborn-whitegrid')
```

Figura 1: Importación de librerías

## 2.3. Modelación vector objetivo

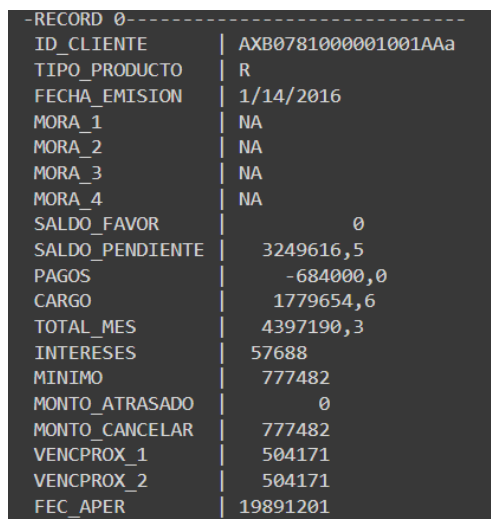
Dada la naturaleza del problema, lo que buscamos como empresa es generar una clusterización, a partir de los antecedentes entregados, y así poder identificar un número determinado de “tipos de clientes”, caracterizarlos según los componentes asociados a cada cluster y poner a disposición esta información para el manejo del cliente.

## 2.4. Preprocesamiento

Debido a que el dataset consta de un csv con más de 30 millones de observaciones, es que se opta por preprocesar la información con técnicas de Big Data, más en concreto mediante Apache Spark, que permite la manipulación de grandes volúmenes de información.

Comenzamos por instalar la librería PySpark para iniciar la manipulación de los datos, además de importar las librerías necesarias.

Se ocupa Google Colab y Drive para conectar con los datos a procesar y tener una mayor reproducción a todo tipo de usuario que ocupe jupyter, en caso de trabajar con este notebook en otro contexto, es necesario modificar las rutas de carga.



```
-RECORD 0-----
ID_CLIENTE      | AXB0781000001001AAa
TIPO_PRODUCTO   | R
FECHA_EMISION   | 1/14/2016
MORA_1          | NA
MORA_2          | NA
MORA_3          | NA
MORA_4          | NA
SALDO_FAVOR     | 0
SALDO_PENDIENTE | 3249616,5
PAGOS           | -684000,0
CARGO           | 1779654,6
TOTAL_MES       | 4397190,3
INTERESES       | 57688
MINIMO          | 777482
MONTO_ATRASADO  | 0
MONTO_CANCELAR  | 777482
VENCPROX_1      | 504171
VENCPROX_2      | 504171
FEC_APER        | 19891201
```

Figura 2: Datos en bruto

Podemos observar, entonces, que los datos corresponden al historial de

```

1 df= spark.sparkContext.textFile(path)
2
3 #borramos la primera fila
4 df_columns = df.first()
5 df_prueba = df.filter(lambda x: x != df_columns)
6 df_prueba.take(1)
7
8 AXB0781000001001AAa;R;1/14/2016;NA;NA;NA;      0; 3249616,5;  -684000,0;  1779654,6;  4397190,3;  57688;  777482;      0;  777482;  504171;  504171;19891201'
9
10 #convierte el tipo de las variables y cambia los NA a 0
11 df_procesado = df_prueba\
12     .map(lambda x : x.split(";"))\
13     .map(lambda x : (str(x[0]),str(x[1]),float(x[3].replace("NA","0")),float(x[4].replace("NA","0")),float(x[5].replace("NA","0")),float(x[6].replace("NA","0"))

```

Figura 3: Pre-procesamiento del dataset (replace : tipo de datos, NaN = 0)

los distintos clientes a lo largo de los meses del año 2016. Como se comento anteriormente sobre la enorme cantidad de datos, se decidió trabajar con el promedio y la desviación estándar de cada cliente, agrupado el total de observaciones que aparece según su ID, sin contabilizar las fechas. Como el objetivo es clasificar a los clientes en base a su comportamiento de pago, se opta por trabajar con los promedios de la variable MORA\_1, MORA\_2, MORA\_3 y MORA\_4, para cada uno de los ID registrados. Mientras que para el resto de las variables se trabajara con el promedio y la desviación estándar. Para mantener una noción de temporalidad o la cantidad de meses de historia del cliente con la institución, se eliminará la columna fecha y se reemplazará por una columna que indique la cantidad de meses involucrados en la obtención de los promedios y desviaciones estándar. De esta manera, podemos reducir las observaciones a una sola por cliente y analizar de manera más eficiente

Un punto para considerar es que todas las variables son reconocidas como STRING, por lo que será necesario transformarlos a FLOAT o INT, según corresponda. Según lo anterior, continuamos con el preproceso del dataset.

```

1 df_csv.printSchema()

root
 |-- ID_CLIENTE: string (nullable = true)
 |-- TIPO_PRODUCTO: string (nullable = true)
 |-- MORA_1: double (nullable = true)
 |-- MORA_2: double (nullable = true)
 |-- MORA_3: double (nullable = true)
 |-- MORA_4: double (nullable = true)
 |-- SALDO_FAVOR: double (nullable = true)
 |-- SALDO_PENDIENTE: double (nullable = true)
 |-- PAGOS: double (nullable = true)
 |-- CARGO: double (nullable = true)
 |-- TOTAL_MES: double (nullable = true)
 |-- INTERESES: double (nullable = true)
 |-- MINIMO: double (nullable = true)
 |-- MONTO_ATRASADO: double (nullable = true)
 |-- MONTO_CANCELAR: double (nullable = true)

```

Figura 4: CSV recodificado

*Vencprox\_1 Vencprox\_2 y Fec\_aper se borraron del procesamiento por la falta de información que aportaban estas variables.*

```

1 df_csv_procesado = df_csv.select(df_csv.columns).groupBy("ID_CLIENTE","TIPO_PRODUCTO")\
2     .agg(f.avg("MORA_1").alias("AVG_MORA1"),
3         f.avg("MORA_2").alias("AVG_MORA2"),
4         f.avg("MORA_3").alias("AVG_MORA3"),
5         f.avg("MORA_4").alias("AVG_MORA4"),
6         f.avg("SALDO_FAVOR").alias("AVG_SALDO_FAVOR"),
7         f.stddev("SALDO_FAVOR").alias("STD_SALDO_FAVOR"),
8         f.avg("SALDO_PENDIENTE").alias("AVG_SALDO_PENDIENTE"),
9         f.stddev("SALDO_PENDIENTE").alias("STD_SALDO_PENDIENTE"),
10        f.avg("PAGOS").alias("AVG_PAGOS"),
11        f.stddev("PAGOS").alias("STD_PAGOS"),
12        f.avg("CARGO").alias("AVG_CARGO"),
13        f.stddev("CARGO").alias("STD_CARGO"),
14        f.avg("TOTAL_MES").alias("AVG_TOTAL_MES"),
15        f.stddev("TOTAL_MES").alias("STD_TOTAL_MES"),
16        f.avg("INTERESES").alias("AVG_INTERESES"),
17        f.stddev("INTERESES").alias("STD_INTERESES"),
18        f.avg("MINIMO").alias("AVG_MINIMO"),
19        f.stddev("MINIMO").alias("STD_MINIMO"),
20        f.avg("MONTO_ATRASADO").alias("AVG_MONTO_ATRASADO"),
21        f.stddev("MONTO_ATRASADO").alias("STD_MONTO_ATRASADO"),
22        f.avg("MONTO_CANCELAR").alias("AVG_MONTO_CANCELAR"),
23        f.stddev("MONTO_CANCELAR").alias("STD_MONTO_CANCELAR"),
24    )

```

Figura 5: Agrupación por cliente con promedio y desviación estandar



### 3. Análisis exploratorio

Para comenzar con el análisis exploratorio, cargamos los datos preprocesados del archivo creado anteriormente. Dado los pares de variables TOTAL\_MES y MONTO\_CANCELAR pueden ser calculadas como una combinación lineal de otras, se opta por eliminarlas para evitar redundancia de información al igual que la Variable MÍNIMO con MONTO\_ATRASADO que pueden confundir al modelo ya que tienden a tener los mismos datos dejando solamente MONTO\_ATRASADO. Procedemos a observar la cantidad de observaciones y la cantidad de atributos, evidenciando la reducción de los datos, obteniendo un total de 3.464.374 registros, con un total de 19 atributos. Además, se explora la cantidad de datos nulos que tiene el data set preprocesado. Se aprecia que las variables asociadas a la desviación estándar contienen datos nulos, esto tendría relación con que algunos clientes tenían valor 0 para este atributo en el dataset original.

```
CLIENTES 0
FRECUENCIA_PAGOS 0
TIPO_PRODUCTO 0
AVG_MORA1 0
AVG_MORA2 0
AVG_MORA3 0
AVG_MORA4 0
AVG_SALDO_FAVOR 0
STD_SALDO_FAVOR 91160
AVG_SALDO_PENDIENTE 0
STD_SALDO_PENDIENTE 91160
AVG_PAGOS 0
STD_PAGOS 91160
AVG_CARGO 0
STD_CARGO 91160
AVG_INTERESES 0
STD_INTERESES 91160
AVG_MONTO_ATRASADO 0
STD_MONTO_ATRASADO 91160
dtype: int64
```

Figura 6: Valores Nulos en el dataset preprocesado

Debido a que la cantidad de observaciones con datos nulos es baja respecto a la totalidad de los datos, 91.160 sobre 3.464.374, se opta por eliminar dichas observaciones, también Identificamos el tipo de dato de cada atributo, observando que casi todas las variables son de tipo FLOAT, a excepción de CLIENTES y TIPO\_PRODUCTO (OBJECT) y FRECUENCIA\_PAGOS (INT). Por otra parte se aplico una función que nos permite revisar la estadística descriptiva de las variables de tipo INT y FLOAT, resaltando en verde los valores mayores a 0 y en azul los valores menores a 0. Como resultado, vemos que se presenta una gran dispersión de los datos, donde algunos atributos presentan un mínimo de 0 y un máximo con valores bajo  $-1e06$  y sobre  $1e06$ . Se procede con el análisis de las distribuciones.

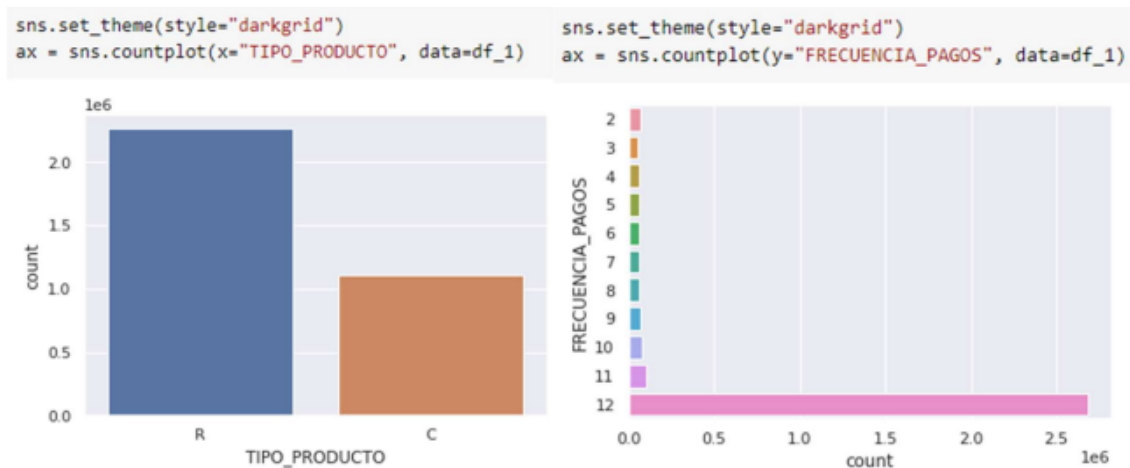


Figura 7: Desbalance de las variables

En ambas variables se observa un desbalance que podría afectar la representatividad de los modelos que se trabajen en etapas posteriores. Para FRECUENCIA\_PAGOS se propone recodificar la variable agrupando las frecuencias mayor o igual a 11 y de esta forma balancear el atributo.

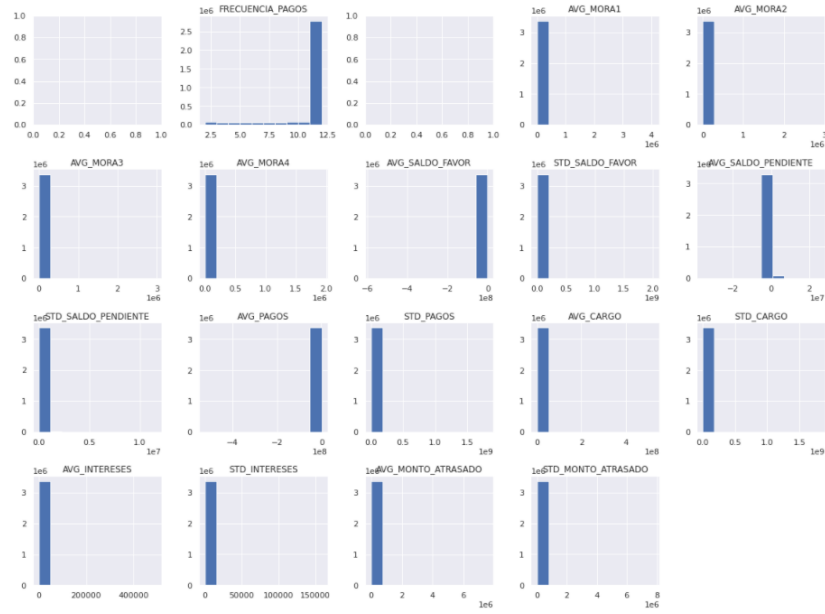


Figura 8: Visualización de atributos INT y FLOAT

No se observan distribuciones claras en ningún atributo. En general todos tienen su mayor cantidad de observaciones en torno a 0, con outliers bajo  $-1e06$  y sobre  $1e06$  que distorsionan su visibilidad y representatividad

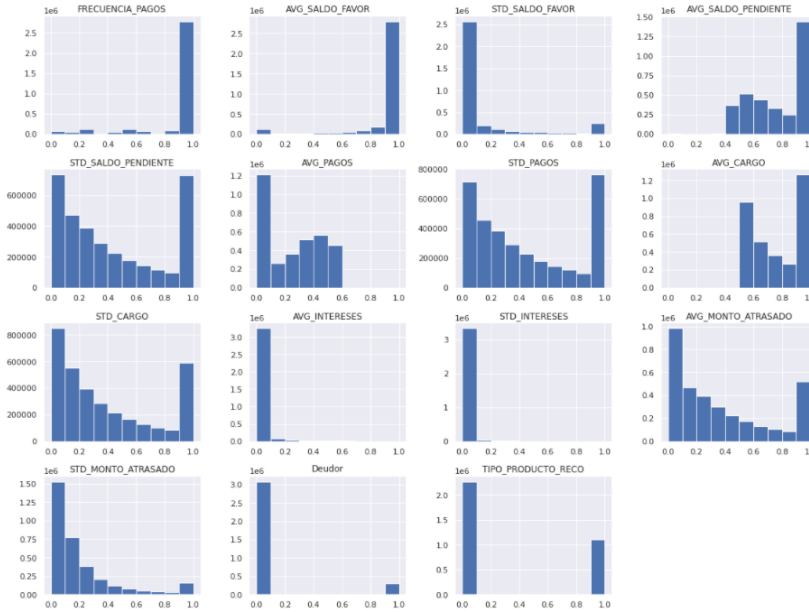


Figura 9: Visualización de atributos con threshold 1e05

Al aplicar estos límites inferior y superior, se puede visualizar de mejor manera las tendencias de las variables, las cuales en su mayoría están concentradas en valores cercanos a 0, y luego, la cantidad decrece. Para algunas variables como STD\_SALDO\_PENDIENTE, STD\_PAGOS, STD\_CARGOS, AGV\_MINIMO, STD\_MINIMO, AGV\_MONTO\_ATRASADO se observan 2 máximos locales, uno cercano a 0 y el otro cercano al límite superior impuesto, lo que podría ser el indicio de 2 tipos de clientes. Según los resultados obtenidos en el análisis exploratorio, es que se hace necesaria la recodificación y normalización de las variables.

### 3.1. Recodificación de los datos

Partimos por la eliminación del atributo cliente, ya que no es necesario para el proceso de clusterización posterior. Lo mismo con el tipo de producto. Siguiendo las buenas prácticas, se define como 1 al tipo de producto C y como 0 al tipo de producto R. Dado el análisis exploratorio previo, se optará por filtrar el dataframe entre  $-1e05$  y  $1e05$  con tal limitar la cantidad de outliers, obteniendo un total de 3.373.214 registros con un total de 18 atributos. En base a lo sugerido por el problema, se recodifican las variables MORA\_1, 2, 3 y 4, categorizando como deudor al usuario si es que presenta valores mayores a 0 en MORA\_3 o en MORA\_4.

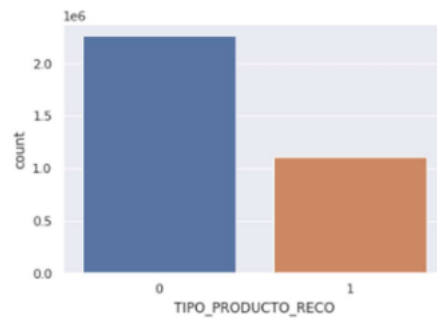


Figura 10: Visualización de la variable TIPO\_PRODUCTO recodificada

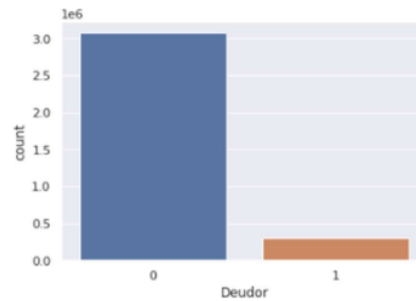


Figura 11: Visualización variable Deudor

## 4. Entrenamiento

Se entrenará el modelo Kmeans, con distintas cantidades de cluster y se guardará en una lista el valor obtenido de la inercia para graficar. Obteniendo el siguiente resultado:



Figura 12: Gráfico de codo

También se aplicará un gráfico de Silueta con el fin de comparar y complementar la información aportada por el gráfico de codo sumando otra métrica para la óptima distribución de los cluster.

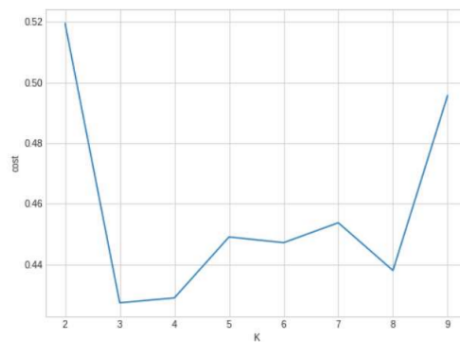


Figura 13: Gráfico de la silueta

Dada la aplicación de Kmeans y Silhouette, para ambas metodologías se observa una mayor tasa de cambio con K=2, por lo que se procedió con el entrenamiento con este parámetro, para finalmente mediante boxplot caracterizar cada cluster y generar los perfiles de clientes. Entrenamiento con K=2

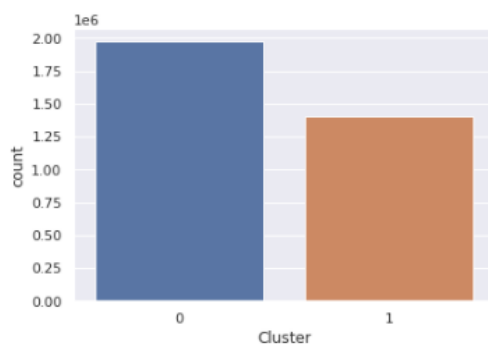


Figura 14: Distribución de los 2 Clusters

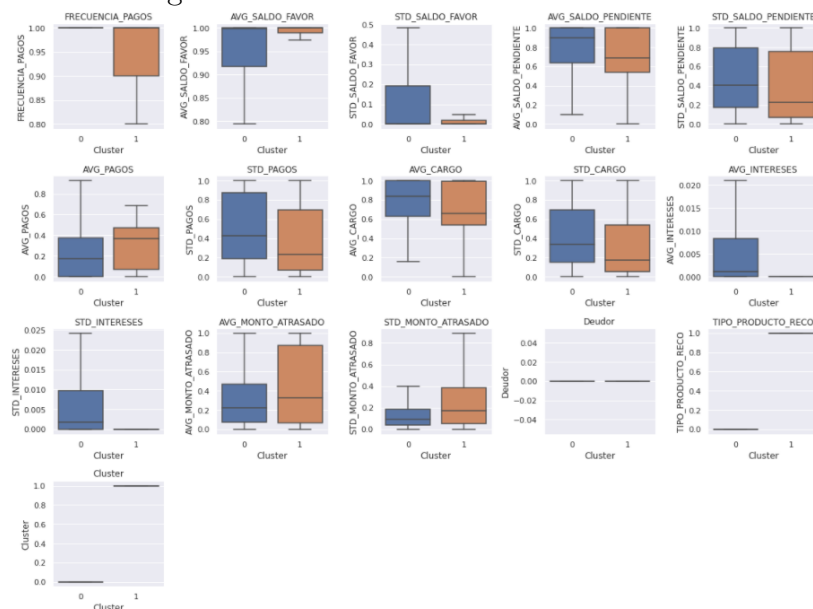


Figura 15: Caracterización de los perfiles

A partir de los gráficos obtenidos, podemos concluir que resulta una mejor diferenciación de manera general con k=2, lo anterior guarda relación con lo obtenido por la gráfica de codo y la silueta

#### 4.1. K=3 y K= 4

Siguiendo el esquema anterior se presenta el total de distribución por cada cluster respectivamente a Cluster 0 : 27 % , Cluster 1: 42 % y Cluster 2 :29 % resultando con mayor aglomeración de datos el cluster 1

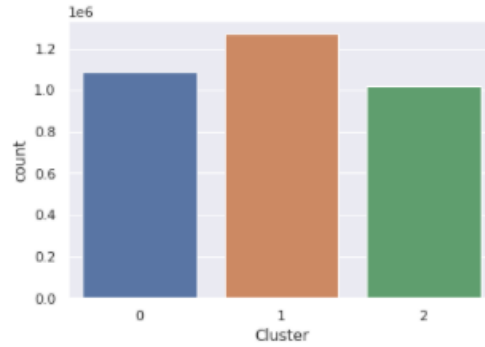


Figura 16: Distribución de los 3 Clusters

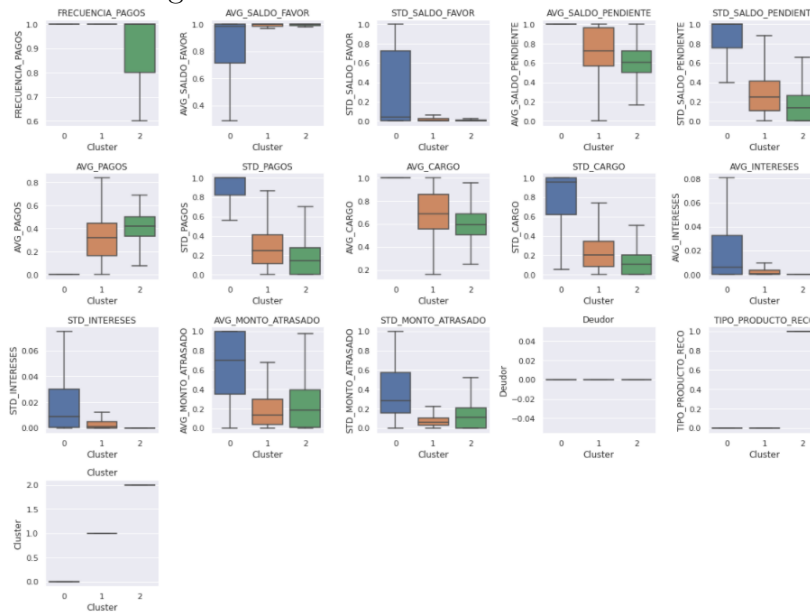


Figura 17: Caracterización de los perfiles K=3



Respecto a la total distribución de trabajar con 4 cluster se observa que Cluster 0 presenta un 20 % del total de datos aglomerados, Cluster 1 29 %, Cluster 2 27 % y Cluster 3 21 % y en comparación al entrenamiento anterior el Cluster 1 sigue manteniendo la mayoría de datos de las divisiones.

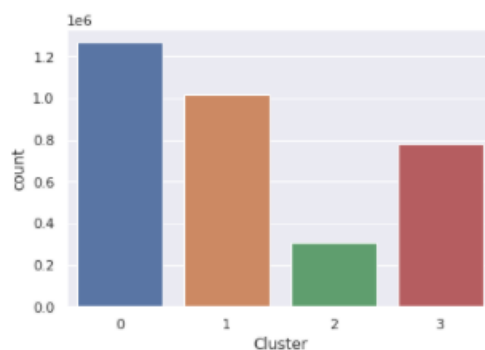


Figura 18: Distribución de los 3 Clusters

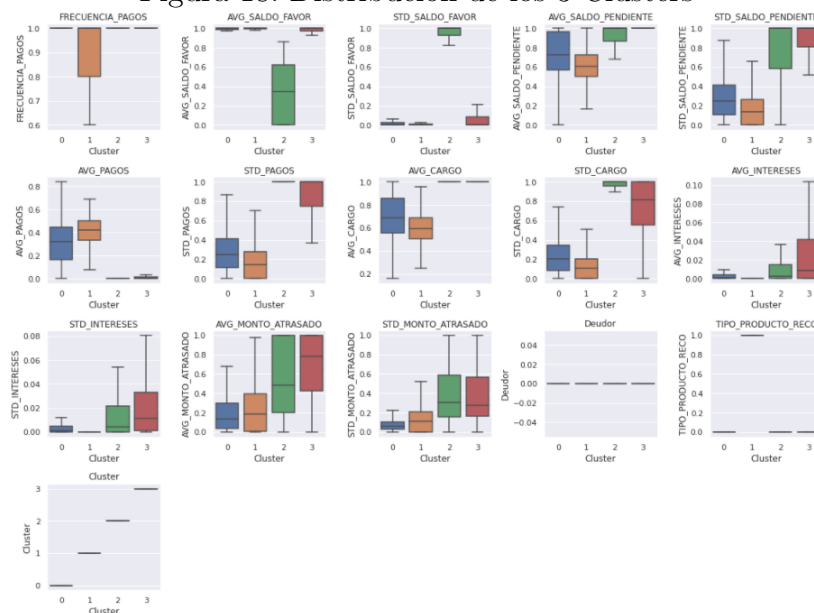


Figura 19: Caracterización de los perfiles K=4

## 4.2. Caracterización de los cluster

Al comparar la distribución por Cluster con las variables de *Promedio de pagos* y *Promedio de cargos* relacionado a *Promedio de montos atrasados*, las variables seleccionadas guardan relación con el objetivo de la propuesta de trabajo que es disminuir la perdida respecto a los prestamos otorgados.

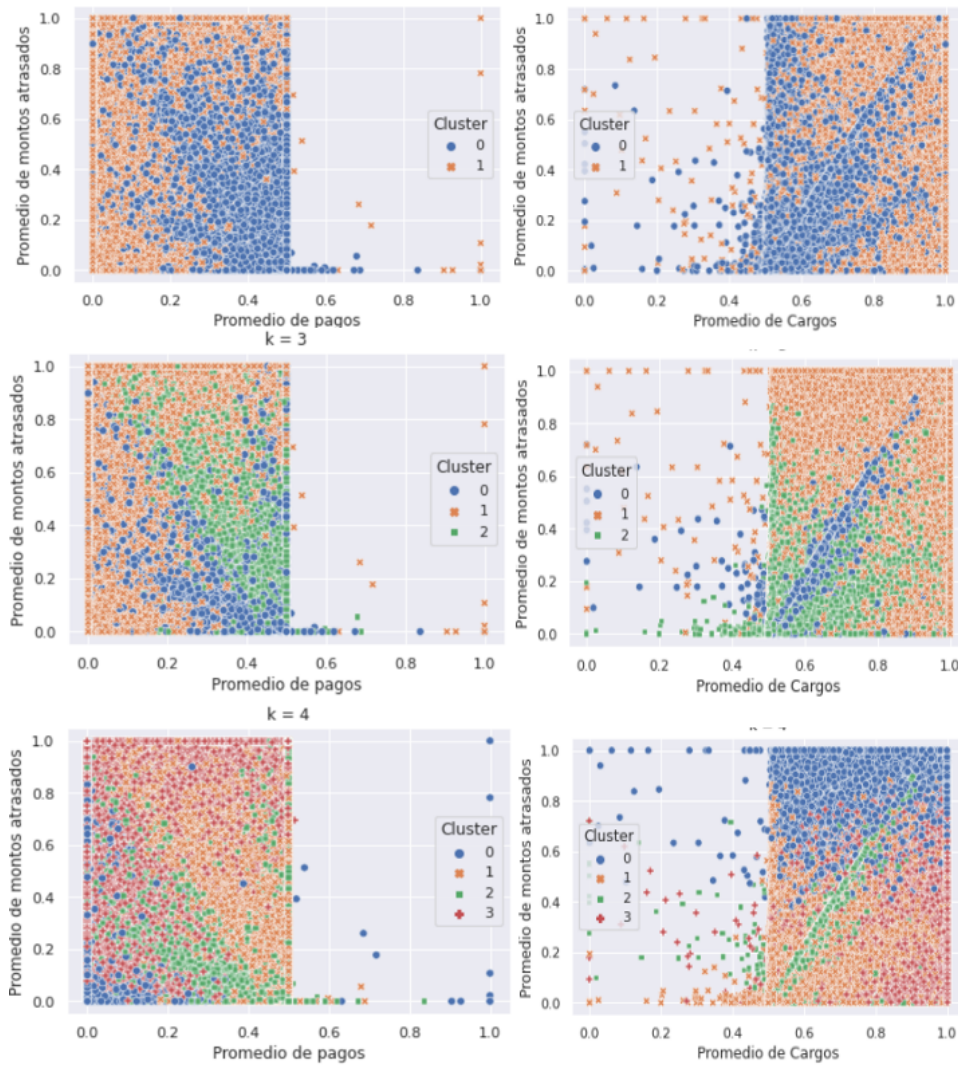


Figura 20: Comparación de la distribución con Promedio de pagos y Promedio de cargos por Montos atrasados

Cuadro 1: Caracterización de clientes con 3 Cluster

Atributos	0	1	2
AVG SALDO FAVOR	-2454	-16.636	-2384
AVG SALDO PENDIENTE	36.441	91.426	25.480
AVG PAGOS	-29.531	-88.474	-20.210
AVG CARGOS	30.699	90.277	22.288
AVG INTERESES	508	2.833	0
AVG MONTO ATRASADO	14.380	60.370	23.016
STD SALDO FAVOR	4.041	26.140	3.699
STD SALDO PENDIENTE	19.705	78.009	15.753
STD PAGOS	19.613	79.545	16.639
STD CARGOS	16.837	70.645	12.732
STD INTERESES	249	1850	0
STD MONTO ATRASADO	6.349	34.337	12.596
FRECUENCIA PAGOS	11	11	10
TIPO PRODUCTO RECO	0	0	1
DEUDOR	0	0	0

Por lo anterior se define trabajar con 3 cluster ya que permite una mejor diferenciación entre las características de cada grupo de clientes en comparación a 2 o 4 clientes, también desde este punto se puede caracterizar de una forma mucho más clara que tipo de cliente tiene cada cluster :

- Cluster 0 : Proporciones de 90 % no deudores, 92 % Tipo de producto revolving y 85 % de Frecuencia de pagos igual a 12 meses, respecto a sus atributos promedio tenemos que el Saldo Pendiente \$34.441 ,Saldo a favor \$2.454 pagos \$29.531 , Cargos \$30.699
- Cluster 1 :Proporciones de 95 % no deudores, 100 % Tipo de producto revolving y 82 % de Frecuencia de pagos igual a 12 meses, respecto a sus atributos promedio tenemos que el Saldo Pendiente \$16.636 , Saldo a favor \$188.474 pagos \$91.426 , Cargos \$90.227
- Cluster : 2 Proporciones de 88 % no deudores, 100 % Tipo de producto Cuotas y 71 % de Frecuencia de pagos igual a 12 meses, respecto a sus atributos promedio tenemos que el Saldo Pendiente \$25.480 , Saldo a favor \$2.384 pagos \$20.210 , Cargos \$22.280

## 5. Implementación Comercial

La implementación comercial tiene como objetivo poder categorizar a los nuevos clientes que luego de 3 meses en el sistema se puedan definir como **deudores** o no. Esto otorga el beneficio de automatizar el trabajo de los ejecutivos que elaboran las campañas de marketing disminuyendo los costos de la empresa y ocupando ese talento en otras tareas que no pueden ser implementadas por el modelo.

Para cada Cluster existen Productos recomendados:

- Cluster 0 : Aumento en el cupo de su tarjeta de crédito por la constancia en sus pagos dándoles la oportunidad de poder aumentar el umbral de gastos.
- Cluster 1 : Cambiar el tipo de producto de Revolving a Cuotas bajando las tasa de interés de sus cuotas y aumento la repactación de sus deudas
- Cluster 2: Aumentar el numero de cuotas que pueden tener y la repactación de sus deudas

Los productos recomendados buscan aumentar la satisfacción de los clientes generando una mayor comodidad tanto al comprar como al pagar sus deudas en la empresa de retail, esto da la posibilidad de reducir las perdidas por morosidad al dar soluciones acorde al comportamiento de pago, atajando a tiempo a las personas que no tienen la posibilidad de pagar su deuda.

## 5.1. Conclusión

Para finalizar existen limitaciones que nos encontramos en el camino al trabajar con el dataset otorgado, ya que existían variables que no entregaban información o no teníamos la posibilidad de comunicarnos con los creadores de los datos, por lo que incluir más atributos financieros al dataset podría ser de mucha ayuda al igual que datos demográficos y socio-educacionales. Lo anterior llevaría a una mayor segmentación de clientes de manera más clara para el equipo de marketing y también generar ofertas mucho más especializadas. En segundo lugar al tener una segmentación más precisa es automatizar la creación de las ofertas generando otra arista a la implementación del modelo, dando a una mayor producción junto a un menor costo a la empresa. Por ultimo se intentó realizar entrenamiento con otros modelos de datos, pero debido a lo largo del dataset no se pudieron llevar a cabo, por lo que es necesario explorar otros algoritmos con mayor capacidad y demanda de recursos para tener en consideración mejores métricas en el modelo.