Universidad Nacional Autónoma de México

Facultad de Ingeniería

PRÁCTICA 4

SELECCIÓN DE CARACTERÍSTICAS

ANÁLISIS DE COMPONENTES PRINCIPALES (PCA)

Minería de Datos

Profesor:

Dr. Molero Castillo Guillermo Gilberto

Grupo 1

Alumna:

Monroy Velázquez Alejandra Sarahí

No. Cuenta: 314000417

OBJETIVO

Realizar un análisis de componentes principales (ACP o PCA, Principal Component Analysis) para reducir la cantidad de variables del conjuntos de datos.

DESARROLLO

El conjunto de datos corresponde a diversas variables relacionada a las hipotecas. Este conjunto de datos incluye:

- 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.

Primero comenzamos la importación de bibliotecas correspondientes que nos ayudarán para la realización del código, las cuales son *pandas* para la manipulación y análisis de datos, *numpy* para crear vectores y matrices, *matplotlib* para la generación de gráficas, así como *seaborn* para la visualización de datos. En esta práctica agregamos una biblioteca más, la cual es *sklearn* que nos ayudara para la estandarización de los datos y para el análisis de componentes. Por último, la biblioteca files para subir el archivo *csv*.

Una vez importadas, el *dataframe* se lee y se despliega en pantalla:

```
[1]  import pandas as pd                          # Para la manipulación y análisis de datos
     import numpy as np                           # Para crear vectores y matrices n dimensionales
     import matplotlib.pyplot as plt              # Para la generación de gráficas a partir de los datos
     import seaborn as sns                        # Para la visualización de datos basado en matplotlib
     %matplotlib inline
     from sklearn.preprocessing import StandardScaler
     from sklearn.decomposition import PCA
```

```
from google.colab import files
files.upload()
```

```
Elegir archivos  Hipoteca.csv
• Hipoteca.csv(application/vnd.ms-excel) - 8014 bytes, last modified: 30/9/2021 - 100% done
Saving Hipoteca.csv to Hipoteca.csv
{'Hipoteca.csv': b'ingresos,gastos_comunes,pago_coche,gastos_otros,ahorros,vivienda,estado_civil,hijos,trabajo,compra
```

```
[3] Hipoteca = pd.read_csv("Hipoteca.csv")
    Hipoteca
```

	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

Ahora que el *dataframe* está cargado, comenzamos con los pasos vistos en clase:

Paso 1)

En este paso estandarizamos o normalizamos el rango de las variables iniciales, para que cada una de ellas contribuya por igual al análisis. Para ello instanciamos el objeto StandardScaker, y con el método *fit()* calculamos media y desviación estándar de cada variable. Por último, normalizamos los datos mediante el método *trasnsform()* y guardamos en una nueva variable llamada *MNormalizada*.

```
[4] normalizar = StandardScaler()                    # Se instancia el objeto StandardScaler
    normalizar.fit(Hipoteca)                         # Se calcula la media y desviación para cada variable
    MNormalizada = normalizar.transform(Hipoteca)    # Se normalizan los datos
```

```
[5] MNormalizada.shape
```

```
(202, 10)
```

```
[6] pd.DataFrame(MNormalizada, columns=Hipoteca.columns)
    #MNormalizada
```

	ingresos	gastos_comunes	pago_coche	gastos_otros	ahorros	vivienda	estado_civil	hijos	trabajo	comprar
0	0.620129	0.104689	-1.698954	0.504359	0.649475	0.195910	-1.227088	0.562374	-0.984420	1.419481
1	1.063927	-0.101625	-0.712042	-0.515401	0.259224	1.937370	-0.029640	1.295273	0.596915	-0.704483
2	0.891173	0.226266	-0.912634	1.667244	1.080309	-0.379102	1.167809	-0.170526	1.387582	1.419481
3	1.274209	1.128886	-1.578599	-1.559015	0.909604	2.114062	-1.227088	-0.903426	-0.589086	-0.704483
4	0.719611	-0.400042	0.090326	0.027279	0.159468	-0.179497	-1.227088	-0.903426	-0.589086	1.419481
...	...	...	...	...	...	...	...	...	...	...
197	-0.671949	-1.037402	1.125381	-0.163554	-1.617963	-0.075199	-1.227088	-0.903426	-0.984420	-0.704483
198	-0.594508	0.215214	0.467439	-0.241079	-0.973876	-0.683130	1.167809	1.295273	1.387582	-0.704483
199	-1.057368	-0.061099	0.515581	1.005294	-0.183849	0.107880	-0.029640	1.295273	1.387582	-0.704483
200	-0.968013	-0.385305	1.261783	0.814462	-1.083273	0.026040	-0.029640	0.562374	0.201581	-0.704483
201	-0.578424	0.683102	-0.856468	-0.795686	-1.545397	-0.851037	-1.227088	-0.903426	-0.193753	-0.704483

202 rows x 10 columns

Paso 2) y 3)

Una vez que los datos han sido normalizados, calculamos la matriz de covarianzas o correlaciones y calculamos los componentes (eigen-vectores) y la varianza (eigen-valores), y los imprimimos.

```
pca = PCA(n_components=10)            # Se instancia el objeto PCA    #pca=PCA(n_components=None), pca=PCA(.85)
pca.fit(MNormalizada)                 # Se obtiene los componentes
print(pca.components_)
```

```
[[-0.54934302 -0.34146007  0.15049138  0.11746503 -0.48939638 -0.4412933
   0.15344247  0.14007281  0.16160045 -0.20409138]
 [ 0.15909755  0.05584992 -0.07658077  0.00556506  0.13553262 -0.07327675
   0.45384378  0.52400611  0.55172097  0.39620952]
 [ 0.01547889 -0.27036268  0.1692496   0.58506991  0.2251488   0.49591877
   0.1478699   0.18628931  0.07154605 -0.44429353]
 [ 0.11615868  0.2491118   0.78967879 -0.40154183 -0.09350787  0.12457483
   0.2603972  -0.04112256  0.0656939  -0.20382827]
 [-0.13379015 -0.52550381 -0.2921437  -0.66008455  0.16757599  0.28199311
   0.19632196  0.00247784  0.03001071 -0.19451342]
 [-0.05732063  0.58507176 -0.47072239 -0.03977609 -0.31551545  0.13904418
   0.30650475  0.07948763 -0.00861148 -0.46045932]
 [ 0.06075777 -0.10974406 -0.04461737  0.20628126  0.05710368 -0.11375472
   0.7121099  -0.58145566 -0.21011973  0.18487727]
 [-0.06201356  0.06098212 -0.05727528  0.03228667  0.03174719  0.0689216
  -0.19665212 -0.56847793  0.78088384 -0.10361354]
 [ 0.30317093 -0.22727848 -0.02493028  0.03860608 -0.73815788  0.46100264
  -0.04341353 -0.03921835  0.02863228  0.30442869]
 [ 0.73385456 -0.243417   -0.08018422 -0.01580487 -0.07686939 -0.45756887
  -0.02577314  0.01679541  0.06580472 -0.41829603]]
```

Paso 4)

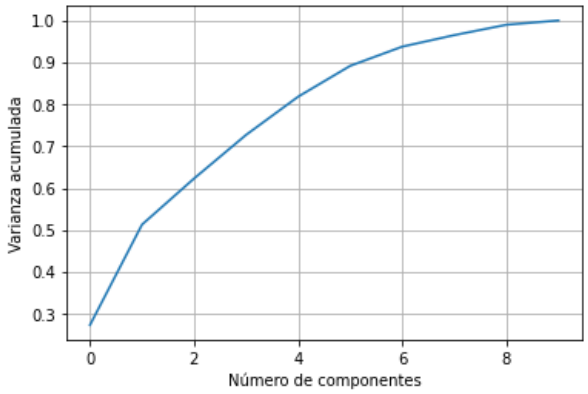
En este paso decidimos el número de componentes principales, primeramente, calculamos el porcentaje de relevancia, es decir, entre el 75% y 90% de varianza total; si tomamos 7 componentes obtenemos el 93%, por lo que lo correcto será tomar 6 componentes ya que nos da un 89% y este porcentaje si entra dentro del rango requerido.

```
[11] Varianza = pca.explained_variance_ratio_
     print('Porporción de varianza:', Varianza)
     print('Varianza acumulada:', sum(Varianza[0:6]))
     #Con 6 componentes se tiene el 89% de varianza acumulada y con 7 el 93%
```

```
Porporción de varianza: [0.27368381 0.23958688 0.10991099 0.10411098 0.09105662 0.07352523
 0.0457761  0.02745036 0.02469122 0.01020781]
Varianza acumulada: 0.8918745043774533
```

Luego graficamos la varianza acumulada de los componentes, como observamos en la gráfica, podemos identificar con mayor facilidad el grupo de componentes con mayor varianza.

```
[9]  # Se grafica la varianza acumulada en los componentes
     plt.plot(np.cumsum(pca.explained_variance_ratio_))
     plt.xlabel('Número de componentes')
     plt.ylabel('Varianza acumulada')
     plt.grid()
     plt.show()
```



Paso 5)

Como último paso examinamos la proporción de relevancia o las cargas. Se revisan los valores absolutos de los componentes principales seleccionados. Como cuanto mayor sea el valor absoluto, más importante es esa variable en el componente principal, en este caso, identificamos las cargas mayores al 50%:

```
CargasComponentes = pd.DataFrame(abs(pca.components_), columns=Hipoteca.columns)
CargasComponentes
```

	ingresos	gastos_comunes	pago_coche	gastos_otros	ahorros	vivienda	estado_civil	hijos	trabajo	comprar
0	0.549343	0.341460	0.150491	0.117465	0.489396	0.441293	0.153442	0.140073	0.161600	0.204091
1	0.159098	0.055850	0.076581	0.005565	0.135533	0.073277	0.453844	0.524006	0.551721	0.396210
2	0.015479	0.270363	0.169250	0.585070	0.225149	0.495919	0.147870	0.186289	0.071546	0.444294
3	0.116159	0.249112	0.789679	0.401542	0.093508	0.124575	0.260397	0.041123	0.065694	0.203828
4	0.133790	0.525504	0.292144	0.660085	0.167576	0.281993	0.196322	0.002478	0.030011	0.194513
5	0.057321	0.585072	0.470722	0.039776	0.315515	0.139044	0.306505	0.079488	0.008611	0.460459
6	0.060758	0.109744	0.044617	0.206281	0.057104	0.113755	0.712110	0.581456	0.210120	0.184877
7	0.062014	0.060982	0.057275	0.032287	0.031747	0.068922	0.196652	0.568478	0.780884	0.103614
8	0.303171	0.227278	0.024930	0.038606	0.738158	0.461003	0.043414	0.039218	0.028632	0.304429
9	0.733855	0.243417	0.080184	0.015805	0.076869	0.457569	0.025773	0.016795	0.065805	0.418296

Las variables restantes que no tuvieron cargas mayores al porcentaje esperado se eliminan del *dataframe*, en este caso las variables son *ahorros, vivienda, estado_civil y comprar*.

```
[13] DatosHipoteca = Hipoteca.drop(columns=['ahorros', 'vivienda', 'estado_civil', 'comprar'])
     DatosHipoteca
```

	ingresos	gastos_comunes	pago_coche	gastos_otros	hijos	trabajo
0	6000	1000	0	600	2	2
1	6745	944	123	429	3	6
2	6455	1033	98	795	1	8
3	7098	1278	15	254	0	3
4	6167	863	223	520	0	3
...	...	...	...	...	...	...
197	3831	690	352	488	0	2
198	3961	1030	270	475	3	8
199	3184	955	276	684	3	8
200	3334	867	369	652	2	5
201	3988	1157	105	382	0	4

202 rows × 6 columns

CONCLUSIÓN

En esta práctica aprendimos un método más para la selección de características, el cual fue el análisis de componentes principales, a diferencia del análisis correlacional de datos, este método fue mucho más rápido de realizar, y personalmente considero que es mucho mejor, sin embargo, hay que considerar con qué conjunto de datos se trabajan y que algoritmos se implementaran sobre el mismo para poder seleccionar el mejor método para la selección de características.