Minería de Datos **Ibañez Guzman Osvaldo**
Grupo: 1 **Semestre 2023-1**
Dr. Guillermo Gilberto Molero Castillo **Práctica #6: Análisis de componentes principales (PCA)**

Objetivo

Identificar una fuente de datos para realizar una selección de características a través de Análisis de Componentes Principales (ACP) y Análisis Correlacional de Datos (ACD).

Fuente de Información

https://www.kaggle.com/datasets/kashishnaqvi/suicidal-behaviours-among-adolescents

Fuente de Datos

- Country – País de procedencia
- Year – Año que se realizado la encuesta
- Age Group – Grupo de edades
- Sex – Genero de personas que realizaron la encuesta
- Currently_Drink_Alcohol – Bebia Alcohol regularmente
- Really_Get_Drunk – Realmente se emborrachaba
- Overwieght - Sobrepeso
- Use_Marijuana – Consumía Mariguana
- Have_Understanding_Parents – Tenia padres comprensivos
- Missed_classes_without_permssion – Faltaba a clases sin permiso
- Had_sexual_relation – Tenía relaciones sexuales
- Smoke_cig_currently – Fumaba regularmente
- Had_fights – Tenia Peleas
- Bullied – Sufría acoso
- Got_Seriously_injured – Tenia heridas serias
- No_close_friends – No tenía amigos cercanos
- Attempted_suicide – Intentos previos de suicidios

Desarrollo

Para iniciar la practica lo primero es la importación de las bibliotecas a utilizar, las cuales nos ayudaran a procesas y manejar de diferentes formas los datos obtenidos, así como en la visualización de diferentes tablas o gráficas, las bibliotecas que usaremos son *pandas, numpy, matplotlib.pyplot* y *seaborn.*

Los datos obtenidos se encuentran en una carpeta en el repositorio de Github por lo que obtendremos la liga de estos y la colocaremos como parámetro de la función *read_csv,* la cual nos ayudara a obtener los datos y colocarlos en una tabla, donde podemos observar que contamos con una tabla de 106 registros y 17 columnas.

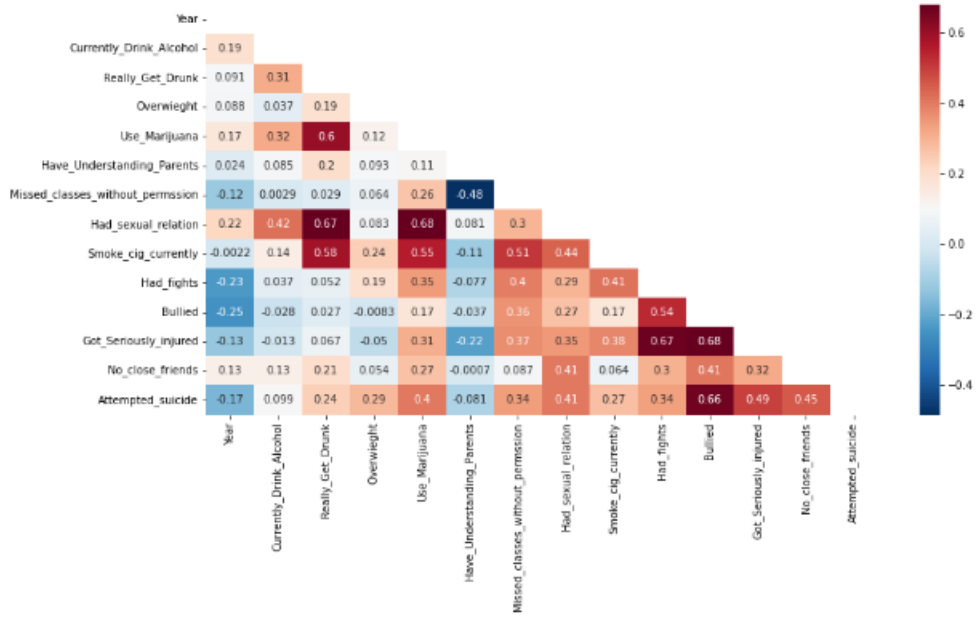
```
url = "https://raw.githubusercontent.com/OsvaldoIG/MineriaDatos/main/Data/GHSH_Pooled_Data1.csv"
Suicidio = pd.read_csv(url)
Suicidio
```

	Country	Year	Age Group	Sex	Currently_Drink_Alcohol	Really_Get_Drunk	Overwieght	Use_Marijuana	Have_Understanding_Parents	Missed_classes_with
0	Argentina	2018	13-15	Female	50.3	30.7	27.8	7.9	41.5	
1	Argentina	2018	13-15	Male	44.9	26.1	39.1	8.4	44.5	
2	Argentina	2018	16-17	Female	67.2	56.3	22.5	21.9	37.1	
3	Argentina	2018	16-17	Male	68.1	55.8	27.9	27.0	39.8	
4	Argentina	2012	13-15	Male	49.3	28.9	35.9	10.6	46.2	
...	...	...	...	...	...	...	...	...	...	
101	Vanuatu	2011	13-15	Female	5.8	4.7	13.6	1.9	20.2	
102	Wallis and Futuna	2015	13-15	Male	32.2	35.5	60.5	4.0	36.3	
103	Wallis and Futuna	2015	13-15	Female	24.4	27.1	63.0	2.0	36.3	
104	Wallis and Futuna	2015	16-17	Male	48.3	53.7	57.8	10.1	36.5	
105	Wallis and Futuna	2015	16-17	Female	42.9	51.7	70.6	3.9	37.8	

106 rows × 17 columns

Como sabemos el análisis de componentes principales cuenta con un total de seis pasos, el primero de estos es **Buscar evidencia de variables correlacionadas**. Esto lo haremos mostrando un mapa de calor donde podremos ver aquellas correlaciones fuertes, para poder identificar que variables tienen dicha correlación mayor a 0.66 o menor a -0.66

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



De este mapa de calor podemos observar una serie de correlaciones fuertes, entre las que se encuentran.

- Tener relaciones sexuales y Emborracharse
- Tener relaciones sexuales y Usar Mariguana
- Tener heridas serias y Tener Peleas
- Tener heridas serias y sufrir acoso
- Intentos de suicidios previos y sufrir acoso

Una vez que identificamos dichas correlaciones debemos proceder a la **estandarización de los datos**, en este proceso usaremos dos objetos el primero es *PCA* de *sklearn.decomposition* y usaremos *StandardScaler* de *sklearn.preprocessing*, este último nos servirá para la estandarización de los datos pero si lo que buscamos es normalizarlos ponemos usar *MinMaxScaler* de la misma biblioteca, para poder usar la función de estandarización es necesario eliminar aquellas columnas con valores categóricos es de ir, aquellas variables que no son numéricas, en este caso son **Country**, **Age Group**, **Sex** y para nuestro análisis eliminaremos **Year**, además de eliminar aquellos registros que contengan algún valor nulo, finalmente calcularemos la Vedia y al desviación estándar para cada una de nuestras variables, además de escalarlos. Obteniendo la siguiente matriz.

```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, MinMaxScaler
Estandarizar = StandardScaler()
NuevaMatriz = Suicidio.drop(columns=['Country','Age Group','Sex','Year']).dropna()
MEstandarizada = Estandarizar.fit_transform(NuevaMatriz)
```

```
pd.DataFrame(MEstandarizada, columns=NuevaMatriz.columns)
```

	Currently_Drink_Alcohol	Really_Get_Drunk	Overwieght	Use_Marijuana	Have_Understanding_Parents	Missed_classes_without_permssion	Had_sexual_relatior
0	0.338487	0.593750	0.793694	0.395384	1.173445	0.161038	1.04129
1	0.364184	0.443975	-0.089511	-0.084082	1.493255	-0.081405	0.28407
2	0.131074	-0.090935	-0.659522	-0.750657	0.265875	-1.069828	0.34914
3	0.211837	-0.269238	-0.308746	-0.820823	0.611616	-2.216771	-0.69202
4	0.287093	0.643675	-1.248326	-0.598631	0.058430	-1.209699	0.57986
...	...	...	...	...	...	...	..
95	-0.459961	-1.132226	-0.603148	-0.622020	-1.073871	1.335956	-1.03514
96	0.024614	1.064470	2.334606	-0.376440	0.317736	0.226311	0.36098
97	-0.118556	0.465371	2.491202	-0.610326	0.317736	-1.312271	-0.72752
98	0.320132	2.362518	2.165481	0.336913	0.335023	0.804445	1.76301
99	0.221014	2.219875	2.967256	-0.388134	0.447389	0.645925	-0.20693

100 rows × 13 columns

Ahora debemos **obtener la matriz de covarianza y calcular los componentes y la varianza** existente. Esto lo haremos con el objeto *PCA* que usaremos todos los componentes de nuestra matriz estandarizada y así obtendremos nuestros componentes.

```
pca = PCA(n_components=None)
pca.fit(MEstandarizada)
print(pca.components_)
```

```
[[ 1.30994001e-01  2.82153756e-01  1.06450762e-01  3.39410728e-01
  -5.51073974e-02  2.64696585e-01  3.60731276e-01  3.13766754e-01
   3.16355977e-01  2.92231718e-01  3.34359781e-01  2.48998103e-01
   3.40211237e-01]
 [ 3.67307611e-01  4.58836720e-01  8.89788445e-02  2.71897388e-01
   3.25082059e-01 -2.58207603e-01  2.94242511e-01  9.24626710e-02
  -2.47007512e-01 -3.59352341e-01 -3.16760542e-01  8.27059262e-03
  -1.21155220e-01]
 [ 4.00516568e-02 -1.10843637e-01 -1.07717497e-01 -6.67843506e-02
   4.67231475e-01 -4.39646984e-01  5.65270694e-02 -4.37389814e-01
   1.55174018e-02  3.40076764e-01  8.72129194e-02  4.34084167e-01
   2.28075230e-01]
 [-2.61817622e-01 -4.96528731e-02  8.35182169e-01 -4.76622882e-02
   2.91240024e-01 -6.83416722e-02 -1.94134393e-01  1.30310628e-01
   1.38346444e-01 -1.25424657e-02 -1.40660079e-01 -8.49696523e-02
   1.96780758e-01]
 [-2.31178440e-01 -2.27868722e-02 -2.89007754e-01  1.78099200e-01
   5.21135259e-01 -1.55132833e-01 -5.03057627e-02  2.81178969e-01
   3.64736907e-01  3.27486957e-02  2.63030863e-01 -3.73900578e-01
  -3.32354748e-01]
 [ 8.08721411e-01 -3.16474874e-01  1.83738229e-01 -9.57244176e-02
   3.68961923e-03 -7.02728279e-03 -9.13977916e-02 -8.45656834e-02
   3.56123438e-01  2.66725961e-02  8.25914915e-02 -1.94736897e-01
  -1.22651463e-01]
 [-1.39191115e-01 -6.51990657e-02  1.07237166e-01  7.13696601e-03
  -1.89061928e-01 -1.37137109e-01  3.18917062e-02  3.17884788e-02
   4.25441939e-01 -3.50096385e-01 -2.54660194e-03  6.21509055e-01
  -4.67329936e-01]
 [-4.63443242e-02 -4.42887561e-03  1.06135031e-01  3.06040242e-01
```

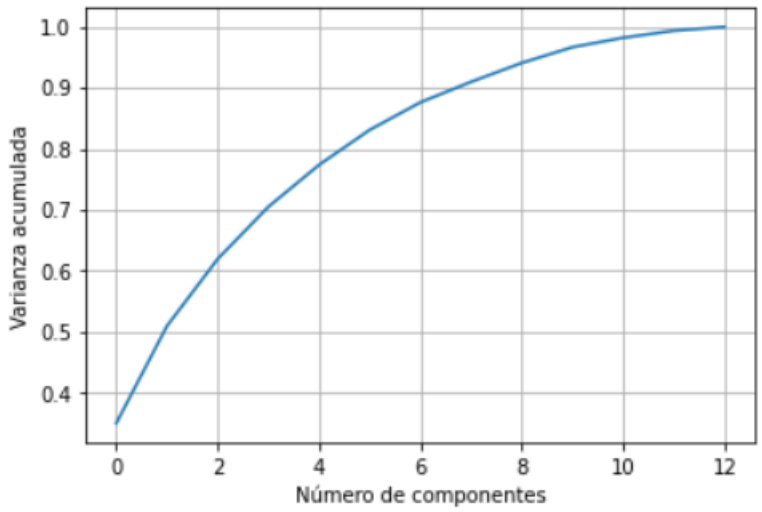
Ya con las componentes, **decidiremos el número de componentes principales** a través de las varianzas, buscamos que la varianza total se encuentre entre un 75% y 90%, al mostrar las varianzas debemos obtener la varianza acumulada, la cual es tomando la porción y sumando todos sus valores anteriores. En este caso usaremos 7 componentes ya que nos da una acumulada del 87%.

```
Varianza = pca.explained_variance_ratio_
print('Proporción de varianza:', Varianza)
print('Varianza acumulada:', sum(Varianza[0:7]))
#4 - 70
#5 - 77
#6 - 83
#7 - 87
#8 - 91
# Usaremos 7 componentes ya que se tiene el 87%
```

```
Porporción de varianza: [0.3500593  0.1593905  0.10995289 0.08550723 0.068725
05 0.05743784
 0.04511767 0.03364819 0.03102975 0.02587329 0.01543708 0.01155555
 0.00626566]
Varianza acumulada: 0.8761904741976413
```

Otra forma de visualizar esto es con una gráfica del número de componentes contra la varianza acumulada, aunque resultaría complicado conocer el valor especifico si es que lo necesitamos.

```
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Número de componentes')
plt.ylabel('Varianza acumulada')
plt.grid()
plt.show()
```



El ultimo paso es **examinar la proporción de cargas** esto lo observaremos viendo los valores absolutos de nuestras componentes principales, entre mayor sea su valor mayor será su importancia y podremos considerarla como componente principal, para este proceso definiremos un mínimo de valor que deberán tener para considerarse componente principal, tomaremos aquellas variables cuyo valor sea mínimo 50% (0.5).

Primero revisaremos los primeros siete registros, ya que son las componentes que decidimos tomar, y buscaremos aquellos valores que sean mayores que 0.5 y los que cumplan dicha condición, esa variable será parte de nuestros componentes principales.

En este caso los registros del 0-2 no cuentan con valores mayores al límite, pero en otros registros encontramos los siguientes componentes fuertes que se convertirán en nuestras componentes principales.

- Overwieght (3)
- Have_undestanding_parents (4)
- Currently_Drink_Alcohol (5)
- No_close_friends (6)

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

	Currently_Drink_Alcohol	Really_Get_Drunk	Overwieght	Use_Marijuana	Have_Understanding_Parents	Missed_classes_without_permssion	Had_sexual_relatior
0	0.130994	0.282154	0.106451	0.339411	0.055107	0.264697	0.36073
1	0.367308	0.458837	0.088979	0.271897	0.325082	0.258208	0.29424
2	0.040052	0.110844	0.107717	0.066784	0.467231	0.439647	0.05652
3	0.261818	0.049653	0.835182	0.047662	0.291240	0.068342	0.19413
4	0.231178	0.022787	0.289008	0.178099	0.521135	0.155133	0.05030
5	0.808721	0.316475	0.183738	0.095724	0.003690	0.007027	0.09139
6	0.139191	0.065199	0.107237	0.007137	0.189062	0.137137	0.03189
7	0.046344	0.004429	0.106135	0.306040	0.420688	0.635196	0.05210
8	0.059344	0.342448	0.097074	0.730597	0.068135	0.254145	0.14920
9	0.199055	0.056388	0.174822	0.101378	0.017825	0.152917	0.63347
10	0.044045	0.371433	0.120177	0.006911	0.278756	0.255580	0.12888
11	0.016401	0.069853	0.277742	0.351278	0.089455	0.037432	0.11832
12	0.058512	0.571724	0.000301	0.045309	0.114773	0.253403	0.52309

Este proceso nos recomienda eliminar aquellas variables que no son componentes principales, por lo que nuestra tabla final de datos quedaría de la siguiente manera. Únicamente con cuatro columnas.

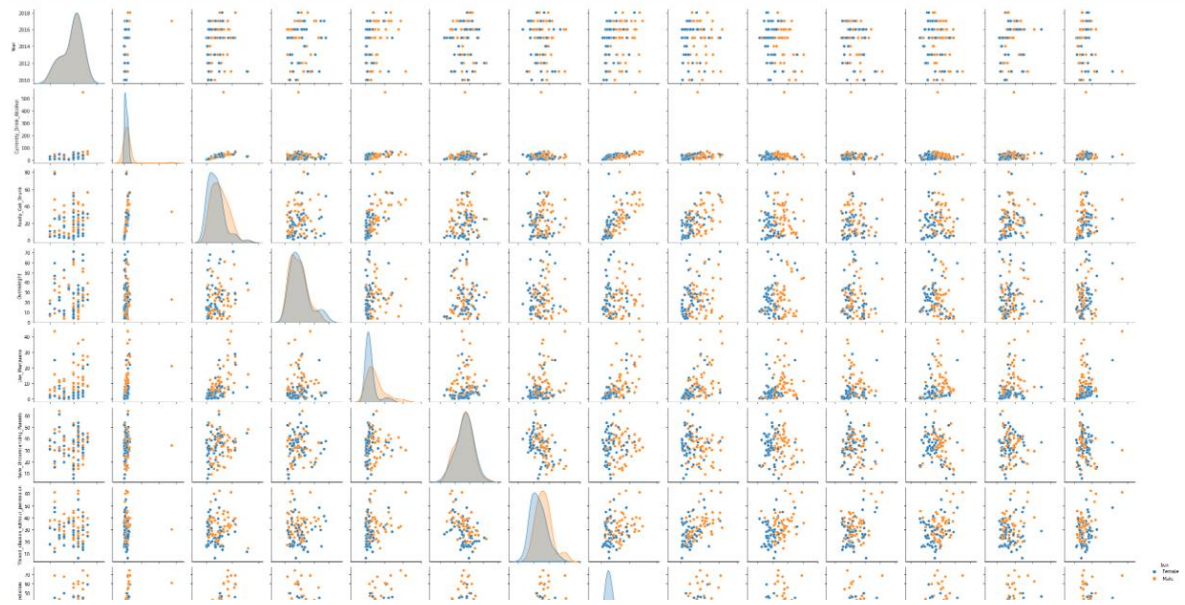
```
DatosSuicidioACP = Suicidio.drop(columns=['Country','Year','Age Group','Sex','Really_Get_Drunk','Use_Marijuana','Missed_classes_
DatosSuicidioACP
```

	Currently_Drink_Alcohol	Overwieght	Have_Understanding_Parents	No_close_friends
0	50.3	27.8	41.5	4.8
1	44.9	39.1	44.5	5.5
2	67.2	22.5	37.1	6.3
3	68.1	27.9	39.8	6.6
4	49.3	35.9	46.2	6.1
...	...	...	...	...
101	5.8	13.6	20.2	14.3
102	32.2	60.5	36.3	4.1
103	24.4	63.0	36.3	3.3
104	48.3	57.8	36.5	7.3
105	42.9	70.6	37.8	1.5

106 rows × 4 columns

Otra forma de hacer este análisis para reducir la dimensionalidad de la tabla, es con un análisis correlacional de los datos, esto puede hacerse de diversas maneras, una de ellas es mostrando en mapas de dispersión todas las correlacione existentes, esto es algo complicado de visualizar ya que asignarle un valor y encontrar las relaciones fuertes es más complicado. Se muestran los mapas de dispersión usando a la variable Sex como punto de comparativa.

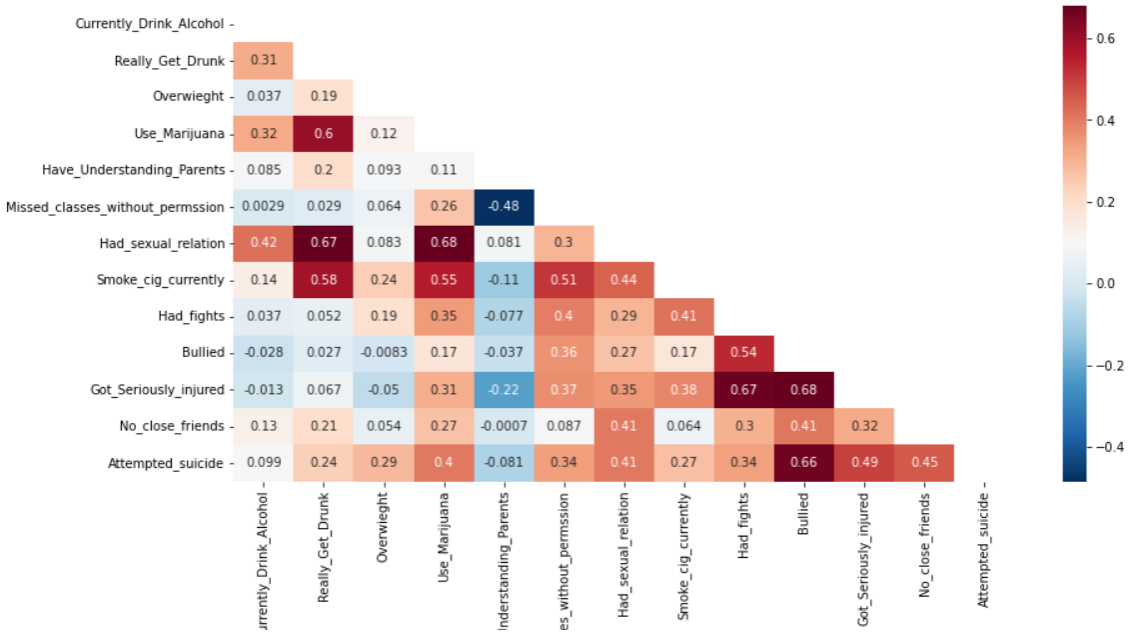
```
SuicidioRec = Suicidio.drop(columns=['Year'])
sns.pairplot(Suicidio, hue='Sex')
plt.show()
```



Y otro de los métodos que hemos utilizado es obtener la matriz de correlación para posteriormente obtener el mapa de calor y de igual forma buscar relaciones fuertes y así reducir la dimensionalidad de nuestros datos, al obtener el mapa de calor y eliminar variables, nos quedaremos con las siguientes variable seleccionadas.

- Currently_Drink_Alcohol
- Overwieght
- Have_Understanding_Parents
- Missed_classes_without_permssion
- Had_sexual_relation
- Smoke_cig_currently
- Got_Seriously_injured
- No_close_friends
- Attempted_suicide

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



Por lo que, si nos quedamos con las variables seleccionadas, obtendremos una matriz con nueve registros, cinco registros mas que con el método anterior.

```
DatosCancerACD = Suicidio.drop(columns=['Country','Year','Age Group','Sex','Really_Get_Drunk', 'Use_Marijuana', 'Had_fights', 'Bu
DatosCancerACD
```

	Currently_Drink_Alcohol	Overwieght	Have_Understanding_Parents	Missed_classes_without_permssion	Had_sexual_relation	Smoke_cig_currently	Got_Serio
0	50.3	27.8	41.5	24.7	25.7	16.8	
1	44.9	39.1	44.5	27.9	38.4	12.1	
2	67.2	22.5	37.1	34.0	59.1	28.5	
3	68.1	27.9	39.8	39.4	68.6	28.0	
4	49.3	35.9	46.2	32.0	43.5	17.0	
...	...	...	...	...	...	...	
101	5.8	13.6	20.2	44.6	8.4	8.2	
102	32.2	60.5	36.3	32.7	32.0	25.7	
103	24.4	63.0	36.3	16.2	13.6	29.9	
104	48.3	57.8	36.5	38.9	55.7	40.0	
105	42.9	70.6	37.8	37.2	22.4	41.7	

106 rows × 9 columns

Conclusión

A lo largo de estas practicas es interesante como existe una variedad de métodos para reducir las dimensiones de nuestra tabla, además de que nos todos los métodos son estrictos ya que muchas veces estos funcionan como sugerencias, ya que depende el análisis que nosotros necesitamos hacer y qué tipo de datos son los que nos interesan.

Pero estos métodos resultan muy fáciles de usar si buscamos reducir las dimensiones de forma general, y en caso de que nos interese una variable en específico podríamos agregarla sin tener complicaciones.

Liga GitHub Practica 6

Cuaderno de Python de la practica 6

https://github.com/OsvaldoIG/MineriaDatos/tree/main/P6