Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 02/03/2022
Practica No.4 Clustering Jerarquico

Objetivo.

Obtener clústeres de casos de usuarios, con características similares, evaluados para la adquisición de una casa a través de un crédito hipotecario con tasa fija a 30 años.

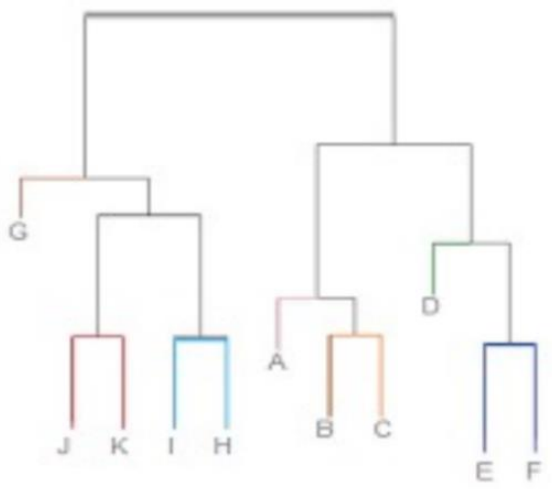
Características.

La fuente de datos tiene las siguientes características.

- ingresos: son ingresos mensuales de 1 o 2 personas, si están casados.
- gastos_comunes: son gastos mensuales de 1 o 2 personas, si están casados.
- pago_coche
- gastos_otros
- ahorros
- vivienda: valor de la vivienda.
- estado_civil: 0-soltero, 1-casado, 2-divorciado
- hijos: cantidad de hijos menores (no trabajan).
- trabajo: 0-sin trabajo, 1-autonomo, 2-asalariado, 3-empresario, 4-autonomos, 5-asalariados, 6-autonomo y asalariado, 7-empresario y autonomo, 8-empresarios o empresario y autónomo
- comprar: 0-alquilar, 1-comprar casa a través de crédito hipotecario con tasa fija a 30 años.

El clustering lo que busca es hacer una segmentación de los registros en el conjunto de datos, con esto se descubren una serie de patrones que normalmente estarían ocultos los cuales relacionan a cada uno de ellos consiguiendo agruparlos, sin embargo, no se ofrece una descripción clara y es tarea de nosotros darles una interpretación adecuada.

En particular el clustering jerárquico organiza los datos en una estructura de tipo árbol de forma recursiva de tal manera que en cada nivel del árbol se tiene cierto número de clusters no definido e incrementa conforme bajamos en el árbol.



Desarrollo.

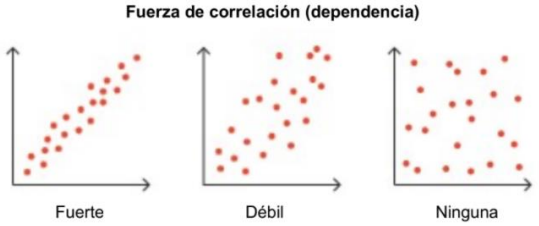
Como primer paso es necesario importar las librerías necesarias, `numpy` para el manejo de matrices, pandas para la manipulación/análisis de los datos, `matplotlib` y `seaborn` para la visualización de los datos con diferentes tipos de gráficas, algunos módulos de `sklearn` como `StandardScaler` y `MinMaxScaler` para los procesos de estandarización y normalización de la matriz de datos, de `scipy hierarchy` para la implementación del modelo para el clustering jerárquico y por ultimo de `AgglomerativeClustering` para etiquetar a cada registro.

Seguido de esto ahora pasaremos a la etapa de la lectura de los datos para los usuarios candidatos al préstamo hipotecario usando pandas para crear un `dataframe` obteniendo nuestra matriz con 202 registros donde cada uno de estos tiene 10 variables o atributos que los caracterizan y a partir de esto podemos comenzar a analizar el conjunto de datos observando que todas las variables del conjunto son numéricas, también podemos contabilizar cuales son las personas que ocuparan el crédito para alquilar (135) o comprar la casa (65) con ese dinero.

Parte de este análisis es observar la relación entre las variables con graficas de densidad y dispersión para esto mismo, tenemos que utilizar una de estas variables para segmentar los registros la cual será comprar y generar esta visualización con el método `pairplot` de la libreria `seaborn` asignándole un matiz a la variable comprar para poder diferenciarlos.

```
sns.pairplot(Hipoteca, hue='comprar')       #Densidad y gráficas de dispersión
plt.show()
```

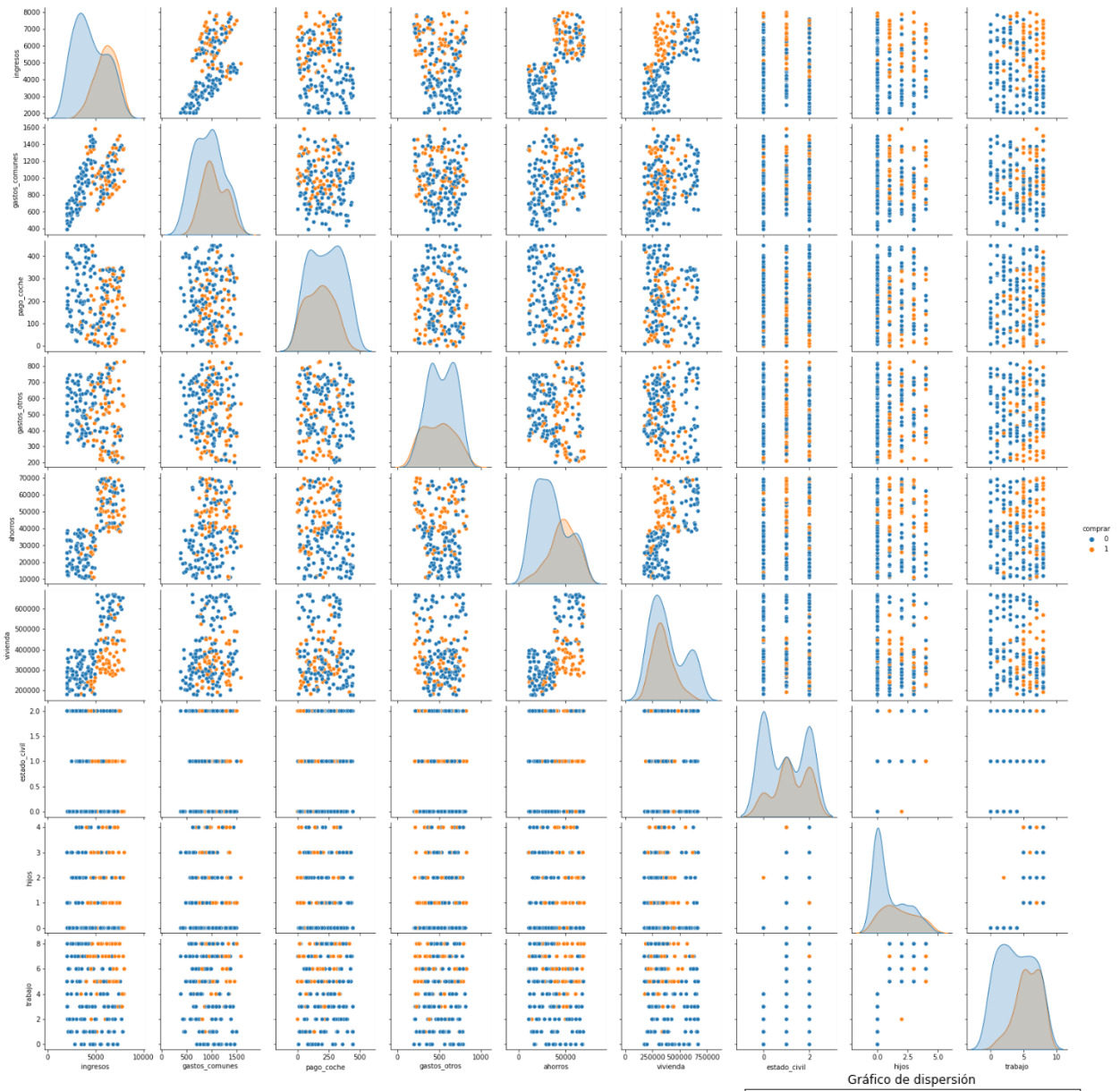
Con esto se genera la siguiente visualización donde podemos observar una matriz simétrica la cual resume cual es la dispersión de cada una de sus variables con respecto a las demás, es decir, ingresos con gastos comunes, ingresos con pago del coche, etc. además de estar conformada en su diagonal principal por la la distribución de las personas que compraran y las personas que alquilaran la casa con el crédito hipotecario. Con la imagen de abajo nos podemos dar una idea de cómo es que se ven este tipo de gráficos para la correlación entre las variables del modelo pudiéndolas catalogar en 3 principales rubros; Fuerte (1 > índice > 0.67), Débil (0.66 > índice > 0.34), Ninguna(0 > índice > 0.33) aplicando a negativos por igual.

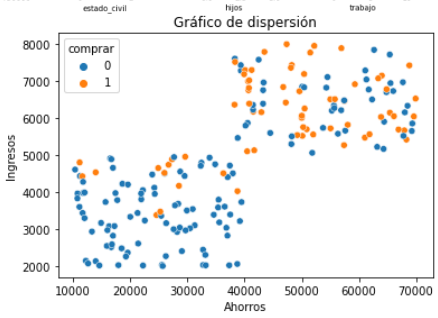


Podemos apreciar que las únicas variables que pudieran presentar una correlación cercana a la débil podrían ser los ahorros y los ingresos por lo cual pudieran ser consideradas en una primera instancia para eliminarse del modelo.

Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 02/03/2022
Practica No.4 Clustering Jerarquico





Pudiéramos incluso hacer mas grande este grafico de dispersión con ayuda de `seaborn` y `matplotlib` para observar mejor la correlación, pero además podría resultarnos útil calcular una matriz de correlaciones la cual detalla con un índice del -1 al 1 las relaciones entre variables numéricas siendo las variables más cercanas a 1 las que se relacionan.

Esta matriz de correlaciones la podemos calcular directo del `dataframe` que ya se genero previamente con ayuda del método `corr` pasándole como parámetro el método con el que queremos que calcule estos índices, en este caso utilizamos `pearson` pero también pudiéramos utilizar los métodos de `K¿kendall`, `spearman`, etc.

```
CorrHipoteca = Hipoteca.corr(method='pearson')
CorrHipoteca
```

	ingresos	gastos_comunes	pago_coche	gastos_otros	ahorros	vivienda	estado_civil	hijos	trabajo	comprar
ingresos	1.000000	0.560211	-0.109780	-0.124105	0.712889	0.614721	-0.042556	-0.024483	-0.038852	0.467123
gastos_comunes	0.560211	1.000000	-0.054400	-0.099881	0.209414	0.204781	-0.057152	-0.072321	-0.079095	0.200191
pago_coche	-0.109780	-0.054400	1.000000	0.010602	-0.193299	-0.094631	0.052239	-0.044858	0.018946	-0.196468
gastos_otros	-0.124105	-0.099881	0.010602	1.000000	-0.064384	-0.054577	-0.020226	0.124845	0.047313	-0.110330
ahorros	0.712889	0.209414	-0.193299	-0.064384	1.000000	0.605836	-0.063039	0.001445	-0.023829	0.340778
vivienda	0.614721	0.204781	-0.094631	-0.054577	0.605836	1.000000	-0.113420	-0.141924	-0.211790	-0.146092
estado_civil	-0.042556	-0.057152	0.052239	-0.020226	-0.063039	-0.113420	1.000000	0.507609	0.589512	0.142799
hijos	-0.024483	-0.072321	-0.044858	0.124845	0.001445	-0.141924	0.507609	1.000000	0.699916	0.272883
trabajo	-0.038852	-0.079095	0.018946	0.047313	-0.023829	-0.211790	0.589512	0.699916	1.000000	0.341537
comprar	0.467123	0.200191	-0.196468	-0.110330	0.340778	-0.146092	0.142799	0.272883	0.341537	1.000000

Si bien esto nos muestra los índices de correlación de cada una de las variables resulta también un poco complicada de leer por la estructura en la que presenta los datos dificultándose el identificar cuales son las variables con mayor correlación. Una primera solución que se pudiera implementar es mostrar el top 10 o top 5 de los valores mas altos o cercanos a 1 para cada una de las variables, por ejemplo, para la variable de ingresos tenemos que el top 10 de valores, es decir, cada una de sus correlaciones ordenadas de mayor a menor es son las siguientes:

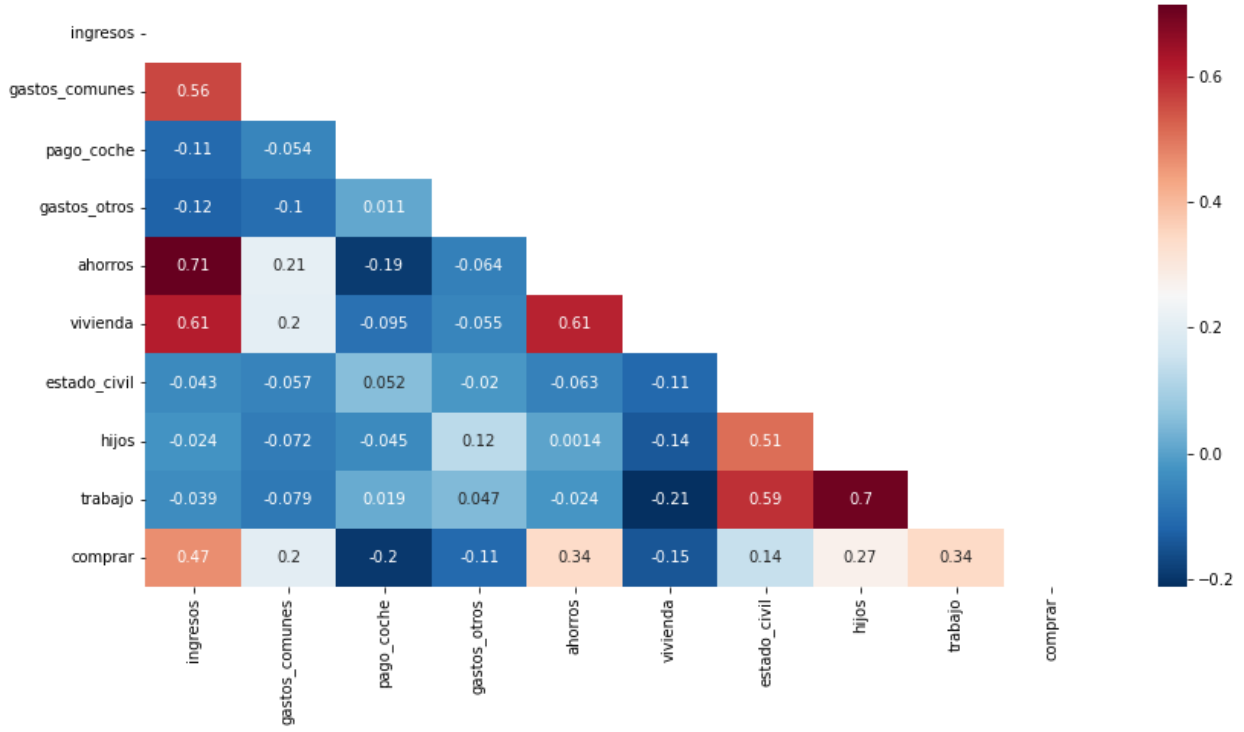
```
print(CorrHipoteca['ingresos'].sort_values(ascending=False)[:10], '\n')   #Top 10
```

```
ingresos          1.000000
ahorros           0.712889
vivienda          0.614721
gastos_comunes    0.560211
comprar           0.467123
hijos            -0.024483
trabajo          -0.038852
estado_civil     -0.042556
pago_coche       -0.109780
gastos_otros     -0.124105
```

Lo que nos interesa para poder eliminar variables del modelo es que haya variables que estén fuertemente correlacionadas, con esto en consideración dentro del conjunto tenemos que ingresos con ahorros cumple con la condición para este tipo de correlación sin embargo es muy importante considerar cuales son las variables que nos pueden servir para nuestro análisis con el fin de evitar eliminar información que nos puede ser útil mas adelante. Así bien, si nos detenemos a pensar nos daremos cuenta de que ahorros nos puede servir más adelante para considerar si se otorgara un crédito o no ya que es el colchón que tiene el usuario para poder pagar en un momento de necesidad las mensualidades que se le presenten, así quien no tenga tantos ahorros no va a tener la misma consideración que alguien que si los tiene.

Otra técnica que pudiéramos utilizar para visualizar de mejor manera estas correlaciones es hacer un mapa de calor con la matriz, esta técnica lo que hace es que colorea con tonos más cálidos (Rojo, vino, naranja) los valores mas altos de la matriz y tonos más fríos (azul marino, azul fuerte, azul cielo) los índices mas bajos en la matriz. Bajo esta técnica entonces tendremos que prestar atención a cuáles son los tonos más fríos y cálidos en la matriz para identificar las variables correlacionadas entre si candidatas a ser eliminadas del modelo. La forma en la que se hace este mapa en Python es con la librería `Seaborn`.

```
plt.figure(figsize=(14,7))
MatrizInf = np.triu(CorrHipoteca)
sns.heatmap(CorrHipoteca, cmap='RdBu_r', annot=True, mask=MatrizInf)
plt.show()
```



A pesar de que *ahorros e ingresos* y *trabajo e hijos* representan una correlación fuerte, éstas se van a considerar ya que pueden representar información útil para otorgar el crédito como se mencionó anteriormente. Sin embargo, la variable *comprar* que se utilizó para una segmentación inicial no se considera útil ya que independientemente de si rentaran o compraran con el crédito todos siguen siendo candidatos a que se les otorgue dicho caretito por lo cual se decidió eliminar esta variable quedándonos con un conjunto final de datos de 202 registros por 9 variables.

Una vez terminada la etapa de exploración de los datos se comenzó con la implementación del algoritmo para lo cual el primer paso es transformar los datos a una escala menor con el proceso de estandarización (Media = 0 y Desviación estándar = 1) o normalización (Escala de valores entre 0 y 1) para mejorar el rendimiento del algoritmo de clustering jerárquico obteniendo las siguientes matrices.

Estandarización. (Media = 0, Desviación Estándar = 1 y utilizando StandardScaler.fit_transform).

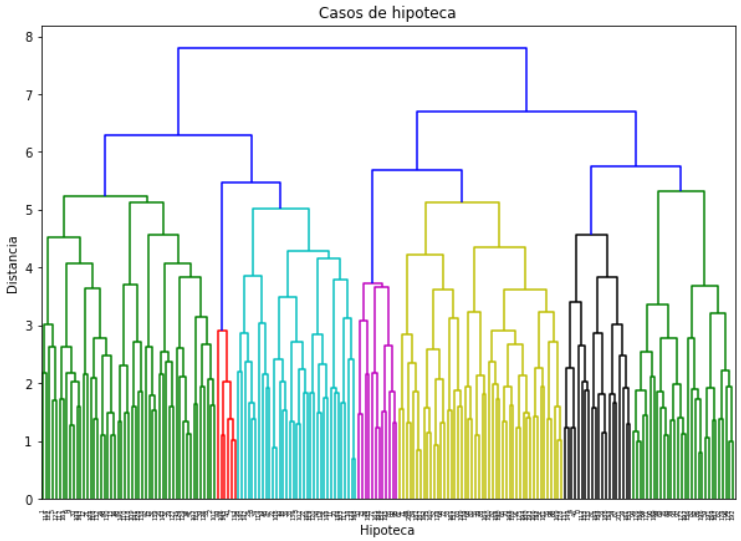
	0	1	2	3	4	5	6	7	8
0	0.620129	0.104689	-1.698954	0.504359	0.649475	0.195910	-1.227088	0.562374	-0.984420
1	1.063927	-0.101625	-0.712042	-0.515401	0.259224	1.937370	-0.029640	1.295273	0.596915
2	0.891173	0.226266	-0.912634	1.667244	1.080309	-0.379102	1.167809	-0.170526	1.387582
3	1.274209	1.128886	-1.578599	-1.559015	0.909604	2.114062	-1.227088	-0.903426	-0.589086
4	0.719611	-0.400042	0.090326	0.027279	0.159468	-0.179497	-1.227088	-0.903426	-0.589086
...	...	...	...	...	...	...	...	...	...
197	-0.671949	-1.037402	1.125381	-0.163554	-1.617963	-0.075199	-1.227088	-0.903426	-0.984420
198	-0.594508	0.215214	0.467439	-0.241079	-0.973876	-0.683130	1.167809	1.295273	1.387582
199	-1.057368	-0.061099	0.515581	1.005294	-0.183849	0.107880	-0.029640	1.295273	1.387582
200	-0.968013	-0.385305	1.261783	0.814462	-1.083273	0.026040	-0.029640	0.562374	0.201581
201	-0.578424	0.683102	-0.856468	-0.795686	-1.545397	-0.851037	-1.227088	-0.903426	-0.193753

Normalización. (Escala de valores entre 0 y 1 utilizando MinMaxScaler.fit_transform).

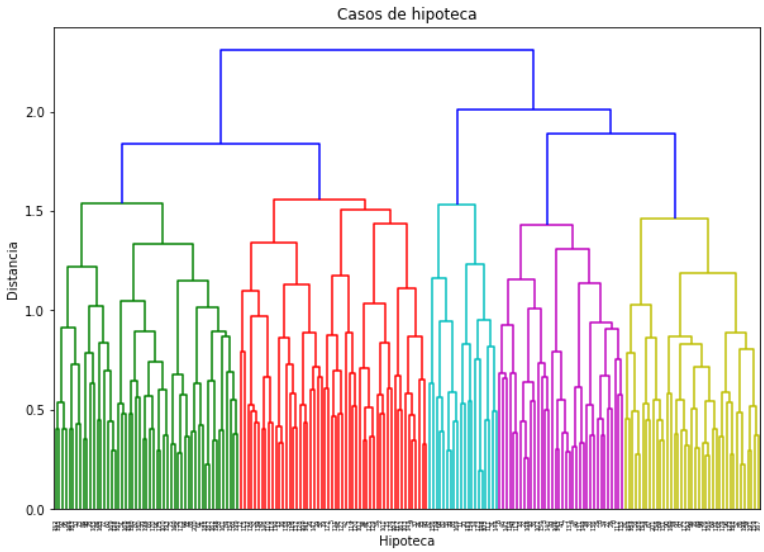
	0	1	2	3	4	5	6	7	8
0	0.668005	0.512906	0.000000	0.636364	0.665621	0.453251	0.0	0.50	0.250
1	0.792671	0.466278	0.274554	0.363636	0.552227	0.933785	0.5	0.75	0.750
2	0.744143	0.540383	0.218750	0.947368	0.790808	0.294584	1.0	0.25	1.000
3	0.851740	0.744380	0.033482	0.084530	0.741206	0.982541	0.0	0.00	0.375
4	0.695950	0.398834	0.497768	0.508772	0.523241	0.349662	0.0	0.00	0.375
...	...	...	...	...	...	...	...	...	...
197	0.305054	0.254788	0.785714	0.457735	0.006777	0.378442	0.0	0.00	0.250
198	0.326807	0.537885	0.602679	0.437002	0.193928	0.210691	1.0	0.75	1.000
199	0.196787	0.475437	0.616071	0.770335	0.423484	0.428961	0.5	0.75	1.000
200	0.221888	0.402165	0.823661	0.719298	0.162140	0.406378	0.5	0.50	0.625
201	0.331325	0.643630	0.234375	0.288676	0.027862	0.164359	0.0	0.00	0.500

Con estas matrices estandarizadas y normalizadas obtenidas a partir del conjunto de datos inicial podemos implementar el algoritmo de forma visual para darnos una idea de cómo están distribuidos los clusters con ayuda del módulo `hierarchy` de `scipy` pasándole como parámetros la matriz, el método que será `complete` en este caso y la métrica para calcular las matrices de distancia las cuales se pueden tomar de las métricas trabajadas la practica anterior; `euclidean`, `chebyshev` y `cityblock` graficándolas además con ayuda de `matplotlib` obteniendo los siguientes resultados para cada una de ellas.

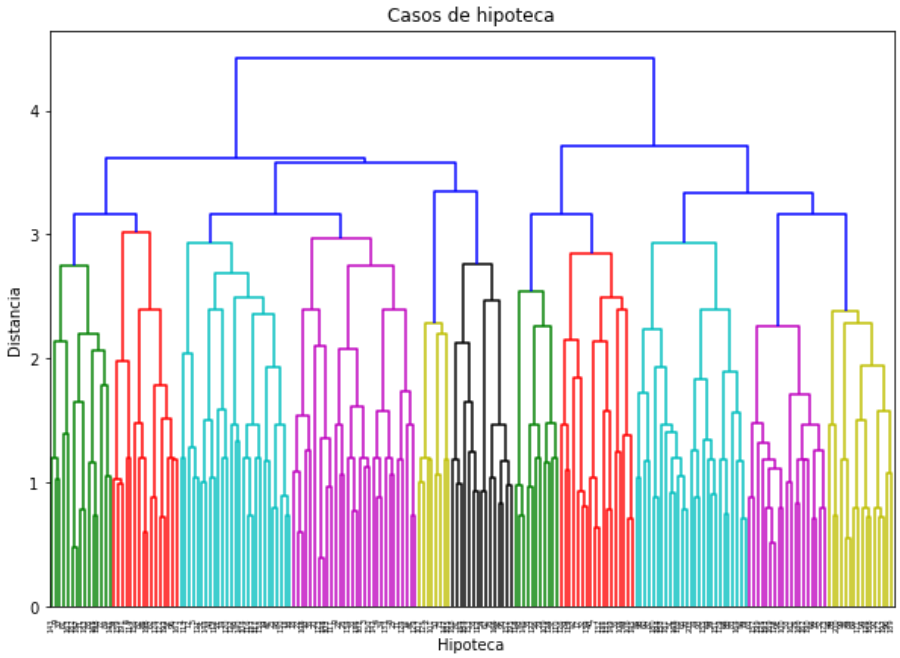
Métrica de distancia: `euclidean`, Matriz: `Estandarizada`, Resultado: `7 clústeres`.



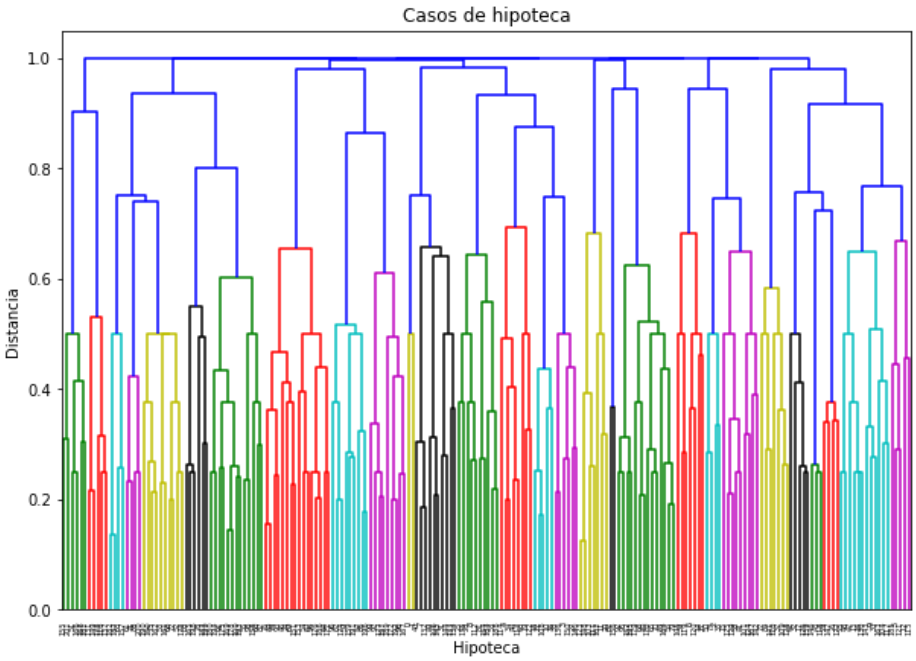
Métrica de distancia: `euclidean`, Matriz: `Normalizada`, Resultado: `5 clústeres`.



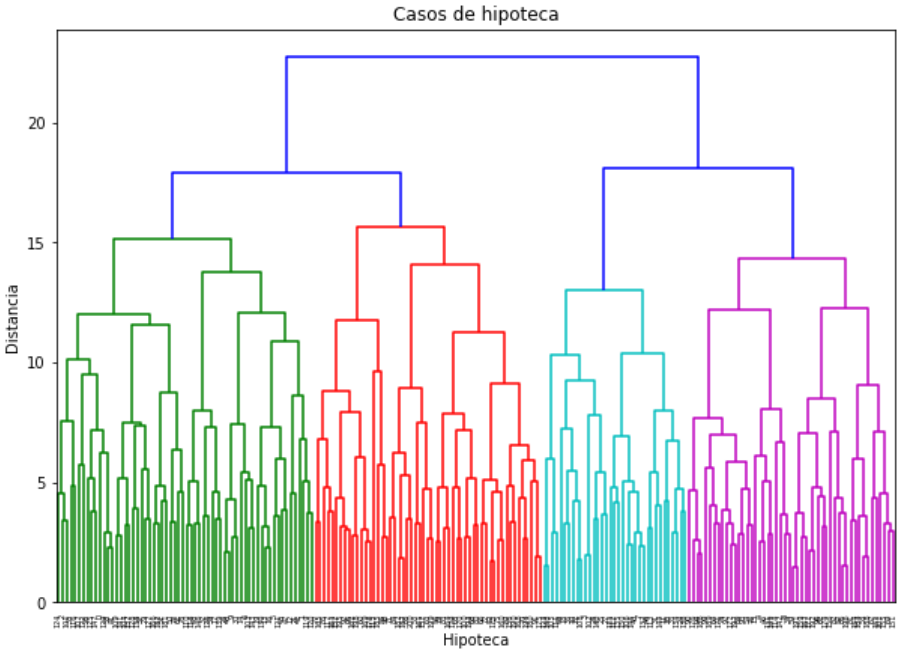
Métrica de distancia: `chebyshev`, Matriz: `Estandarizada`, Resultado: `11 clústeres`.



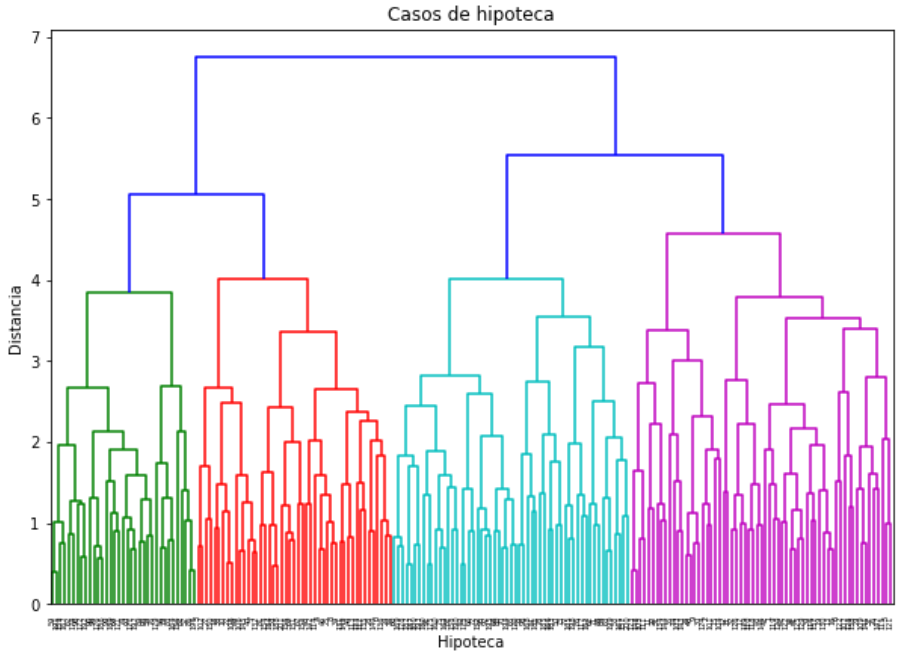
Métrica de distancia: `chebyshev`, Matriz: `Normalizada`, Resultado: `28 clústeres`.



Métrica de distancia: `cityblock`, Matriz: `Estandarizada`, Resultado: `4 clústeres`.



Métrica de distancia: `cityblock`, Matriz: `Normalizada`, Resultado: `4 clústeres`.



De estas pruebas podemos observar que la cantidad de clústeres formados en el árbol dependerá directamente de 2 cosas, la primera y mas importante es la métrica de distancias que decidamos utilizar; euclidiana, chebyshev o manhattan. La segunda es el proceso que utilicemos para el escalado de los datos, es decir, estandarizarlos o normalizarlos.

Como este tipo de algoritmos es parte del aprendizaje no supervisado se vuelve algo impredecible el saber cuál va a ser la cantidad de clústeres que nos entregue cierta configuración, por ejemplo, con distancias `euclidianas` tenemos más clústeres con la matriz de datos estandarizada respecto a la matriz normalizada, 7 en comparación con 5. Para las distancias de `chebyshev` ocurre lo contrario pues tenemos muchos más clústeres con la matriz normalizada en comparación con la estandarizada 28 en comparación con 11 y observamos que esta es la métrica de distancias que más clústeres nos entrega para este conjunto de datos. Por último, para las distancias `manhattan` ambas configuraciones de matrices iniciales, estandarizada y normalizada, nos entregan 4 clústeres por igual además de que podemos resaltar que esta es la métrica que menos clústeres nos devuelve como resultado.

En una segunda instancia, una vez que tenemos el numero de clusters podemos implementar el algoritmo de cauterización jerárquico con el modulo AgglomerativeClustering de la libreria sklearn con el objetivo de ahora etiquetar con su clúster a cada uno de los registros del conjunto de datos, en este caso 202 registros el cual recibe como parámetros el numero de clusters que queremos, el método de enlace que determina que distancia usar entre los conjuntos y la afinidad que calcula la métrica de distancia a utilizar.

```
MJerarquico = AgglomerativeClustering(n_clusters=7, linkage='complete', affinity='euclidean')
MJerarquico.fit_predict(MEstandarizada)
MJerarquico.labels_
```

Esta ultima línea nos devuelve las etiquetas de los clusters para cada uno de los registros los cuales podemos añadir a nuestro dataframe de pandas como una nueva columna de la siguiente manera.

```
Hipoteca = Hipoteca.drop(columns=['comprar'])
Hipoteca['clusterH'] = MJerarquico.labels_
Hipoteca
```

	ingresos	gastos_comunes	pago_coche	gastos_otros	ahorros	vivienda	estado_civil	hijos	trabajo	clusterH
0	6000	1000	0	600	50000	400000	0	2	2	4
1	6745	944	123	429	43240	636897	1	3	6	1
2	6455	1033	98	795	57463	321779	2	1	8	1
3	7098	1278	15	254	54506	660933	0	0	3	2
4	6167	863	223	520	41512	348932	0	0	3	4
...	...	...	...	...	...	...	...	...	...	...
197	3831	690	352	488	10723	363120	0	0	2	0
198	3961	1030	270	475	21880	280421	2	3	8	5
199	3184	955	276	684	35565	388025	1	3	8	3
200	3334	867	369	652	19985	376892	1	2	5	3
201	3988	1157	105	382	11980	257580	0	0	4	4

clusterH	
0	30
1	51
2	35
3	48
4	20
5	12
6	6

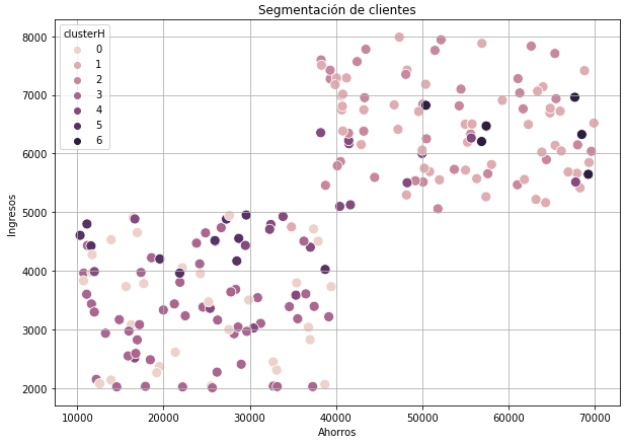
Podemos además contabilizar el numero de registros que pertenecen a cada clúster agrupándolos por su columna `clusterH` que acabamos de añadir y sumando el numero de registros en cada agrupación obteniendo los resultados que se muestran a la derecha.

El ultimo paso de la cauterización vendría a ser interpretar a cada una de las agrupaciones que se generaron y describirlas lo mejor que podamos enfocándonos en las partes que precisamente las diferencian unas de otras.

El primer paso para hacer esta interpretación seria obtener los centroides de cada clúster calculando los valores medios para cada una de sus variables tomando en cuenta todos los registros que pertenezcan a dicho clúster. Es decir, nuevamente se hará una agrupación `group by` por la columna `clusterH` solo que esta vez en lugar de sumar el numero de registros obtendremos la media de cada uno de los campos (ingresos, gastos, hijos, etc) con el método `mean` y obtendremos la siguiente matriz que nos describe de manera general los clústeres.

```
CentroidesH = Hipoteca.groupby('clusterH').mean()
CentroidesH
```

clusterH	ingresos	gastos_comunes	pago_coche	gastos_otros	ahorros	vivienda	estado_civil	hijos	trabajo
0	3421.133333	846.466667	309.933333	527.233333	24289.633333	295590.700000	0.233333	0.000000	2.000000
1	6394.019608	1021.627451	192.274510	533.039216	54382.529412	421178.764706	1.490196	2.254902	6.313725
2	6599.542857	1087.428571	204.771429	362.600000	51863.028571	515494.257143	0.685714	0.228571	2.885714
3	3189.687500	785.020833	243.208333	548.270833	23616.854167	277066.687500	1.645833	1.979167	6.208333
4	4843.750000	1009.200000	122.200000	572.850000	36340.650000	337164.850000	0.050000	0.100000	1.900000
5	4466.416667	1315.083333	114.416667	502.750000	23276.166667	269429.916667	1.666667	2.416667	6.750000
6	6404.500000	1176.166667	168.333333	769.333333	61715.500000	625138.833333	0.000000	0.000000	1.166667



Donde además con ayuda de `matplotlib` podemos obtener las mismas graficas de dispersión que calculamos al principio con `seaborn` pero ahora segmentadas, es decir, donde cada registro se identifique por un clúster la cual se muestra a la izquierda.

A continuación, se muestra la descripción de cada uno de los 7 clústeres para la jerarquía obtenida a partir de la matriz de datos estandarizada con la métrica de distancia euclidiana.

Clúster 0.

Conformado por 30 casos de una evaluación hipotecaría, con un ingreso promedio mensual de 3421 USD, con gastos comunes de 846 USD, otros gastos de 527 USD y un pago mensual de coche de 309 USD. Estos gastos en promedio representan casi la mitad del salario mensual (1682 USD). Por otro lado, este grupo de usuarios tienen un ahorro promedio de 24289 USD, y un valor promedio de vivienda (a comprar o hipotecar) de 295590 USD. Además, en su mayoría son solteros (0-soltero), sin hijos menores y tienen un tipo de trabajo asalariado (2-asalariado).

Se descartan como candidatos por su alto índice de gastos en comparación con sus ingresos.

Clúster 1.

Conformado por 51 usuarios candidatos a la evaluación hipotecaria, con un ingreso promedio mensual de 6394 USD, con gastos comunes de 1021 USD, otros gastos de 533 USD y un pago mensual en promedio a su coche de 192 USD. Estos gastos representan en promedio menos de un 30% de los ingresos mensuales (1746). Por otro lado, este grupo de usuarios tiene un ahorro promedio de 54382 USD, y un valor promedio de vivienda a comprar o hipotecar de 421178 USD. Además, en su mayoría casados (1-casado), con 2 hijos en promedio y con un trabajo autónomo y asalariado (6- autónomo y asalariado).

Se conservan como candidatos al crédito por su bajo índice de gastos y su alto índice de ahorros.

Clúster 2.

Conformado por 35 usuarios candidatos a la evaluación hipotecaria, con un ingreso promedio mensual de 6599 USD, con gastos comunes de 1087 USD, otros gastos de 362 USD y un pago mensual en promedio a su coche de 205 USD. Estos gastos representan en promedio aproximadamente un 25% de los ingresos mensuales (1654). Por otro lado, este grupo de usuarios tiene un ahorro promedio de 51863 USD, y un valor promedio de vivienda a comprar o hipotecar de 515494 USD. Además, en su mayoría casados (1-casado), sin hijos en su mayoría y con un trabajo como empresarios (3-empresario).

Se conservan como candidatos al crédito por su bajo índice de gastos y su alto índice de ahorros.

Clúster 3.

Conformado por 48 usuarios candidatos a la evaluación hipotecaria, con un ingreso promedio mensual de 3189 USD, con gastos comunes de 785 USD, otros gastos de 548 USD y un pago mensual en promedio a su coche de 243 USD. Estos gastos representan en promedio de más de un 45% de los ingresos mensuales (1576). Por otro lado, este grupo de usuarios tiene un ahorro promedio de 23616 USD, y un valor promedio de vivienda a comprar o hipotecar de 277066 USD. Además, en su mayoría divorciados (2-casado), con 2 hijos en su mayoría y con un trabajo autónomo y asalariado (6-autónomo y asalariado).

Se descartan como candidatos al crédito por su alto índice de gastos en comparación con sus ingresos.

Clúster 4.

Conformado por 20 usuarios candidatos a la evaluación hipotecaria, con un ingreso promedio mensual de 4844 USD, con gastos comunes de 1009 USD, otros gastos de 573 USD y un pago mensual en promedio a su coche de 122 USD. Estos gastos representan en promedio de más de un 35% de los ingresos mensuales (1704). Por otro lado, este grupo de usuarios tiene un ahorro promedio de 36341 USD, y un valor promedio de vivienda a comprar o hipotecar de 337165 USD. Además, prácticamente todos divorciados (0-casado), de igual manera sin hijos en su mayoría y con un trabajo asalariado (2-asalariado).

Se descartan como candidatos al crédito por su alto índice de gastos en comparación con sus ingresos.

Clúster 5.

Conformado por solo 12 usuarios candidatos a la evaluación hipotecaria, con un ingreso promedio mensual de 4466 USD, con gastos comunes de 1315 USD, otros gastos de 503 USD y un pago mensual en promedio a su coche de 114 USD. Estos gastos representan en promedio de más de un 30% de los ingresos mensuales (1932). Por otro lado, este grupo de usuarios tiene un ahorro promedio de 23276 USD, y un valor promedio de vivienda a comprar o hipotecar de 269429 USD. Además, en su mayoría divorciados (2-casado), con 2 hijos en su mayoría y un trabajo de empresario y autónomo (2-asalariado).

Se conservan como candidatos al crédito ya que, si bien su índice de gastos sobrepasa por poco el 30% de sus ingresos, tienen buenos niveles de ahorro y el costo de su vivienda no es tan elevado como los demás.

Clúster 6.

Es un segmento de clientes conformado por solo 6 usuarios, con un ingreso promedio mensual de 6404 USD, con gastos comunes de 1176 USD, otros gastos de 769 USD y un pago mensual de coche de 168 USD. Estos gastos en promedio representan casi una tercera parte del salario mensual (2113 USD). Por otro lado, este grupo de usuarios tienen un ahorro promedio de 61715 USD, y un valor promedio de vivienda (a comprar o hipotecar) de 625138 USD. Además, todos son solteros (0-soltero), sin hijos y tienen un tipo de trabajo en su mayoría autónomos (1-autónomo).

Se conservan como candidatos ya que su índice de gastos es menor al 30%, además de que cuentan con uno de los mejores índices de ahorros en la segmentación y al ser solteros sin hijos esto representa menos gatos que se puedan presentar a futuro.

Conclusiones.

En conclusión, la técnica de clustering jerárquico nos sirve para segmentar un determinado conjunto de datos en diversos grupos, cada cual con sus características y particularidades que los diferencian. Estas segmentaciones pueden ser muy útiles para obtener perfiles de usuario, patrones climáticos, etc. por ejemplo, en nuestro caso logramos etiquetar a un conjunto de 202 registros en 7 grupos para que en lugar de analizar caso por caso desde un inicio solamente se analicen estos grupos de usuarios reduciendo así el trabajo de la persona que se encarga de otorgar estos créditos.

Es importante considerar la dimensionalidad de nuestros datos ya que cada dimensión o variable en el modelo aumenta la complejidad y desempeño del modelo, es por eso que se buscan técnicas de reducción de variables como una matriz de correlaciones que nos indique cual es el nivel de dependencia entre las variables del modelo, donde para índices altos de una correlación positiva podamos considerar eliminar o juntar ciertas columnas para quedarnos con una sola, como desventaja tenemos que si hacemos una mala consideración podríamos estar eliminando información útil.

El numero de grupos en los cuales se van a dividir los grupos depende principalmente de la técnica que se utilizara para el escalado de los datos con el fin de optimizar los tiempos de calculo en el algoritmo, esto es, estandarizar o normalizar la matriz además de la métrica que se utilizara para obtener las distancias entre los registros; euclidiana, chebyshev o manhattan.