Basile Álvarez Andrés José

No. Cuenta: 316617187

Email: andresbasile123@gmail.com

Fecha: 19/10/21

Inteligencia Artificial

Grupo III

Semestre 2022-1

Reporte Práctica 4: Clustering Jerárquico (Segmentación de Clientes)

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 Generales:

En esta práctica, utilizaremos el clustering no jerárquico. Éste es un tipo de aprendizaje no supervisado en donde, a partir de una fuente de datos, se segmenta y delimitan grupos de objetos que son unidos por características que éstos tienen en común. El objetivo de este tipo de aprendizaje es dividir una población heterogénea de elementos en un número de grupos homogéneos, de acuerdo con sus similitudes.

Dentro de este tipo de aprendizaje, utilizaremos otros algoritmos de IA vistos previamente, como las métricas de distancia, las cuales nos serán útiles para conocer las similitudes entre objetos.

El clustering jerárquico organiza elementos de forma recursiva en una estructura de árbol y consiste en cuatro pasos fundamentales: medir la similitud de los elementos (utilizando métricas de distancia), agrupar los elementos similares, decidir la cantidad adecuada de grupos, interpretar los grupos obtenidos.

Se utilizarán datos que ya habíamos utilizado para otras prácticas. La fuente de datos es el archivo *hipoteca.csv*, el cual cuenta con 202 registros con los siguientes datos:

- 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.

Desarrollo

Primeramente, tenemos que definir aquellas bibliotecas de Python que nos serán útiles para importar, limpiar y analizar los datos contenidos en el archivo separado por comas *Hipoteca.csv*. Estas serán: *pandas* (manipulación y análisis de datos), *matplotlib* (para la creación de gráficas y visualización de los datos), *numpy* para utilizar vectores y matrices de *n* dimensiones y *seaborn*, para la visualización de datos basado en *matplotlib.*

Más tarde, importamos el archivo *Hipoteca.csv* y lo primero que hacemos es guardarlo en un DataFrame. La importación del archivo se realizó a partir del explorador de archivos que abrimos utilizando el comando *files.upload()*. Una vez importados los datos, mostramos el DataFrame que los contiene:

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

202 rows × 10 columns

Figura 1: DataFrame *Hipoteca*.

Como los datos ya se encuentran preparados, comenzaremos mostrando información sobre ellos, para después poder hacer una selección de variables para evitar la alta dimensionalidad de los datos. Observamos que al imprimir información sobre la variable *comprar*, esta ya se encontraba previamente agrupada (entre los que sí tienen posibilidad de compra y los que no) en un análisis previo.

Antes de eliminarla, utilizamos la biblioteca *seaborn* para mostrar un *pairplot* (el cual nos permitirá visualizar rápidamente gráficas de dispersión que muestran la relación entre las variables de nuestro *dataframe*). Dentro de este *pairplot*, utilizamos la variable *compra* como *hue*, destacando su valor con respecto de las demás variables que se están graficando. En la siguiente

imagen, de color azul se muestra a aquellas personas que no tienen la posibilidad de comprar (y por lo tanto rentarán) y en naranja a aquellas que sí tienen dicha posibilidad:

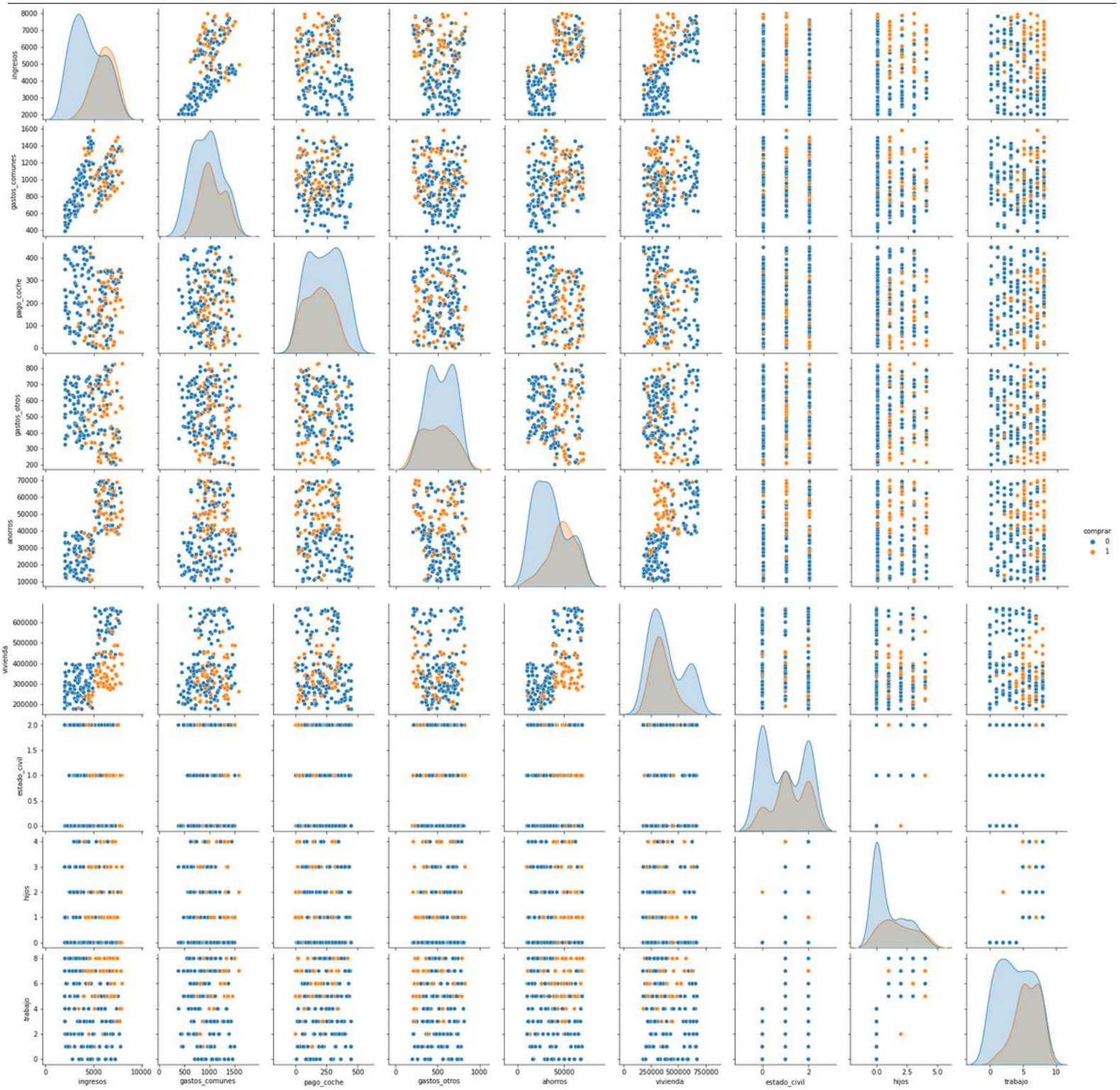

Figura 2: Resultado de utilizar la función *pairplot* sobre el *dataframe*.

Las gráficas de tipo *pairplot* son una gran manera de identificar tendencias para un análisis posterior de los datos.

Más tarde, a manera de pormenorizar el análisis, obtenemos nada más algunas gráficas (para un par de variables) utilizando la función *scatterplot* de *seaborn*. A continuación, se muestra un ejemplo con las variables *ahorros* e *ingresos*, utilizando como *hue* la variable *comprar*:

```
sns.scatterplot(x='ahorros', y ='ingresos', data=Hipoteca, hue='comprar')
plt.title('Gráfico de dispersión')   # dispersión moderada
plt.xlabel('Ahorros')
plt.ylabel('Ingresos')
plt.show()
```

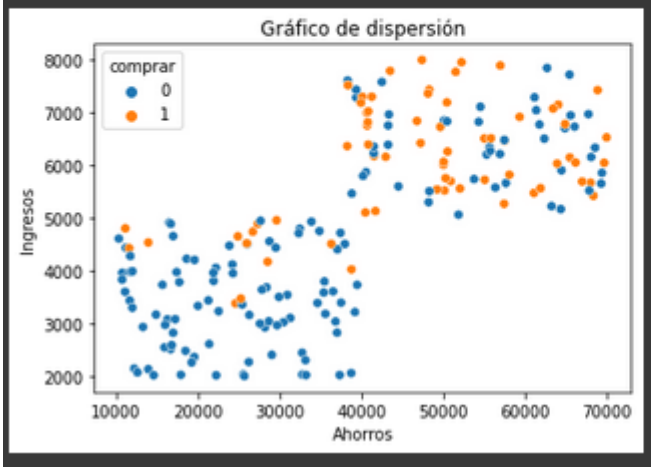

Figura 3: Resultado de utilizar la función *scatterplot* sobre el *dataframe.*

En la gráfica anterior, notamos una dispersión moderada de los datos, sin tendencias demasiado claras para definir la relación entre *ahorros* e *ingresos.*

Más tarde, utilizamos un método más preciso para definir la correlación entre las variables, la *matriz de correlaciones*, con el método *corr* de Python, pasando como parámetro que queremos utilizar (en este caso utilizaremos el método Pearson para calcular la correlación). Este método asigna un valor entre -1 y 1, donde 0 implica que no hay correlación, 1 es una correlación total positiva y -1 es una correlación total negativa. Una correlación con valor de 0.7, por ejemplo, implicaría que dos variables tienen una relación significativa y positiva, donde si la variable A sube, la variable B también subirá. Si la correlación fuera negativa y la variable A sube, la variable B bajaría.

Para nuestros datos, obtuvimos lo siguiente de la matriz de transición:

	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

Figura 4: Matriz de correlación entre los datos. Vemos que, por ejemplo, existe una fuerte relación entre los ingresos y los ahorros, con un valor de 0.712 positivo.

Al imprimir los 10 mayores valores, obtenemos lo siguiente:

```
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
Name: ingresos, dtype: float64
```

Figura 5: 10 mayores valores de correlación.

Otra forma de obtener una representación gráfica de la correlación entre las variables es utilizando un *heatmap*, como se muestra a continuación:

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

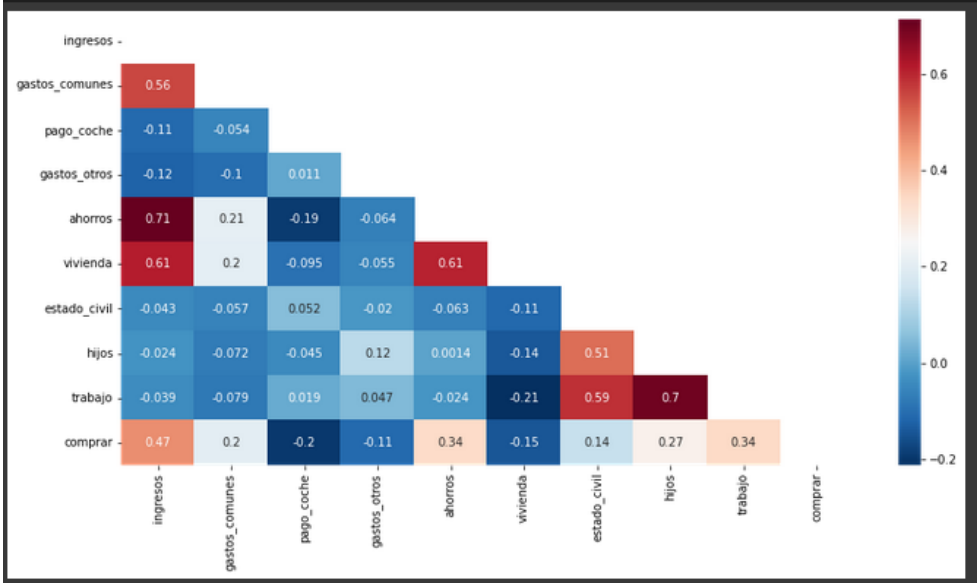

Figura 6: *Heatmap* de los valores de correlación.

Vemos aquí que existe una correlación alta entre *trabajo* e *hijos*, también. Hecho el análisis anterior, decidimos eliminar la variable *comprar* ya que este fue un campo calculado con base en un análisis hipotecario anterior, ya representaba una condición.

Posteriormente, se hace la aplicación del algoritmo jerárquico y, ya que son algoritmos basados en distancias, es fundamental hacer un escalamiento de los datos para que cada una de las variables que utilizamos contribuya de manera igual en el análisis.

Importamos *StandardScaler* y *MinMaxScaler* de *sklearn.preprocessing* para poder hacer el escalamiento de la matriz de datos con el método *StandardScaler*. El método *MinMax* podrá ser útil cuando se identifica que ninguna variable tenga un dato fuera de rango (por ejemplo, una edad de 150 años). Por defecto, utilizaremos la matriz euclidiana para la matriz de distancias.

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler  #min max scaler tien
estandarizar = StandardScaler()                                 # Se instancia el obje
MEstandarizada = estandarizar.fit_transform(MatrizHipoteca)   # Se calculan la media

#MINMAXSCALER puede ser útil cuando se identifica que ninguna variable tenga un dato
# por defecto se utiliza distancia euclideana para matriz de distancias.
```

```
pd.DataFrame(MEstandarizada) #
```

	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

202 rows × 9 columns

Figura 7: Matriz estandarizada utilizando el método *StandardScaler*.

Luego, importamos la biblioteca *cluster.hierarchy* de *scipy* y el método *AgglomerativeClustering* de *sklearn.cluster* para realizar el árbol de clustering jerárquico. Creamos un *dendogram* con la matriz estandarizada previamente y la métrica de distancia euclidiana. El *dendogram* permitirá visualizar los grupos que se generan en el clúster jerárquico. Al utilizar la métrica de distancia euclidiana, obtenemos 7 clústeres diferentes, como se muestra a continuación:

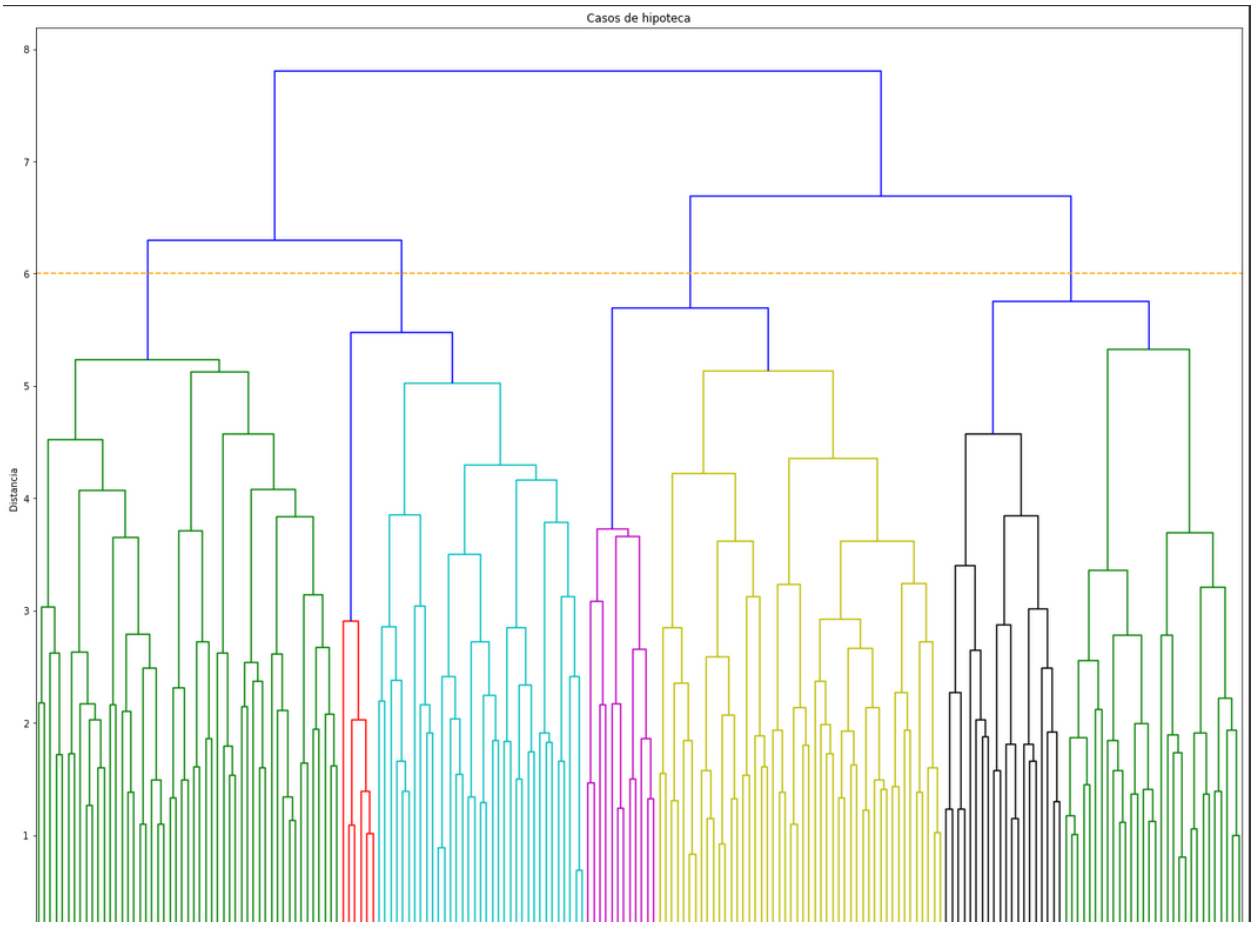

Figura 8: *Dendogram utilizando la métrica euclidiana de distancia.*

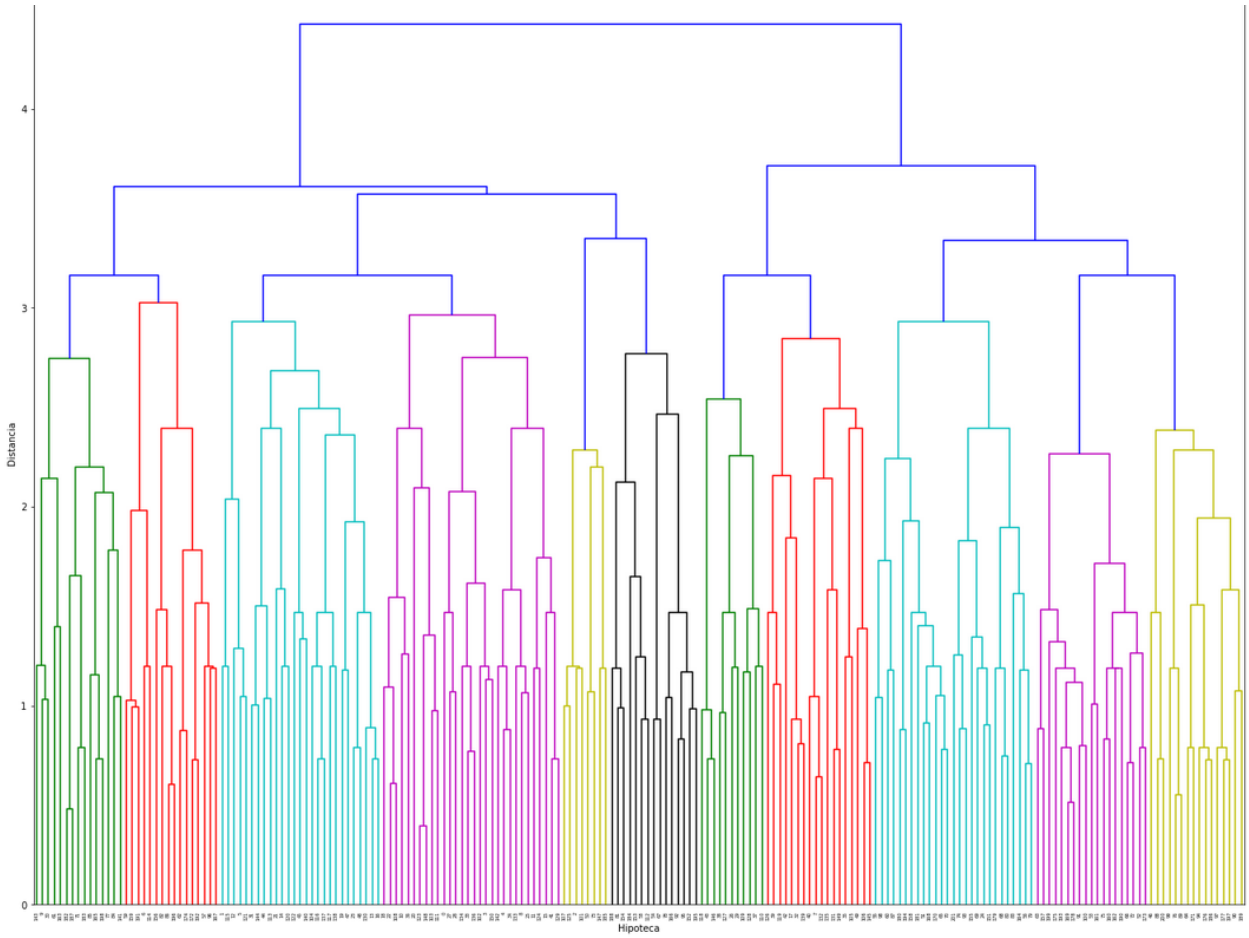

Figura 9: *Dendogram utilizando la métrica chebyshev de distancia*. Vemos que para esta métrica de distancia, aparecen 11 clústers diferentes.

Para cada uno de los registros, le asignamos una etiqueta que indique a qué clúster pertenece, como podemos ver a continuación:

```
#Se crean las etiquetas de los elementos en los clústeres
MJerarquico = AgglomerativeClustering(n_clusters=7, linkage='complete', affinity='euclidean')
MJerarquico.fit_predict(MEstandarizada)
MJerarquico.labels_   #poner etiquetas a cada cluster. #el array indica qué cluster va a tomar
```

```
array([4, 1, 1, 2, 4, 1, 1, 6, 2, 1, 2, 2, 1, 1, 2, 1, 1, 1, 2, 2, 2, 1,
       2, 1, 4, 2, 1, 2, 2, 1, 1, 2, 1, 2, 2, 1, 2, 1, 1, 1, 6, 1, 1, 1,
       2, 2, 4, 2, 1, 6, 5, 3, 3, 3, 4, 3, 3, 0, 4, 0, 3, 3, 0, 3, 0, 3,
       3, 4, 3, 0, 3, 3, 3, 5, 0, 3, 0, 5, 5, 3, 3, 4, 0, 3, 3, 5, 0, 3,
       3, 0, 0, 3, 5, 0, 0, 5, 0, 0, 3, 0, 3, 1, 2, 1, 1, 2, 6, 1, 2, 1,
       1, 2, 4, 2, 2, 1, 1, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 4,
       6, 4, 2, 4, 2, 1, 1, 1, 2, 1, 2, 1, 2, 6, 1, 1, 2, 4, 2, 4, 5, 4,
       4, 4, 0, 3, 3, 0, 3, 3, 3, 1, 3, 5, 3, 0, 3, 3, 3, 0, 0, 3, 0, 3,
       0, 0, 3, 3, 3, 3, 3, 3, 4, 5, 0, 3, 4, 0, 3, 0, 0, 3, 3, 5, 0, 0,
       5, 3, 3, 4])
```

Figura 10: *Labels* para los registros, indicando a qué clúster pertenece cada uno.

	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

202 rows × 10 columns

Figura 11: Los datos y el *label* al que pertenece cada uno, indicado por la columna *clusterH*.

Contamos cuántos elementos hay en cada clúster y mostramos los resultados:

```
clusterH
0    30
1    51
2    35
3    48
4    20
5    12
6     6
Name: clusterH, dtype: int64
```

Figura 12: Cantidad de registros en cada clúster

Mostramos los promedios o *centroides* para cada uno de los clústeres que obtuvimos anteriormente:

```
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

Figura 13: Centroide de cada una de las columnas para los clústers.

Conclusiones

A lo largo de esta práctica, tuvimos como principal objetivo obtener clústeres de usuarios con características similares, evaluados para la adquisición de una casa a través de un crédito hipotecario con tasa fija a 30 años. Para lograr tal objetivo, se utilizó el algoritmo de clústering jerárquico.

Haciendo una descripción de los clústeres obtenidos, tenemos que:

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).

Clúster 1: El clúster más grande formado, con 51 casos de evaluación hipotecaria. En este clúster se tiene un ingreso promedio de 6394 USD, con gastos comunes de 1021 USD, otros gastos de 533 USD y un pago mensual de coche de 192 USD. Estos gastos representan una cuarta parte de los ingresos promedio, lo cual no es un gasto demasiado alto comparado con los ingresos. Este grupo de usuarios tiene un ahorro promedio de 54382 USD, y un valor promedio de vivienda de 421178 USD. En su mayoría, se encuentran casados o divorciados, con 2 hijos en promedio y tienen un tipo de trabajo autónomo y asalariado.

Clúster 2: En este clúster, se encuentran 35 casos de evaluación hipotecaria, con un ingreso promedio de 6599 USD, gastos comunes de 1087 USD, otros gastos por 362 dólares y un pago de coche de 204 USD. Los ingresos en este grupo de personas representan un número 4 veces mayor que sus gastos. Este grupo tiene un ahorro promedio de 51863 USD y un costo de vivienda de 515494 USD. En este grupo, el promedio se encuentra soltero o casado, sin hijos y con trabajo de tipo asalariado o empresarial.

Clúster 3: El segundo clúster más grande formado, con 48 casos de evaluación hipotecaria. En este clúster se tiene un ingreso promedio de 3189 USD, con gastos comunes de 785 USD, otros gastos de 548 USD y un pago mensual de coche de 243 USD. Estos gastos representan la mitad de los ingresos promedio, lo es un gasto demasiado alto comparado con los ingresos. Este grupo de usuarios tiene un ahorro promedio de 23616 USD, y un valor promedio de vivienda de 277066 USD. En su mayoría, se encuentran casados o divorciados, con 2 hijos en promedio y tienen un tipo de trabajo autónomo y asalariado.

Clúster 4: En este clúster, se encuentran 20 casos de evaluación hipotecaria, con un ingreso promedio de 4843 USD, gastos comunes de 1009 USD, otros gastos por 572 dólares y un pago de coche de 122 USD. Los ingresos en este grupo de personas representan un número casi tres veces mayor que sus gastos. Este grupo tiene un ahorro promedio de 36340 USD y un costo de vivienda de 337164 USD. En este grupo, el promedio se encuentra soltero, sin hijos y con trabajo de tipo asalariado.

Clúster 5: Es un segmento de clientes conformado por 12 usuarios, con un ingreso promedio mensual de 4466 USD, con gastos comunes de 1315 USD, otros gastos de 502 USD y un pago mensual de coche de 114 USD. Los ingresos representan 2.3 veces los gastos de estos usuarios, en promedio. Por otro lado, este grupo de usuarios tienen un ahorro promedio de 23276 USD, y un valor promedio de vivienda de 269429 USD. La mayoría se encuentra casado o divorciado, con 2 hijos en promedio y con trabajo autónomo y asalariado o autónomo y empresario.

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).

A través de la creación de clústeres, logramos definir propiedades o características comunes a grupos de usuarios dentro del conjunto de datos para créditos hipotecarios con tasa fija a 30 años.