DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
Inteligencia de negocios 202220 – Laboratorio 1
PROFESORA: Haydemar Nuñez

Nombres	Apellidos	Código	Login
María Sofía	Álvarez López	201729031	ms.alvarezl
Brenda Catalina	Barahona Pinilla	201812721	bc.barahona
Alvaro Daniel	Plata Márquez	201820098	ad.plata

Informe de laboratorio #2

Perfilamiento de los datos

En esta sección tendremos un primer acercamiento con los datos, en esto vemos que tenemos 5530 datos (filas) y 15 columnas (variables). Comparando con el diccionario de datos, vemos que el negocio nos entregó 15 columnas también. Luego, hicimos uso de la herramienta de pandas profiling, donde podemos destacar

Primero, tenemos 5933 celdas con ausencias, lo que corresponde al 7.2% del total del dataset. Para ellas, más adelante, debe implementarse una estrategia de imputación o eliminación, según sea el caso. Asimismo, vemos que no tenemos filas duplicadas.

Las variables que tienen valores vacíos son: OneOffPurchases_frequency con 2740 celdas vacías (i.e. 49.5%), Gender con 2714 celdas vacías (i.e. 49.1%), Cash_advance_frequency con 166 celdas vacías (i.e. 3%), Tenure con 163 celdas vacías (i.e. 2.9%), Cash_Advance_Trx con 150 celdas vacías (i.e. 2.7%). Asimismo, se tienen 9 variables numéricas y 6 categóricas. Esto NO concuerda con el diccionario dado por BancAlpes. Encontramos que el tipo de dato identificado por pandas profiling es incorrecto para las siguientes variables, según lo indicado por el documento del diccionario, pues estas variables deben ser numéricas y fueron clasificadas como categóricas:

CashAdvance: De acuerdo con el diccionario de datos, es numérica. No obstante, pandas profiling la determina como categórica por la presencia de valores como "??" u otros, en teoría numéricos, que finalizan con "?ñ", como" 00.?ñ". Esto debe tratarse en la limpieza de datos, seguramente asumiéndolos como errores de tipeo.

PurchasesTrx: En este caso, la variable debe ser numérica también. Por tener caracteres con "?", pandas profiling la etiqueta como categórica. Esto debe tratarse en la limpieza de datos, seguramente asumiéndolos como errores de tipeo.

Minimum_payments: Debe ser numérica y es catalogada como categórica por la presencia de caracteres como "??", "ñ?, ?". De nuevo, esto será tratado más adelante.

Tenure: Debe ser numérica y, por caracteres extraños como "?" o "ñ" en algunas celdas (o acompañando a ciertos valores numéricos), es definida como categórica.

Para todas estas variables, hay algunos datos con inconsistencias o datos atípicos y, por tanto, deben ser procesados más adelante.

Ahora, de las variables catalogadas como numéricas, tenemos que las que tienen una distribución continua son:

Balance: Es una variable real distribuida en el rango [-4587.892398, 7390.19856]. De acuerdo con el diccionario y los expertos, esta debe ser siempre positiva y estar en el rango [0, 5000].

Balance_frequency: De acuerdo con el negocio, debe estar distribuida entre 0 y 1. Vemos algunos outliers en 1000.

Purchases: Es una variable real distribuida en el rango [0, 9661.37]. De acuerdo con el negocio, debe estar distribuida entre [0, 15000], por lo que concluimos que los valores se encuentran en el rango definido. Nos parece curioso que haya valores decimales, pues según Bancalpes, esta variable representa la cantidad de compras realizadas por los clientes, y estas deberían ser números enteros. Suponemos, entonces, que se refiere al número promedio de compras realizadas por una cuenta en un intervalo de tiempo determinado.

OneOffPurchasesFrequency: Tiene una distribución continua entre 0 y 1, como es de esperarse de acuerdo con los expertos

Purchases_frequency: De acuerdo con el negocio, debe estar distribuida entre 0 y 1. Vemos algunos outliers en 1000.

Cash_advance_frequency: Tiene una distribución continua entre 0 y 1.5, cuando su valor máximo debería ser 1. Vemos entonces que hay datos fuera de rango.

Credit_limit: Es una variable con distribución continua en el rango [50, 12500]. De acuerdo con el negocio, esta variable, que representa límite de crédito por usuario, debe estar distribuida entre [0, 10000]. Vemos que hay algunos pocos valores que se salen del límite superior. No obstante, puede tener sentido que algunos usuarios tengan un límite crediticio entre [10000, 12500]. Este tipo de decisiones las trataremos más adelante.

Payments: Cantidad pagada por el usuario, debe estar distribuida entre [0, 10000] y se encuentra en el rango [0.056466, 9933.62261]. Está adecuadamente distribuida según el negocio.

Además de las variables antes mencionadas, encontramos que hay otras que también siguen una distribución continua y que, debido a los outliers y los valores corruptos (i.e. con ñ), no se evidenciaba una clara distribución de los datos, ya que los rangos con los que pandas construyó el histograma eran muy grandes.

Por otro lado, vemos que las variables numéricas con distribución discreta son:

CashAdvanceTrx: Parece ser que tiene una distribución discreta, lo cual concuerda con la descripción dada en el diccionario. No obstante, tiene muchos outliers y es difícil saberlo.

PurchasesTrx: Sucede algo similar que con CashAdvanceTrx.

Asimismo, las variables (realmente) categóricas son:

CustID: La identificación del cliente, en que todos los valores son únicos. No da información relevante para el modelo de agrupación.

Gender: Representa el género de la persona. Tiene muchas ausencias, como se mencionó previamente. Una distribución de esta variable la veremos más adelante.

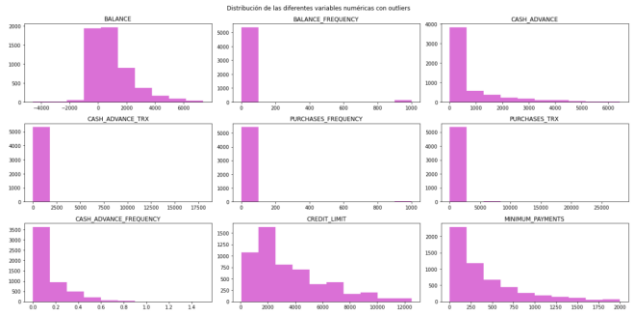
Vemos entonces que la calidad de los datos, en general, no es muy buena. No sólo hay variables con datos fuera de rango según lo definido por el negocio; sino que, además, hay algunos datos inconsistentes (con ?, ñ?, ??, por ejemplo) o nulos (vacíos). Para poder implementar los modelos, es necesario corregir estas inconsistencias en los datos

Preparación de los datos

1. **Eliminación de columnas:** La columna CUSTID es la identificacion de la tarjeta de crédito. Como esta columna no representa ningún dato significativo para el proceso de la creación del modelo, procedemos a eliminarla. Como se indicó en el perfilamiento de los datos, las columnas OneOffPurchases_frecuency y Gender cuentan con un porcentaje de celdas faltantes significativo: 49.5% y 49.1%, respectivamente. Por esta razón si realizamos imputación de datos para estas columnas, estaríamos sesgando los datos, por lo que también serán eliminadas.

2. **Manejo de valores atipicos / no válidos:** En el perfilamiento de los datos mencionamos los datos no válidos. Podemos ver que la mayoría de estos datos no válidos tienen un número y al final un "?ñ". Asumiremos que esto fue un error de tipeo al momento de ingresar los datos, por lo que eliminaremos el "?ñ" y dejaremos el número. Mas adelante revisaremos si los datos cumplen con los rangos establecidos. Después de este proceso, el único valor no válido es "??**", que representa el 5.26% de los datos.** Para evitar eliminar el 5% de los datos, estos se convertirán en datos `np.NaN` y serán manejados en más adelante, con los demás datos nulos del dataset. Para esto, usaremos una funcion que nos convertirá todo lo que no se pueda convertir a `float` a `np.NaN`.

3. **Outliers:** Para ver la información de las columnas con outliers, construimos los siguientes diagramas:



Se realizó un análisis de cada columna con outliers y según el caso se tomaron decisiones respectivas para manejarlos. Debido a la extensión de esta parte de la preparación de los datos, le sugerimos al lector remitirse al notebook para verificar toda la información sobre este proceso.

4. **Valores fuera de rango:** encontramos que en total hay 1589 valores fuera de rango antes de realizar el procesamiento para los outliers. Después de esto, quedan muy pocos, por lo que se convierten en nulos y se procesarán en el siguiente punto.
5. **Manejo de nulos:** El porcentaje de celdas vacías es de 24.38%. Este es un porcentaje considerable de los datos. En realidad, no vale la pena eliminarlos. Por lo tanto, como se dijo antes, seguiremos una estrategia de imputación con la media (al ser variables continuas).

Modelamiento y Validación de cada algoritmo

Algoritmo KNN (Encargada: Brenda Barahona):

Con este algoritmo se ubican los centros dentro del conjunto de datos de manera aleatoria. La cantidad de centros depende de la cantidad de grupos que se deseen o se necesiten armar. Estos centros usan la distancia euclidiana al cuadrado para los demás puntos, esto con el objetivo de identificar la menor distancia entre sí. Este algoritmo realiza un proceso iterativo, donde, en las primeras iteraciones se identifican grupos de puntos cercanos y se reubican los centros por medio del promedio, esto para que los centros tengan una posición más representativa, en las demás iteraciones se vuelven a calcular las distancias, usando la ubicación de los centros.

Proceso:

Con la librería KMeans de sklearn.cluster se realiza el agrupamiento. Para hacer uso de esto, se debe colocar un hiperparámetro "n_clusters", el cual determina el número de grupos que desea el usuario o que se obtiene al analizar los coeficientes silueta.

Para conocer los valores de los centros usados para realizar la agrupación, se hace uso del atributo "cluster_centers_" y para encontrar el cluster al que pertenece cada uno de los grupos, se hace uso del atributo llamado "labels_"

Se realizaron 4 pruebas, donde se variaban las columnas o variables seleccionadas para realizar la clasificación. En primer lugar, para realizar esta decisión, se miraron la correlación entre las variables, por ejemplo, las variables con nombres parecidos como "CASH_ADVANCE", "CASH_ADVANCE_TRX", "CASH_ADVANCE_FREQUENCY" tenían una correlación positiva, así que solo se dejó solo una de estas. Para cada una de las pruebas se realizó una gráfica para cada uno de los clusters en la que se mostraba el número de clusters y la distorsión (escogiendo el número de cluster donde el cambio de la distorsión es muy grande. Adicional a esto, también se realiza otra gráfica en la que se muestra como varia el coeficiente silueta dependiendo el número de cluster s. Con estas dos gráficas se obtiene el hiperparámetro n_clusters.

Después de realizar esto, optamos por dejar la combinación que mayor coeficiente silueta presentaba. Esta es la prueba numero 3 con un coeficiente de 0.57 y con las siguientes columnas seleccionadas: ['PURCHASES', "CREDIT_LIMIT", "TENURE"].

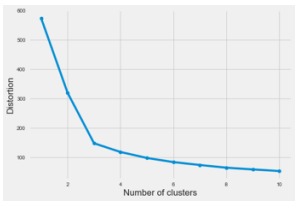


Ilustración 1. Número de clusters vs Distorsión

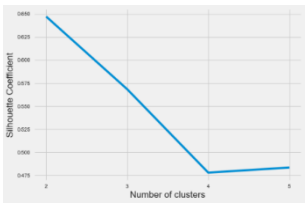


Ilustración 2. Número de clusters vs coeficiente siluera

Con esta selección, podemos graficar el siguiente resultado:

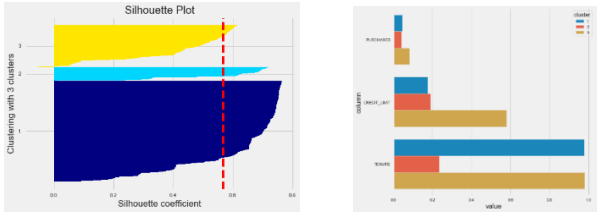


Ilustración 3. Grafica de clusters y el coeficiente de silueta

Algoritmo Affinity Propagation (Encargado: Alvaro Plata):

Escogimos este algoritmo para observar el comportamiento de un algoritmo que no requiere indicar el número de clusters necesarios en el modelo y poder compararlos con los otros dos algoritmos que sí requieren este parámetro.

Proceso:

1. Seleccionamos las columnas de la data set con las cuales crearemos el modelo. Para esto, creamos grupos con las columnas que estaban correlacionadas, a partir de la información del gráfico creado por Pandas Profiling. De cada uno de estos grupos escogimos una columna que estará incluida en la creación del modelo. Estas columnas fueron **BALANCE_FREQUENCY, CASH_ADVANCE_TRX, PURCHASES_TRX, CREDIT_LIMIT, PAYMENTS y TENURE**
2. Debido a que en la documentación se indica que este algoritmo es sensible a las escalas de las variables, escalamos los valores de las columnas escogidas utilizando la función MinMaxScaler() de la librería sklearn.preprocessing.
3. El módulo AffinityPropagation de ScikitLearn requiere el parámetro "preference" que tiene incidencia en la cantidad de clusters que serán definidos por el modelo. Debido a pruebas iniciales en las que evidenciamos que la creación de un modelo con una configuración de columnas y preferencia específicas toma varios minutos, decidimos utilizar un acercamiento "Divide and conquer" para determinar el valor adecuado para construir el modelo. Es decir, el valor que nos permitiera crear un modelo con una cantidad de clusters adecuados para el caso de negocio y un **valor** aceptable del coeficiente silueta

   Para esto, decidimos comenzar con un rango entre -100 y -1 (El valor de preferencia debe ser un número negativo), donde vimos que con un valor de **-1 obteníamos más de 180 clusters y con -**100 obteníamos 1 cluster. Probamos Probamo preferencia de -50, donde obtuvimos 11 clusters, lo que disminuye nuestro rango de búsqueda, pero aún eran demasiados clusters para el propósito del negocio. Continuamos con este proceso hasta llegar al valor de -37, con el que obtuvimos 7 clusters, ya que, con un valor menor, desde -38 en adelante, obteníamos únicamente 1 cluster.

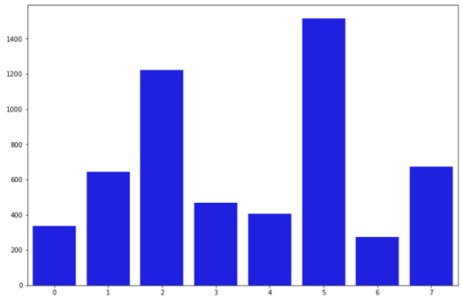


Ilustración 4: Clusters creados por el modelo de Affinity Propagation con la cantidad de clientes que pertenecen a cada cluster

4. Para la validación del modelo, creamos una gráfica donde se pudiera observar el coeficiente silueta y la calidad del modelo. Observamos que el coeficiente silueta se encuentra entre 0.2 y 0.3, por lo que no se considera el modelo más confiable para la clasificación de los clientes de BancAlpes.

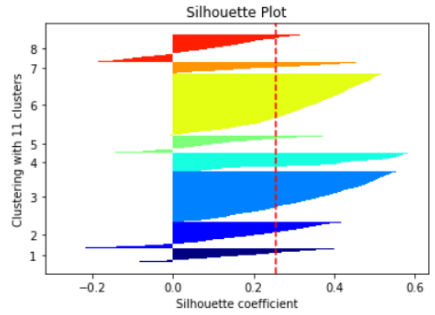


Ilustración 5: Gráfica del coeficiente silueta para el modelo creado por el algoritmo Affinity Propagation

Algoritmo Hierarchical Clustering (Encargada: María Sofía Álvarez):

El tercer algoritmo que utilizamos fue el clustering jerárquico. Lo anterior pues, de acuerdo con la literatura, es uno de los que mejor se comporta para datasets pequeños (consideramos pequeño un dataset de aproximadamente 5000 datos, como el que se tenía en este laboratorio) [1,2].

El clustering jerárquico es un método de análisis que busca construir una jerarquía de clústeres (agrupando objetos similares); es decir, construye estructuras tipo árbol basadas en la jerarquía. Existen dos tipos de clustering jerárquico: aglomerativo y divisivo. Debido a que el aglomerativo es el que se encuentra implementado en sci-kit learn, este es el que se usó en este laboratorio [3].

El clustering jerárquico aglomerativo, también conocido como aproximación ascendente, trata cada fila de datos como un clúster singletón (i.e. un clúster de un solo dato) y, sucesivamente, aglomera pares de clústeres hasta que todos son unidos en un único clúster de datos [5]. En la siguiente imagen, podemos ver un ejemplo de este tipo de algoritmo:

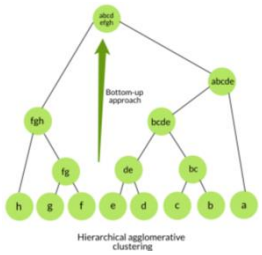


Ilustración 46. Funcionamiento del algoritmo de aglomeración jerárquica. Los datos empiezan como clústeres individuales hasta convertirse en un único clúster.

Proceso:

Este modelo, a diferencia de los anteriores, buscó construirse con todas las columnas del conjunto de datos. Esto porque al funcionar bien para conjuntos de datos pequeños en un tiempo computacional corto, podíamos ver la incidencia de utilizar varias columnas de datos. Se propuso que, como se mencionó previamente, en una versión más avanzada de este proyecto se realizara un análisis por PCA para reducir la Dimensionalidad. En esta etapa no se realizó PCA con el fin de obtener información limpia de cada una de las columnas (y no combinaciones lineales de ellas, que serían los vectores obtenidos del PCA).

A diferencia de K-Medias, una de las mejores formas de encontrar el valor adecuado para un clúster en el modelo jerárquico aglomerativo es usando el método de los dendrogramas. Esta fue la primera alternativa usada para encontrar un número de clústeres preliminar para el modelo base. Los dendrogramas son estructuras en forma de árboles que permiten visualizar el historial de agrupamientos y encontrar el número óptimo de clústeres. Analíticamente, la idea es mirar cual es la distancia vertical más larga que NO sea atravesada por una línea horizontal. Para el conjunto de datos en cuestión, el dendrograma realizado se presenta a continuación:

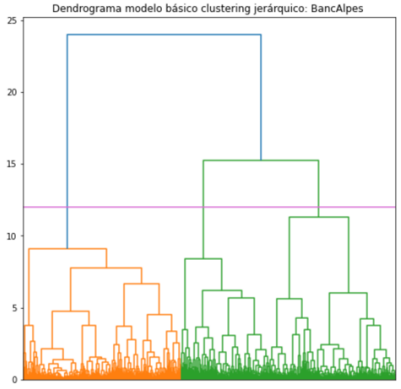


Ilustración 5. Dendrograma para los datos de BancAlpes.

Note que la línea morada intersecta tres líneas tal que, en este caso, el número óptimo de clústeres sería 3. Esto se confirmó posteriormente haciendo uso del coeficiente silueta. Con ello, se construyó un modelo básico con 3 clústeres, cuya distribución de los datos puede apreciarse en la ilustración 5.

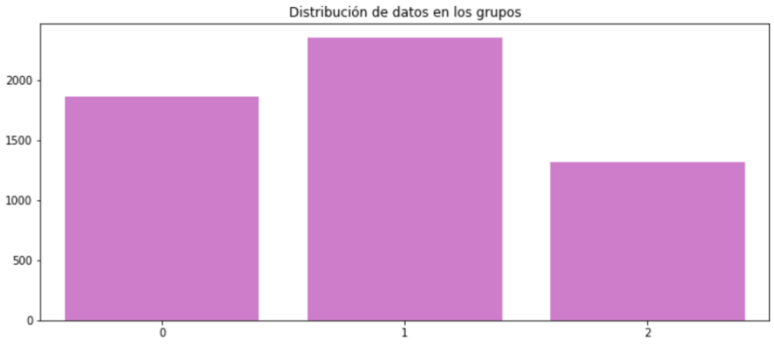


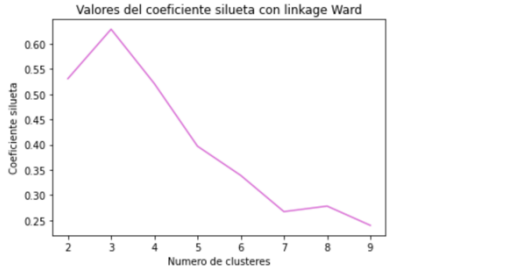
Ilustración 6. Distribución de datos en los clústeres definidos.

Vemos que la cantidad de datos en los clústeres es bastante similar: no hay clúster con muchos datos y otro con pocos, lo cual es un buen indicio de este primer modelo. Ahora, se decidió que era más conveniente hacer una búsqueda de hiperparámetros antes de analizar todo el algoritmo. Para ello, se encontró lo siguiente sobre los hiperparámetros de la aglomeración en scikit-learn:

- n_clusters: el número de clústeres a encontrar. En el caso del modelo inicial, usamos el obtenido con el método del dendrograma. En el modelo con hiperparámetros, se probaron valores de entre 2 y 9 clústeres.
- affinity: La forma de calcular las distancias. Puede ser manhattan (i.e. valor absoluto), l1 (regresión Lasso), l2 (regresión Ridge) o euclideana. Sólo se acepta distancia euclidiana si el linkage fue Ward. *Nota:* Las regresiones Lasso y Ridge fueron explicadas en el laboratorio pasado. Para la búsqueda de hiperparámetros, se utilizaron l1 (porque es la más robusta de las lineales), coseno (novedosa para nosotros) y euclidiana (la que está por defecto y funciona con Ward, que se utilizará más adelante).
- memory: Guarda en una caché las combinaciones del árbol. No es relevante para el modelo y no afecta en la construcción de hiperparámetros. Por defecto, no se guarda.
- connectivity: para cada muestra, define sus muestras vecinas. Por defecto no hay, y debe ser definida por el usuario. Por lo tanto, no se tendrá en cuenta en la búsqueda de hiperparámetros.
- compute_full_tree: Parar la construcción temprana del árbol en n_clusters. El valor por defecto es auto, tal que el algoritmo decida cuando es hora de parar de construir el árbol. Este hiperparámetro lo dejaremos así, tanto para el modelo inicial, como para el de búsqueda de hiperparámetros.
- linkage: Criterio de unión de clusteres. Son los mismos definidos previamente para el dendrograma. En la búsqueda de hiperparámetros, se usaron las dos más robustas de acuerdo con la literatura: average y Ward.
- distance_threshold: La distancia a partir de la cual no se unirán clusteres. Es un método alternativo para cuando no se sabe el número de clústeres. Por defecto, es None, pues es difícil y dependiente del problema encontrar el valor para el cual no deben unirse más los clusteres. Lo dejaremos de esta forma tanto para el modelo inicial como para el de búsqueda de hiperparámetros.
- compute_distances: Por defecto es False, pues puede incurrir en un gran gasto de memoria. Crea la visualización de dendrogramas. No lo usaremos pues los construimos con scipy.

Una vez considerados los hiperparámetros a usar (n_clusters entre 2 y 9), linkage (Ward o Average) y affinity (l1, coseno y euclidiana), se procedió a encontrar el coeficiente silueta en todos los casos descritos. Para ver la información de todos los *runs,* remítase al notebook adjunto a este laboratorio. Para este caso, el mejor coeficiente silueta que se obtuvo fue de 0.629, el cual es bastante bueno y es el más elevado de todos los considerados. Este funcionó para dos casos: 3 clústeres, Ward linkage y afinidad euclidiana, y 3 clústeres, Average linkage y afinidad euclideana. Notamos que el primer caso es el mismo del modelo inicial considerado y, además, coincide con el número de clústeres hallado con el método del dendrograma. Por tanto, nos quedamos con este modelo.

Note cómo el coeficiente silueta es máximo para 3 clústeres.




```
El valor del mejor coeficiente silueta es: 0.6286932706533063
```

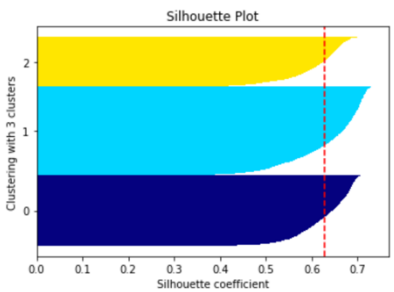


Ilustración 7. Coeficiente silueta para diferentes valores de clústeres usando Ward y afinidad euclidiana. Note que es máximo en 3.

De esta última gráfica también es posible notar que el clúster 2 es el de menor tamaño, seguido del 0. El 1 es el mayor. Entre más alto sea el valor del coeficiente silueta, más alejados están los clústeres entre sí. En nuestro caso, un valor de 60% es bastante bueno para separar tres grupos de clientes de BancAlpes. Es importante aclarar que, aunque se sale del enfoque del curso y del laboratorio, una forma de obtener mejores resultados sería aplicando un PCA sobre los datos. No hicimos esto para obtener mejor información sobre las variables "limpias" del dataset.

Vemos que el coeficiente silueta de este algoritmo fue el mejor de todos los tres algoritmos implementados (K-Means, 0.57 y Affinity Propagation, 0.2). Por lo tanto, elegimos este modelo como el recomendado para BancAlpes, dado que es el que mejor segrega los clientes.

Una vez elegido el algoritmo cualitativamente, se procedió a hacer un análisis cualitativo. Para ello, se siguieron dos enfoques. Se agrupó cada variable de acuerdo a la cantidad de datos en cada clúster definido. Por otro lado, se graficó la media de cada clúster para cada variable. En la primera aproximación, se obtuvieron las siguientes visualizaciones:

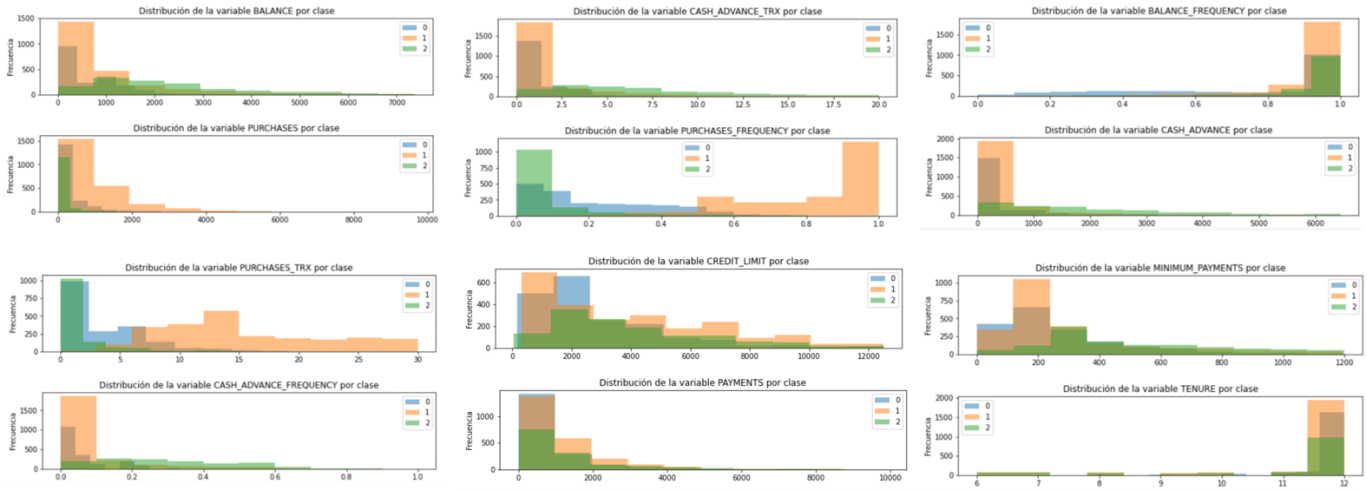


Ilustración 8. Distribución de frecuencias para cada variable según el clúster definido.

A grandes rasgos, podemos ver que una de las variables que mayor distinción entre clústeres genera es PURCHASES_FREQUENCY. Parece ser que los de la clase 1 compran mucho más que los de la clase 0 y 1. Lo cual es fundamental. Los *insights* de las otras variables se encuentran a continuación:

- **BALANCE:** Del histograma obtenido podemos ver que la clase 2 tiene un balance más distribuido en todo el rango de valores. Podemos concluir que los clientes de los grupos 0 y 1 consumen más o tienen cupos más pequeños, pues tienen balances más pequeños que los del grupo 2 (quienes, en general, están más distribuidos).
- **PURCHASES:** De esta variable la información más relevante que podemos obtener es que casi todos los clientes tipo 1 realizan 1000 compras en su cuenta. Son, además, los que más compras realizan. Los clientes tipo 2 realizan todos menos de 100 compras. Esto tiene sentido pues los clientes tipo 2 son los que mayor balance distribuido tienen. Asimismo, vemos que los clientes tipo 0 también realizan muy pocas compras. Esto se debe a que tienen un balance pequeño.
- **BALANCE_FREQUENCY:** La distribución de balance es más alta para los clientes tipo 1. Esto significa que su balance es el que más se actualiza. Tiene sentido, al ser los que más compras hacen.
- **CASH_ADVANCE:** Esta variable nos dice de cuanto es el valor de los pagos realizados por los clientes. Los lientes tipo 0 son lo que menos hacen, seguidos de los tipo 1. Los clientes tipo dos están distribuidos a lo largo de todo el rango, pero, como son los de mayor poder adquisitivo al parecer, pueden hacer pagos de grandes sumas monetarias.
- **CASH_ADVANCE_TRX:** Nos da la misma información de CASH_ADVANCE. Esto justifica el hecho de querer hacer un PCA a futuro.
- **PURCHASES_FREQUENCY:** Esta variable da muy buena información de clusterización. Vemos que los clientes tipo 1 son los que compran más a menudo; mientras que, los tipos 0 y 2, son los que menos compras realizan.
- **PURCHASES_TRX:** Los clientes tipo 1 hacen, en promedio, una mayor cantidad de transacciones de compras. Les siguen los clientes tipo 0 y, finalmente, los tipo 2. Estos son los que menos compras realizan.
- **CASH_ADVANCE_FREQUENCY:** Nos da información similar a CASH_ADVANCE y CASH_ADVANCE_TRX.
- **CREDIT_LIMIT:** Vemos que los usuarios tipo 1 y tipo 2 tienen una distribución similar de límite de crédito en la tarjeta. Los de tipo 0, por el contrario, tienen poquito cupo en general.

- **PAYMENTS:** No da información muy diciente sobre el clustering realizado.
- **MINIMUM_PAYMENTS:** No da información muy diciente sobre el clustering realizado.
- **TENURE:** No da información muy diciente sobre el clustering realizado. Casi todos los usuarios han tenido la tarjeta por 12 meses.

De esta parte concluimos que, para una próxima iteración, es posible eliminar variables como CASH_ADVANCE_TRX. Esto, pues ellas no aportan mucho y están altamente correlacionadas con con CASH_ADVANCE. Asimismo, vemos que variables como PAYMENTS, MINIMUM_PAYMENTS y TENURE no dan información relevante y pueden ser eliminadas del clustering para un mejor desempeño.

Finalmente, se hizo una agrupación por la media también:

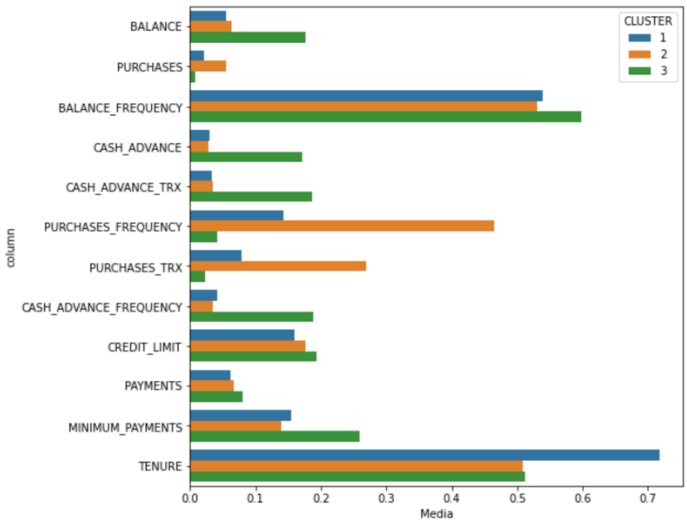


Ilustración 9. Distribución de frecuencias para la media de cada variable según el clúster definido. En esta imagen, todos los clusteres están corridos por 1 (i.e. cluster 0= cluster 1, cluster 1=cluster 2, cluster 2=cluster 3).

Vemos que el cluster con mayor balance (i.e. con mayor cupo disponible) es el 3, y es el que menos compras realiza. Los del tipo 2 tienen un balance más bajo, pero realizan más compras considerablemente. Los del grupo 1 tienen un balance más bajo inclusive, pero compran menos.

Los del grupo 3 son los que hace pagos más grandes por adelantado (CASH_ADVANCE). Vemos que, en general, toda la información es concordante con lo descrito previamente.

Conclusión y recomendaciones

De acuerdo con lo visto previamente y el algoritmo de aglomeración jerárquica, es posible concluir que los datos se dividen más óptimamente en tres clústeres, los cuales pueden definirse como 3 *personas* diferentes a las cuales dirigir la campaña de marketing:

- Grupo 0 (azul): Son individuos cuyo poder adquisitivo es más bajo, ya que tienen menos cupo (balance) en su tarjeta. Por lo mismo, realizan un número reducido de compras (menos que el grupo 1, pero más que el 2) y realizan pagos de dinero al banco en cuotas con montos bajos. Es el grupo con menor límite crediticio. La campaña de marketing para este grupo podría estar encaminada a universitarios o gente joven que esté empezando su vida crediticia para que haga compras, desarrolle un historial crediticio y después sea promovida a otra categoría. Con ello, podrían empezar a obtener beneficios desde una corta edad. Incluso, si mejoran mucho, podría aumentárseles el cupo de su tarjeta.

- Grupo 1 (naranja): Es el grupo más grande de todos los definidos en el problema. Son los que más compras realizan, tienen un balance bajo (un poco mayor que los del grupo 0, pero mucho más bajo que los del 1 por la cantidad de compras que realizan). Su frecuencia de compras es la más elevada. La campaña de marketing debe estar orientada a incrementar el número de compras que ellos realizan, o mantenerlo. Podría orientarse a acumular puntos o millas al usar la tarjeta de crédito, para que la sigan usando más. Estas personas por lo general son de edad media, ya llevan una vida crediticia y tienen más beneficios que los del grupo 0. Les interesa mejorarla por lo general, compran bastante y debe incentivarse a que compren más. Podrían motivarse a obtener un cupo mayor.

- Grupo 2 (verde): Son los que mejor balance tienen, por dos razones: primero, no realizan muchas compras (debe incentivarse esto) pero, también, porque son los de mayor poder adquisitivo (en general, son los que realizan los pagos más elevados de tarjetas de los tres grupos). Estas personas, por lo general, son de edad un poco más madura y en general prefieren no usar las tarjetas de crédito y prefieren siempre tener sus cuentas saldadas con los bancos. A pesar de que son los que menos compras realizan, sus compras suelen ser las más elevadas (por eso, CASH_ADVANCE) es el más alto. Para este grupo, debemos buscar que la gente haga más compras más costosas (por sumas elevadas pues su balance se los permite). Para ello, también podría implementarse beneficios, pero esta vez, algo que los motive más a comprar: como convenios con otras empresas, clubes o patrocinios que sean mejores.

Bibliografía

[1] Géron (2019). Hands-On Machine Learning with Scikit-Learn, Keras and Tensorflow.

[2] https://www.sciencedirect.com/topics/mathematics/hierarchical-clustering

[3] https://stats.stackexchange.com/questions/217875/clustering-very-small-datasets#:~:text=For%20tiny%20data%20sets%2C%20hierarchical,and%20how%20many%20clusters%20exist.

[4] https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering

[5] https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/

[6] https://medium.com/@sametgirgin/hierarchical-clustering-model-in-5-steps-with-python-6c45087d4318

[7] https://arxiv.org/pdf/1503.00900.pdf#:~:text=Normalization%20is%20used%20to%20eliminate,in%20the%20differences%5B3%5D.

[8] https://journalofinequalitiesandapplications.springeropen.com/articles/10.1186/1029-242X-2013-203#:~:text=Hands%20and%20Everitt%20%5B18%5D%20compared,overall%20than%20other%20hierarchical%20methods.

[9] Hands S, Everitt B: A Monte Carlo study of the recovery of cluster structure in binary data by hierarchical clustering techniques. Multivar. Behav. Res. 1987, 22: 235–243. 10.1207/s15327906mbr2202_6