

Wisconsin Breast Cancer Dataset

Este Dataset tiene aplicación médica, enfocada en el diagnóstico de cáncer de mama. Entre las variables predictivas se encuentran: el radio, la textura, perímetro, área, entre otros (10 features). Se calcula la media, la desviación estándar y "peor" (media de los 3 valores más grandes) de cada feature, dando un total de 30. La variable objetivo es Diagnosis (Maligno y Benigno).

Data Cleaning

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Cargar el dataset (Añadiendo los títulos de las columnas,
# proveniente de la documentación del dataset)
column_names = ['id', 'diagnosis', 'mean_radius', 'mean_texture',
'mean_perimeter', 'mean_area', 'mean_smoothness', 'mean_compactness',
'mean_concavity', 'mean_concave_points', 'mean_symmetry',
'mean_fractal_dimension', 'radius_se', 'texture_se', 'perimeter_se',
'area_se', 'smoothness_se', 'compactness_se', 'concavity_se',
'concave_points_se', 'symmetry_se', 'fractal_dimension_se',
'worst_radius', 'worst_texture', 'worst_perimeter', 'worst_area',
'worst_smoothness', 'worst_compactness', 'worst_concavity',
'worst_concave_points', 'worst_symmetry', 'worst_fractal_dimension']

df = pd.read_csv('wdbc.data', names=column_names)

# Limpieza de la columna ID (irrelevante) y mapeo de la columna
# 'diagnosis' para convertirla en valores numéricos
df = df.drop(columns=['id'])
df['diagnosis'] = df['diagnosis'].map({'M': 1, 'B': 0})

# Validación de datos y manejo de valores faltantes
# Conteo por columna y suma de todas las columnas.
if df.isnull().sum().sum():
    print("Existen valores faltantes en el dataset.")
else:
    print("No existen valores faltantes en el dataset.")

No existen valores faltantes en el dataset.

# Visualización del dataset para entender la distribución de las
# variables y su relación con el diagnóstico
df

    diagnosis  mean_radius  mean_texture  mean_perimeter
mean_area  \
```

0	1	17.99	10.38	122.80	1001.0
1	1	20.57	17.77	132.90	1326.0
2	1	19.69	21.25	130.00	1203.0
3	1	11.42	20.38	77.58	386.1
4	1	20.29	14.34	135.10	1297.0
..
564	1	21.56	22.39	142.00	1479.0
565	1	20.13	28.25	131.20	1261.0
566	1	16.60	28.08	108.30	858.1
567	1	20.60	29.33	140.10	1265.0
568	0	7.76	24.54	47.92	181.0
mean_smoothness mean_compactness mean_concavity					
mean_concave_points \					
0	0.11840	0.27760	0.30010		
0.14710					
1	0.08474	0.07864	0.08690		
0.07017					
2	0.10960	0.15990	0.19740		
0.12790					
3	0.14250	0.28390	0.24140		
0.10520					
4	0.10030	0.13280	0.19800		
0.10430					
..
..					
564	0.11100	0.11590	0.24390		
0.13890					
565	0.09780	0.10340	0.14400		
0.09791					
566	0.08455	0.10230	0.09251		
0.05302					
567	0.11780	0.27700	0.35140		
0.15200					
568	0.05263	0.04362	0.00000		
0.00000					
mean_symmetry ... worst_radius worst_texture worst_perimeter					
\					

0	0.2419	...	25.380	17.33	184.60
1	0.1812	...	24.990	23.41	158.80
2	0.2069	...	23.570	25.53	152.50
3	0.2597	...	14.910	26.50	98.87
4	0.1809	...	22.540	16.67	152.20
..
564	0.1726	...	25.450	26.40	166.10
565	0.1752	...	23.690	38.25	155.00
566	0.1590	...	18.980	34.12	126.70
567	0.2397	...	25.740	39.42	184.60
568	0.1587	...	9.456	30.37	59.16
0	worst_area	worst_smoothness	worst_compactness	worst_concavity	
1	2019.0	0.16220	0.66560	0.7119	
2	1956.0	0.12380	0.18660	0.2416	
3	1709.0	0.14440	0.42450	0.4504	
4	567.7	0.20980	0.86630	0.6869	
..
564	1575.0	0.13740	0.20500	0.4000	
565	2027.0	0.14100	0.21130	0.4107	
566	1731.0	0.11660	0.19220	0.3215	
567	1124.0	0.11390	0.30940	0.3403	
568	1821.0	0.16500	0.86810	0.9387	
	268.6	0.08996	0.06444	0.0000	
0	worst_concave_points	worst_symmetry	worst_fractal_dimension		
1	0.2654	0.4601	0.11890		
	0.1860	0.2750	0.08902		

2	0.2430	0.3613	0.08758
3	0.2575	0.6638	0.17300
4	0.1625	0.2364	0.07678
..
564	0.2216	0.2060	0.07115
565	0.1628	0.2572	0.06637
566	0.1418	0.2218	0.07820
567	0.2650	0.4087	0.12400
568	0.0000	0.2871	0.07039

[569 rows x 31 columns]

Exploratory Data Analysis

```
# Resumen estadístico del dataset
df.describe()
```

	diagnosis	mean_radius	mean_texture	mean_perimeter
mean_area \count	569.000000	569.000000	569.000000	569.000000
mean	0.372583	14.127292	19.289649	91.969033
std	0.483918	3.524049	4.301036	24.298981
min	0.000000	6.981000	9.710000	43.790000
25%	0.000000	11.700000	16.170000	75.170000
50%	0.000000	13.370000	18.840000	86.240000
75%	1.000000	15.780000	21.800000	104.100000
max	1.000000	28.110000	39.280000	188.500000

	mean_smoothness	mean_compactness	mean_concavity
mean_concave_points \count	569.000000	569.000000	569.000000
mean	0.096360	0.104341	0.088799
std	0.014064	0.052813	0.079720
min	0.052630	0.019380	0.000000
25%	0.086370	0.064920	0.029560

50%	0.095870	0.092630	0.061540
0.033500			
75%	0.105300	0.130400	0.130700
0.074000			
max	0.163400	0.345400	0.426800
0.201200			
mean_symmetry ... worst_radius worst_texture			
worst_perimeter \			
count	569.000000	569.000000	569.000000
569.000000			
mean	0.181162	16.269190	25.677223
107.261213			
std	0.027414	4.833242	6.146258
33.602542			
min	0.106000	7.930000	12.020000
50.410000			
25%	0.161900	13.010000	21.080000
84.110000			
50%	0.179200	14.970000	25.410000
97.660000			
75%	0.195700	18.790000	29.720000
125.400000			
max	0.304000	36.040000	49.540000
251.200000			
worst_area worst_smoothness worst_compactness			
worst_concavity \			
count	569.000000	569.000000	569.000000
569.000000			
mean	880.583128	0.132369	0.254265
0.272188			
std	569.356993	0.022832	0.157336
0.208624			
min	185.200000	0.071170	0.027290
0.000000			
25%	515.300000	0.116600	0.147200
0.114500			
50%	686.500000	0.131300	0.211900
0.226700			
75%	1084.000000	0.146000	0.339100
0.382900			
max	4254.000000	0.222600	1.058000
1.252000			
worst_concave_points worst_symmetry worst_fractal_dimension			
count	569.000000	569.000000	569.000000
mean	0.114606	0.290076	0.083946
std	0.065732	0.061867	0.018061
min	0.000000	0.156500	0.055040

25%	0.064930	0.250400	0.071460
50%	0.099930	0.282200	0.080040
75%	0.161400	0.317900	0.092080
max	0.291000	0.663800	0.207500

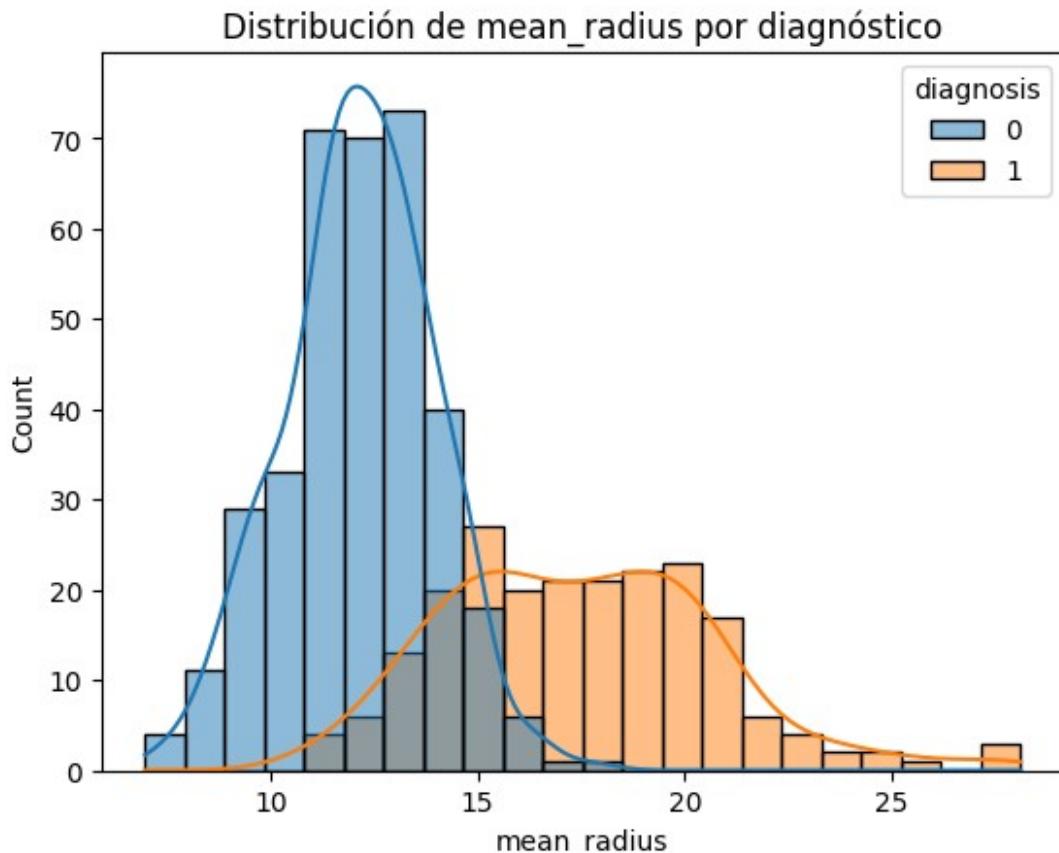
[8 rows x 31 columns]

Las escalas de los datos son muy diferentes, esto nos da la indicación de estandarizar los datos para poder hacer uso del PCA.

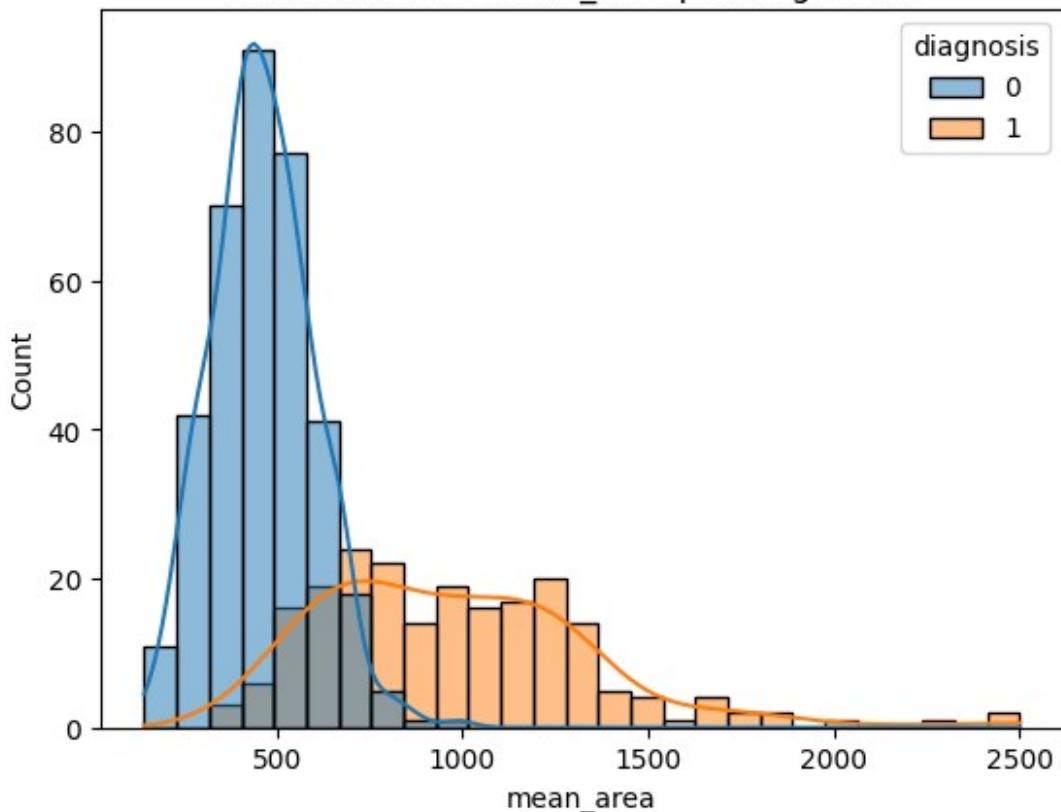
Variables como el mean area tiene valores arriba de 2000, mientras que mean smoothness tiene valores máximos menores de .2

```
# Histograma de las variables 'mean_radius' y 'mean_area' para
observar su distribución
sns.histplot(data = df, x='mean_radius', hue='diagnosis', kde=True)
plt.title('Distribución de mean_radius por diagnóstico')
plt.show()

sns.histplot(data = df, x='mean_area', hue='diagnosis', kde=True)
plt.title('Distribución de mean_area por diagnóstico')
plt.show()
```

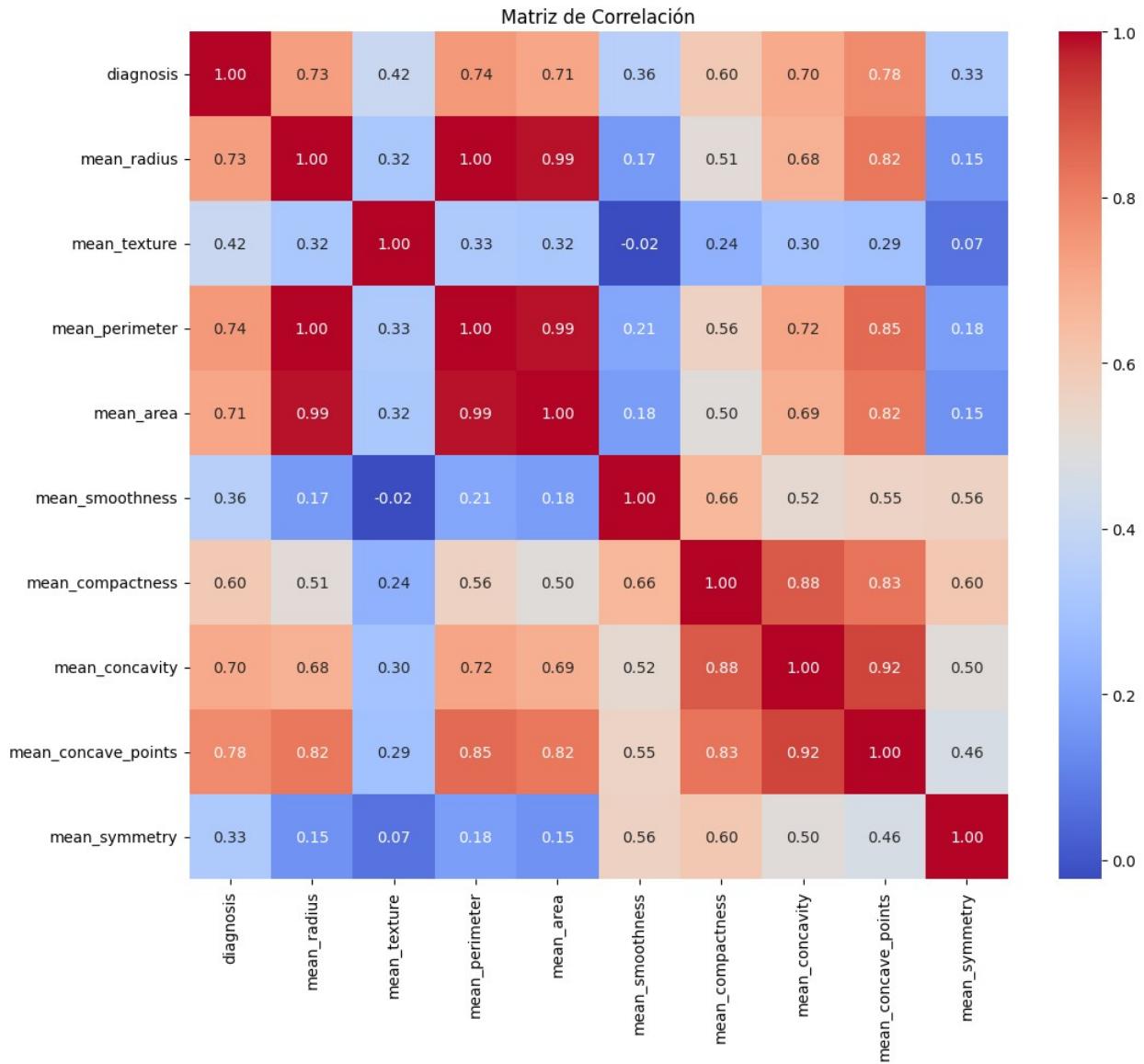


Distribución de mean_area por diagnóstico



Las células cancerígenas tienden a tener núcleos más grandes. La separación entre las curvas podría indicar un alto valor predictivo de estas variables.

```
# Análisis de correlación entre las variables para identificar
# posibles relaciones
correlation_matrix = df.iloc[:, 0:10].corr() # Primeros 10 valores
# para claridad
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f",
cmap='coolwarm')
plt.title('Matriz de Correlación')
plt.show()
```



Es posible notar una correlación de más de .8 entre varios grupos de características.

Por ejemplo, área, radio y perímetro. Y Compactness, concavidad y puntos concavos.

Esta fuerte correlación nos sirve como indicador para el uso del PCA y "simplificar" la dimensionalidad

Data Standardization

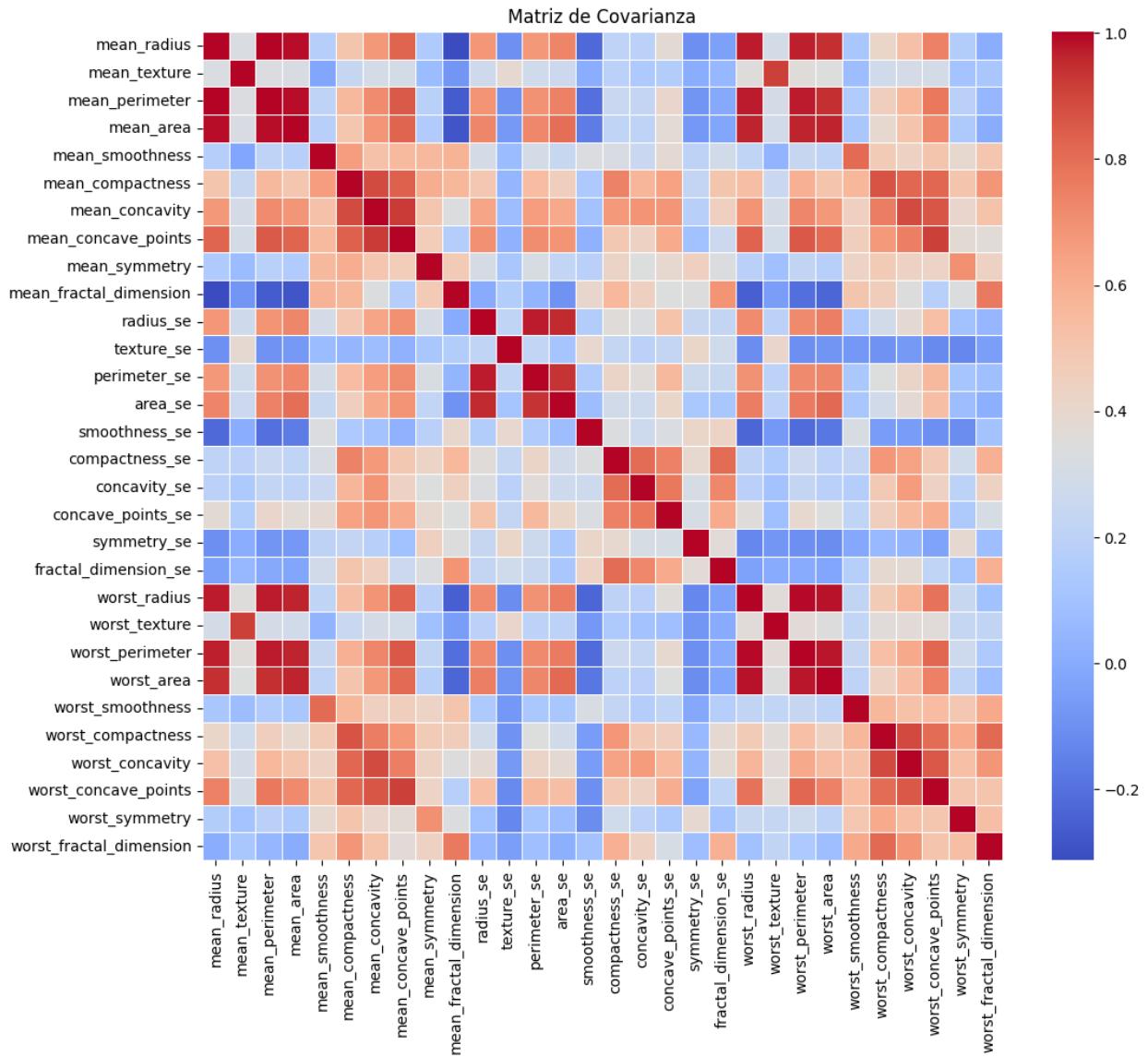
La estandarización de datos es importante debido a que el PCA se basa en la varianza. Al existir una gran diferencia entre el rango de valores de las características, se puede asumir que una variable es más importante de manera errónea debido a su magnitud.

```
import numpy as np
from sklearn.preprocessing import StandardScaler

# Estandarización
X = df.drop(columns=['diagnosis'])
y = df['diagnosis']
X_scaled = StandardScaler().fit_transform(X)

# Matriz de covarianza
cov_matrix = np.cov(X_scaled, rowvar=False)

cov_df = pd.DataFrame(cov_matrix, index=X.columns, columns=X.columns)
plt.figure(figsize=(12, 10))
sns.heatmap(cov_df, annot=False, cmap='coolwarm', linewidths=0.5)
plt.title('Matriz de Covarianza')
plt.show()
```



Como se notó anteriormente, las características geométricas tienen una fuerte relación entre si. De igual manera, es posible notar que hay características fuertemente independientes de las demás, como lo son 'smoothness' y 'fractal_dimension'.

Estas podrían aportar información única.

```
# Eigenvalores y eigenvectores
eigenvalues, eigenvectors = np.linalg.eig(cov_matrix)
print("Eigenvalores:\n", eigenvalues)
```

Eigenvalores:

```
[1.33049908e+01 5.70137460e+00 2.82291016e+00 1.98412752e+00
 1.65163324e+00 1.20948224e+00 6.76408882e-01 4.77456255e-01
 4.17628782e-01 3.51310875e-01 2.94433153e-01 2.61621161e-01
 2.41782421e-01 1.57286149e-01 9.43006956e-02 8.00034045e-02]
```

```
5.95036135e-02 5.27114222e-02 4.95647002e-02 1.33279057e-04  
7.50121413e-04 1.59213600e-03 6.91261258e-03 8.19203712e-03  
1.55085271e-02 1.80867940e-02 2.43836914e-02 2.74877113e-02  
3.12142606e-02 3.00256631e-02]
```

Los autovectores explican la dirección de los componentes principales, mientras que los autovalores definen la magnitud de cada eje. En este caso, los autovalores representan la cantidad de varianza que explica cada eje.

```
# Varianza explicada en porcentaje  
explained_variance = eigenvalues / np.sum(eigenvalues) * 100  
  
for i, val in enumerate(explained_variance):  
    if eigenvalues[i] > 1:  
        print(f"Componente {i+1}: {val:.2f}% de la varianza explicada,  
Autovalor mayor a 1: {eigenvalues[i]:.2f}")  
        accumulated_variance = np.sum(explained_variance[:i+1])  
        k = i + 1  
    else:  
        print(f"Componente {i+1}: {val:.2f}% de la varianza explicada,  
Autovalor menor a 1: {eigenvalues[i]:.2f}")
```

```
Componente 1: 44.27% de la varianza explicada, Autovalor mayor a 1:  
13.30  
Componente 2: 18.97% de la varianza explicada, Autovalor mayor a 1:  
5.70  
Componente 3: 9.39% de la varianza explicada, Autovalor mayor a 1:  
2.82  
Componente 4: 6.60% de la varianza explicada, Autovalor mayor a 1:  
1.98  
Componente 5: 5.50% de la varianza explicada, Autovalor mayor a 1:  
1.65  
Componente 6: 4.02% de la varianza explicada, Autovalor mayor a 1:  
1.21  
Componente 7: 2.25% de la varianza explicada, Autovalor menor a 1:  
0.68  
Componente 8: 1.59% de la varianza explicada, Autovalor menor a 1:  
0.48  
Componente 9: 1.39% de la varianza explicada, Autovalor menor a 1:  
0.42  
Componente 10: 1.17% de la varianza explicada, Autovalor menor a 1:  
0.35  
Componente 11: 0.98% de la varianza explicada, Autovalor menor a 1:  
0.29  
Componente 12: 0.87% de la varianza explicada, Autovalor menor a 1:  
0.26  
Componente 13: 0.80% de la varianza explicada, Autovalor menor a 1:  
0.24
```

```
Componente 14: 0.52% de la varianza explicada, Autovalor menor a 1:  
0.16  
Componente 15: 0.31% de la varianza explicada, Autovalor menor a 1:  
0.09  
Componente 16: 0.27% de la varianza explicada, Autovalor menor a 1:  
0.08  
Componente 17: 0.20% de la varianza explicada, Autovalor menor a 1:  
0.06  
Componente 18: 0.18% de la varianza explicada, Autovalor menor a 1:  
0.05  
Componente 19: 0.16% de la varianza explicada, Autovalor menor a 1:  
0.05  
Componente 20: 0.00% de la varianza explicada, Autovalor menor a 1:  
0.00  
Componente 21: 0.00% de la varianza explicada, Autovalor menor a 1:  
0.00  
Componente 22: 0.01% de la varianza explicada, Autovalor menor a 1:  
0.00  
Componente 23: 0.02% de la varianza explicada, Autovalor menor a 1:  
0.01  
Componente 24: 0.03% de la varianza explicada, Autovalor menor a 1:  
0.01  
Componente 25: 0.05% de la varianza explicada, Autovalor menor a 1:  
0.02  
Componente 26: 0.06% de la varianza explicada, Autovalor menor a 1:  
0.02  
Componente 27: 0.08% de la varianza explicada, Autovalor menor a 1:  
0.02  
Componente 28: 0.09% de la varianza explicada, Autovalor menor a 1:  
0.03  
Componente 29: 0.10% de la varianza explicada, Autovalor menor a 1:  
0.03  
Componente 30: 0.10% de la varianza explicada, Autovalor menor a 1:  
0.03
```

A partir de los autovalores podemos interpretar cuantos componentes capturan la mayoría de la varianza. Solo serán considerados los componentes con Varianza mayor a 1, ya que se considera que todos los componentes originales tienen una varianza de 1. Aquellos valor mayor a 1, significan que condensaron información de múltiples variables, mientras que los que tengan valor menor a 1 representan que cuentan con menos información que los originales.

En este caso, los primeros 6 componentes son los que mejor representarían.

```
print(f"Varianza acumulada hasta el componente {k}:\n{accumulated_variance:.2f}%")  
  
Varianza acumulada hasta el componente 6: 88.76%  
  
# Matriz de transformación  
# Asegurar que los eigenvectores estén ordenados según los
```

```

eigenvalores
idx = np.argsort(eigenvalues)[::-1]
eigenvectors_sorted = eigenvectors[:, idx]

W = eigenvectors_sorted[:, :k]

W_df = pd.DataFrame(W, index=df.drop(columns=['diagnosis']).columns,
                     columns=[f'PC{i+1}' for i in range(k)])
print("Matriz de transformación W:\n", W_df.head(10))

Matriz de transformación W:


|                        | PC1       | PC2       | PC3       | PC4       |
|------------------------|-----------|-----------|-----------|-----------|
| PC5 \                  |           |           |           |           |
| mean_radius            | 0.218902  | -0.233857 | -0.008531 | 0.041409  |
| 0.037786               |           |           |           |           |
| mean_texture           | 0.103725  | -0.059706 | 0.064550  | -0.603050 |
| 0.049469               |           |           |           |           |
| mean_perimeter         | 0.227537  | -0.215181 | -0.009314 | 0.041983  |
| 0.037375               |           |           |           |           |
| mean_area              | 0.220995  | -0.231077 | 0.028700  | 0.053434  |
| 0.010331               |           |           |           |           |
| mean_smoothness        | 0.142590  | 0.186113  | -0.104292 | 0.159383  |
| 0.365089               |           |           |           |           |
| mean_compactness       | 0.239285  | 0.151892  | -0.074092 | 0.031795  |
| 0.011704               |           |           |           |           |
| mean_concavity         | 0.258400  | 0.060165  | 0.002734  | 0.019123  |
| 0.086375               |           |           |           |           |
| mean_concave_points    | 0.260854  | -0.034768 | -0.025564 | 0.065336  |
| 0.043861               |           |           |           |           |
| mean_symmetry          | 0.138167  | 0.190349  | -0.040240 | 0.067125  |
| 0.305941               |           |           |           |           |
| mean_fractal_dimension | 0.064363  | 0.366575  | -0.022574 | 0.048587  |
| 0.044424               |           |           |           |           |
|                        | PC6       |           |           |           |
| mean_radius            | 0.018741  |           |           |           |
| mean_texture           | -0.032179 |           |           |           |
| mean_perimeter         | 0.017308  |           |           |           |
| mean_area              | -0.001888 |           |           |           |
| mean_smoothness        | -0.286374 |           |           |           |
| mean_compactness       | -0.014131 |           |           |           |
| mean_concavity         | -0.009344 |           |           |           |
| mean_concave_points    | -0.052050 |           |           |           |
| mean_symmetry          | 0.356458  |           |           |           |
| mean_fractal_dimension | -0.119431 |           |           |           |


```

La transformada sirve para proyectar los datos del espacio original hacia un subespacio simplificado. (Pasar de un espacio 569x30 a uno 30x6 que representa la misma información)

$$X' = X_{\text{scaled}} * W$$

```

# Proyección de los datos originales en el nuevo espacio de
# características
X_pca = X_scaled.dot(W)

# Dataframe con los componentes principales
pca_df = pd.DataFrame(X_pca, columns=[f'PC{i+1}' for i in range(k)])
pca_df['diagnosis'] = y.values

pca_df

```

	PC1	PC2	PC3	PC4	PC5	PC6
diagnosis						
0	9.192837	1.948583	-1.123166	3.633731	1.195110	1.411424
1	2.387802	-3.768172	-0.529293	1.118264	-0.621775	0.028656
1	5.733896	-1.075174	-0.551748	0.912083	0.177086	0.541452
1	7.122953	10.275589	-3.232790	0.152547	2.960878	3.053422
1	3.935302	-1.948072	1.389767	2.940639	-0.546747	-1.226495
1
...
564	6.439315	-3.576817	2.459487	1.177314	0.074824	-2.375193
1	3.793382	-3.584048	2.088476	-2.506028	0.510723	-0.246710
1	1.256179	-1.902297	0.562731	-2.089227	-1.809991	-0.534447
1	10.374794	1.672010	-1.877029	-2.356031	0.033742	0.567936
1	-5.475243	-0.670637	1.490443	-2.299157	0.184703	1.617837
0						
[569 rows x 7 columns]						

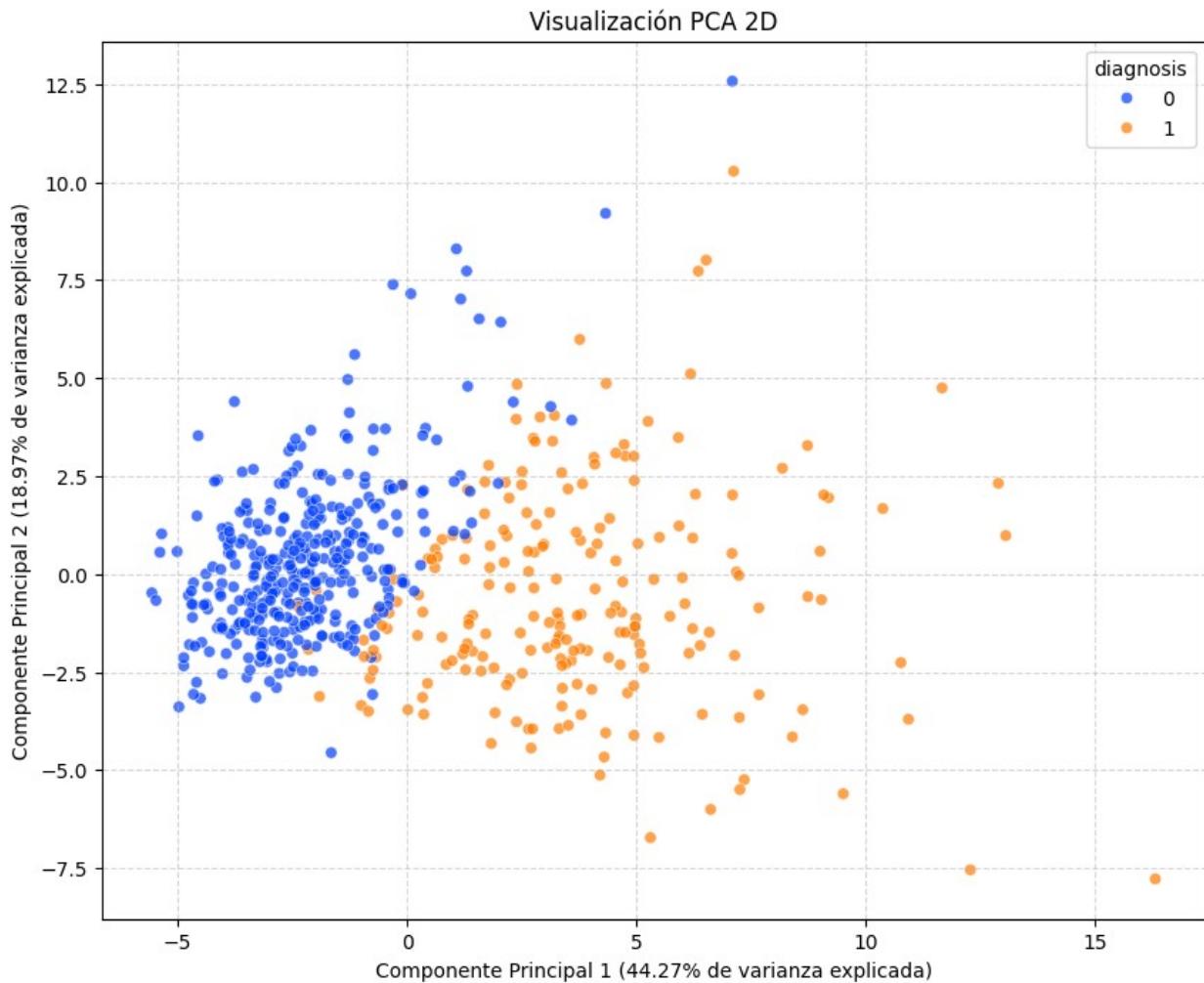
2D Visualization

```

plt.figure(figsize=(10, 8))
sns.scatterplot(data=pca_df, x='PC1', y='PC2', hue='diagnosis',
palette='bright', alpha=0.7)
plt.title('Visualización PCA 2D')
plt.xlabel(f'Componente Principal 1 ({explained_variance[0]:.2f}% de
varianza explicada)')
plt.ylabel(f'Componente Principal 2 ({explained_variance[1]:.2f}% de
varianza explicada)')

```

```
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```



Se puede notar una fuerte separación entre las muestras malignas y benignas. Esta clusterización nos da un indicador de que el subespacio es representativo.

3D Visualization

```
import plotly.express as px

fig= px.scatter_3d(pca_df, x='PC1', y='PC2', z='PC3',
                    color='diagnosis', title='Visualización PCA 3D',
                    labels={'PC1': f'Componente Principal 1 ({explained_variance[0]:.2f}% varianza)',
                            'PC2': f'Componente Principal 2 ({explained_variance[1]:.2f}% varianza)',
                            'PC3': f'Componente Principal 3 ({explained_variance[2]:.2f}% varianza)'}
```

```
{explained_variance[2]:.2f}% varianza')})  
fig.show()  
  
{ "config":{ "plotlyServerURL": "https://plot.ly", "data": [ { "hovertemplate": "Componente Principal 1 (44.27% varianza)=%{x}<br>Componente Principal 2 (18.97% varianza)= %{y}<br>Componente Principal 3 (9.39% varianza)= %{z}<br>diagnosis=%{marker.color}<br><br>", "legendgroup": "", "marker": { "color": "#B2EBF2", "size": 100, "symbol": "circle"}, "name": "bdata", "scene": "scene", "showlegend": false, "type": "scatter3d", "x": { "bdata": "xhksgrtiIkCcZ/ LTNxoDQDxqoIGC7xZApGRsced9HECXoFKnf3sPQF0QmQS/ CgNAvCndpjvpAUDNfyWveSUBQIGQMLU+ZglASvN1TzBoGUBPRkDc607pvy1UwSp0NQVA0c Ek1rxeIECN6uwWZOXVP5xfz6aYXhFAG2j7z3hNECAq2BYcXLNP2QStg0LrBFA3BqN8XjL E0A0PlPyAsvzxNvtf4lQPm// iXKxWx1DMDGvRHPzu4SQJ4syV6H1RBApvizWWzME0Btyjmp7WQcQFdyY48EFQxAGJU0ry6 DCEC0vW1AcAcQQEI3oJzocfs/qpfhSN88GEC39CGQxzchQB5nA/ QYOBJAxnNhj2D2E0B5WwBa+uEHQFSMKieGIQZAgxS06jq99D+EFZAFmP4NwMXoDTEu80+/ 5byNKwt96D/vCeAXJzUBwJ4b/9SQ5Le/uMovSw4tIkAKJGEYzLHvP/2+Bx13ydK/ PBT2PRWGFUAclf0051MSw0gL32s6M/s/ 7cdn3AXAcCX0GLbZBT5v9r8cN7eVwzA0AZDAzRDCcD7/ My8mZoLwBlfTNwfawpARgUZL3uW5b9EShYsgt0GwLgJwwBPmBJA32Y4FjdqAUCQ9NxKjMU NwHwutN45ixDABFgWAjEWA8BCQduKjaEEwGZnS7JUCRNAK1vNxFeDAsAepTp40xD7P3bmzfMz//w/ Qf4365JJBsB1SScZMSYMwEGl5TbsURFAg2M1sRkXCsBIrHJRA6MFQBpUhbDos90/ oweAU8sDFkD0BFJQPibXv6L7HjExCgXAoxQLQ8bz9j9vzB3YXrTqv46HnSJw7BhANN0xxG FWJ0CBCeR0MCwAwJfdwx/TNvq/LEfu0eyt8j+s7t4x2oklQEi/ ZoSlIxRAvY8+i01lAcDBVHdaTk4KQD0BqjeGceM/ 6aAIaC0rDEDsWnWJn+TtvysW5DNXJ/Q/KsGee4dp+b/U0La06yjQP/ ynMzaKxwbAPHCVh29w/7/qqucB+TQGQLfV7Me8lRFAi5Pv65W8BMA/ dNujFTwPwC0aaM68iwbAUJ0hqJK44z/uXglb+7Pbvxs/ +HpZ0BLA2R0hQOGVC8DpKy2Tf00AwIWjMqrjhXACwY0/hm2CUDm0mvsB/ rnvz+AFPwkrQnAvBkXAbwcKkDw6v8UVtQBwNE+42VR8QPA7s15bSx1ur8BvzBp+LoMQJGy zjZLlfW/7BVtk7gUBMABo6LARF4AwHDbYTPMxgDA4P/ hV1ziCECYoCner9MTQNpgUk0C0+s/Sj/ Dyl26B8DdCrAGTqwKQL4ABJoKyilAGSkTpe6p6L87n3x1wJ4BwCitxVG1JQnAFZethLVx5 L9E+W4/xbT+P0lCko25ivY/Adkf0S0TEkCteeA3Gwr+v74km6HB6fY/ShFcPFyk9T+ +kWuWEbHmvzsChVwPXwFA+ho4Wi0h/7/
```

VZTy6as4EwKlemClmEArcpcaFpdBDkDYxgHNX9//v11oInBeDxTANo6IJjSF9T+fNc/
50qUDw0jhXaBZf2/h7D5Ms5iEcBNkJ0SNHPnv8WKthEyMwNAZgRsTIco2T/
3DswxlGPav03zEww/
ZgjAGV8atb0U979cHTQ2r0u1P+TC18uLXXXAbu6+FH7yDcCFebu0dgPvv/
STvNKJWQPAEELW5TRnEEA2Mwu/
IQ3ov7hPqlG3PA3A2Fc1Yom4EsA7Cjg6ByHjvwLyNt9AGQtAGBvw3uZGECskdsRVDP1v6
715DAD+hVAM3FciK76B8CqyJWpA4wRwJSahyAMa/M/X/YKPFVHFEAvocdDqqT/
v63pJId3HgTA/jqNINJR7r/
Ug3T9EI4GQBGqTTU03wvA9hapAYqiEsADVkGUDmgVwAK6NU//u/I/
6efjl4GmB0C2dv5wHucTwDuT0fGPQRDAC65sqXwGI0ADKyozqwMiQIH6NYi7S0g/
1faBV5GBcBt6B0sEvbYv0UlGmPOHA/
AucTPgCFalz+iC+ST568GwEp9nPlewrA24cdacLbCsBLmyDiqxcaQAInSeHxe/u/
DDCBRVBDFsAcQmNTfnn8P7z6EDFS5ARANpXnD2SNBsAeCbeFzfDQ0o4IAy0bvQ/
5MJAsBvJC0AKybpcVvx0Pw70esUsk/e/5i8R5jBk+j/
KkzrbuxIiQAT003XmNxJAXln5GYL/6L/
j8A5tTI7Mv0EQMzGgowvA6Mm08i6U3D+weEeCwR/UP3laC6A5eQDARl8Gb2I8E0B/
oQfyW/0HwHTjIxzuKShAj/rDNEl6CkCtzW0S9BEEQP05ivj0FvU/
YD1RGWLh2L8Js1XiMg4GwNi6+Rk6xhNAmj9LjsxEUCL1k0D7vwCwAXtqhg58+6/
AHZNmG/0B8Bp0DiMGdr8Px706iau0wPAg1xH90lu6r+tS6gSVygPwPUzLvHY7ti/
smPzMjBR978Q4HY8ZwsGQGetyonbDgpA9ArNf5kTEMAv3mVfS2kNwEqF0e/
UhwpAXyx2ZnwLEMbhxe0IgqwCwJjEgQzFziBATB0odjguBUCc5NHwDNHlvwyIJeFnzQpAg
aunIl/SAcA5vLXzC+kRwBjwrhN1KfU/pm8/0/eZ/
b8QreuEUV00QGwNILQjQALAYm2CbYXUCC2Rb7xFmrwP/W/
6QgTqQTAUcyZvhbjA8C3FtZTS5McQDjX6niFQgjAL2sRCA1XHEALm0KtjAb1P7WLAU11sQ
1AKrXrSn143T/15R3SEpIZQG/
SR50tBRVAMPblsbx3IUCcxTWr0YcNQFJMv2SoBgtAZWsgW//i6r/
qTPUmZUYKQL53n0lRiv6/mTV5kiXW+T8fiSCLmXoaQIQZFwhQb/e/
13BYX+MYAMBzo5wuVEYCwKUwCbH5q+e/
BQejUrgGEsBMJ6X5IEUJwIMd8IZgtB5AUy1lUfMcDcCk7ZaMEZfVPzgrgbHibPS/
XIEbglFiEcCE6oadYNmRP4/E7g998AvAE1yvVGLd/
b+22YWGUNcTQFkoVEEKxAPAjppb181HDkD5kZS0GRkFQMK2RZKE9vW/
NIe40ze0EMAKgLo0avvlv8n6t+WePxHAM80+UdLP2T9Y2qd5YgcIwLEV3uRyEglAtowU19
a92r9INOPk07n2v2UpEAb39gbA/
vIyg0DtCsAMDBrNh0UKwDg+beYuFBPA1w0cs0kWA8DQjBr+
+EsGwCLCw40WJgvAJVqmz2ZcGkDtyT0nCEn3v+9Dq9BSuxxAfJ08dHShDMCSNZ2HzXQIwM
CiMdBxmAjA51vBowzUDsBCQkFfn48VwIU/
V3p3oRLALbZgEjQcEMBxYMDvVsgKwBDJiZETZgrAHX8TDlsXAcCPyjH37SYQwLYGUaqRHw
7AbYzMLsz8E8D76A63Lv0Sw0ba5Sg3U/4/gJg817kq+T8jVepWQhsQwIYT1JbFt/S/
W218rYFb/
T8JCtyaUff8vxgxzTQp8hxA2NVipUoqCcCdZ0f+HxkHwPXQscl9MgnAv7VE6XWDEsBZiEC
VCwECQKi14ePG3ABAvtRcsUCT/
D+ZZD4E50Lgv0RShwBAtnQnA83h0drKLEcBLySCm6no0wCd8KDqxjgpALM5o3m01A8DAZz0
BfbgSQPf297+jigjAlFZQIBJpHUDJHwhEaebSP2w+FeEEBv+/
k+R7+w+eAMDRuhb2k0YTQIVln2VTagHafHD0P9FzA8A4odIKmPEKwNrYGnw5Tv0/69/
ob8+bCsCsaFSRWNz+vze4XNHWKhDA56TUlqi6GEARUAZUXN4lQP1w+oSu8gBAyQwwrfkYC
MBJbgBw23Hvv3Ybva1759Y/
UqhTzySNCcA4JRFNJPIHwG70s9wijQnAVCSFPyteEsCeYz4CKOMDwNI1AHV3MAPAjtDmk5
vy6L8nanNAsh4JwGwrDNpN4QVAFA0zgHnoGEBjNSfwSsoBwL0m9+mC0hVAHy1+/
fP9HEDZ4VWkMdQQCeh9VpB/wPAysxt4IUDUD/

f72jaycMQPIdogRq6wTAdKeToYqauL/vtdfz3VsAQE4P7/
mziAjAMHEqQkqo+b97r4YJoSA0Q0eDYdo9nu2/
kNRj04p0CsCn882lrl0D4v2kNFDSchdm/CN0WGPBnAsD44IGucl/
hv2pCuVV2eADAuDneuGllC8BMB6iw6DvevzBv7X5i0BJAf+zdia0lDcDW+o987MEMwBL0R
vmsmBBAhD8L4EWwHkCuH1XNQRUAwDYWi/
PdXQfABX0ARwsG4L9GUZqHF5X+v7j5mxriRw7A2kjxdMqqBcDwr3gpoCgZQK2Rd+c3cgvA
093N9NcCAMB8FsvjTckDwDDEJniVLQ3AzNKYnP3EAsDzNi8lt0nyv0H6NYwcnfw/
c700JGIbBkDf+013WYL/vyiW7DIScwfaF/Ji+TZpb8DBFkGPMfgPwMMa5F2Q28M//
Y1BaRLk7b8njREJu9/+v67aKqmM+gTA865WEbQDGEBYdGx4AWAHwGj+r45NvArAVMjlxYs
eAMAZ2gqkvD/P149c5yytvq/Kmaj2GHB178APuQBV5X9vzAah0vqzxLARGP0P++H9L/
qSx7Wk4MCwJPadznnohLAvpS5xTuMD8ATv1Cyp8wTQCHfoi5DTu2/
YItCMUoyEkAG+QWE9cQRQFfv401TjwHA6yTXlxld5T9iS6ZsU00BwBBe1ZleAAHA8N7uiC
vsBMA1x1/6D7AJwHINn2zMod0/bqgsEGPHA0DP68hiQ6MLwI+mjGtHNA/
A3b3A4E9q7z+fNsbssx30v4FAYG6b9glAb68Y517q+L8qUdk09vzzvx6QMe+cUxFABcc0m
z098b8tsa4CEhgEQNfkA0SavAHADlHvpALI8r/PE8jPvrAGwPt6GLSftv2/
vpsNJQFC9b9iJ5xjE2wGwLoESF0EZQnAXbDFzipZEMB1ZY2qz80LQC8U30S5UTBAdQwjLR
17BMCvZd/
ESMAHw0gBQbI3DwbAocFWgens9T9bww3aA7vavw06VVYKtQ7ARI0m2k65F0BPFifWwqfkP
80LPXaEeAXANbJp9lNVAMDBStKRhGP2v9wlNxqUSgzA3VKtZzXZ/79JpjI4kfj/
v10QmTM7qeC/
9ZJ0xwNhCcBZhASFJp8BwIBk6VlwmA5ApUgdzEXiB8DtprM9kDw0DoMIyRwvS/
vMs975nhAcC+nDyNCMSsv59DEBYEigJA6bbUd09GAsDpY7rCuQQTQNIpYzL4nvu/
hcXJHa2857+WFOxzJzCcHwDD1K+brePq/z+R2IwLUCEDp/M0m0EEQwK69Zb69WAbAv/
cDpkWZ97/+Rf2vzVzNv/5+d74SbQTAIaXnmM05DUCQmGU4HmEUQNWNpUg/000/
83RRJb1SBkD9PVavU4r2v9QVlc1mCR1AC3l0SijI9D+5s2q6IjrxP9qkW+QE//0/
7sC49WAc9L8dBT3UIKryv28JEuhmYBBApJCENMUz/
r9rDWxUtSMGwEIYnHzi0AFAIIyqXvAa8r/
jmaJdEHvvn6LlMZBbqwLASR85D0qTBUBLk5TX0AgLQI35kfJblcm/
HoJJjADR8r9YQ92mo670v+UnlW7yQiFARCuzac03EcBnUeANuNXjv2ehMyG+BArAq4FGa8
vSCsBrTDyNgYPrv60fygTDQAnA9PpXQIQt1j9aBV/49bAdwIrd70RKe/i/ZHf2u5pg/
L8LKs5gi3EFwNF3ESIeHRBAC9AcUD8XAcAqfNV8YakUQJuFdcJaJuE/
y0IOx7v11T8RNQBwz8k0wFZ0tBAKSfk/KOM1Rsml+r822CFx+S/wPxHIdKyb0PS/
KG0KM8j8AsCPQLBcaKf6v+chlxbF1/6/
uX7A6ejyEMDydw8R2wFwZJcklsQ7Axmumh1lpBMDJr/
gCmscSwBLiSK9GMwDAceeM50YqB8D00Kn6gQQMwM+Zkdgx0wHAsGx06s9wAMDIFCVmsAQw
wIzX+BFlQRDA3pEUJ0NCub/nRla6FW7xvxapi+JV1d6/
P1MWazJ7E8AUEH0ioqsXQB1KGqqQeyFAFdYQ69vBGUC+GyTb2Fg00Aqnz3BPGfQ/
outKAeW/JEC0SCgupuYVwA== , "dtype": "f8"} , "y": {"bdata": "4ogocWUt/z/
jIe85NyU0wC0MYHDpM/G/hqNyBBqNJEDHgZ8XTSv/
v450H1J0mQ9AxLJJHi+FBcD7EJTZ0bgCQMGmMtRuIgtAQby0UaDoHKD49Gw7MkYFwC7AkH
uYCrE/0Uw4kJmbBUDD9NjJJPzuv6cm6dK/
cRNAgPa+cwXRB0CWZ4NlPwn5v8l53anfsvY/sz362BN1EMCaSIB3bRfIv3GgtthxVOI/
WT2vW3Gb+j9j57vSkHAKQMkL7YJygxTA44dqmTwz+L8p/
iktHSYAQDLvFv18XwFAZcwowVsG/r8t0LB4FjHhP7hnpoQYYfi/CwyZNMw66L/
E3NEvbAUQQIBZjz0xmtU/kShcixUf8r/69+LRAEXoPwwWBIZErda/Z8yoZec47T/
sy+D8lZ32vz/
IAqpyzwrAs8I8Rxxc7D9gt7xuq8j+v3e61yUFFgJAab33FD4mAECeakz2i4TvP3GVsYd2j
MG/8Q8SGKZDwb9+Qc7NXb/3P+m0yN9z0QJAFDa9DkqIuD8MScoQvlXzv2LhbBUahfS/
VjTCcvkF+7/56uG+00LfV667YD1SGPK/p/

C2N2juAMDXNb2FbozDP6HpgKSPewLAGtovHJM**b**7z+kW5shLpr8v7JV+4330gNAjcnFEK8P
BkB**i**d2gGABwJQAI**P**/PrFwhAUurfxdckCkBnoid2nKb4Px**i**ykw*m*wI0c/
mUBfZDV8AkB35iMF3xDlxvqpWXKxZyJAyPL7FsUC7r9v6ZA**x**or8RwLXnpqIejR1AUHxbBc
MG7j8a3+mD0Ke+P4pPrLxbU0a/23eAVD95/7/NJxMWhnP/PzjZeq41ce0/
TUsGgpX+EkbLTkpR1DQv9ct2w6HdPs/ZmBofLodBEA+VnJWWRACwDs6Crb1x+g/E/
K5UdnM3z9RtHdYGbH6v+GYzQ9yw8Q/zwKEUj6nAcBkeSEEu6/tP4/A9L3NP/A/
LffMRx3q+b8YE*s*P11ADhv1taS7YwJgfAv7hp6JD*i*7r+RhrPrI9vhPxkkhJzx/
ADA+h81S1tYyz/ES4NPgwrzP9HQE0Fpi9c/BHZt4vZx5D+
+CB0+0EX2v2v76in00gxAQ*t*WG6kDL9r9WUfrGCJf9PyWdtaiSuPY/
aa*f*830AvEEBpxGIMP8Px3MsP1tkvC/wwj/
bI5o7z+aNQiNbVvlP+bYrAYcygRAB0MH5bs9AkA0sXzbIWkPQMG4PSUmcwxAwRbDzLMCCK
Asm03Q04PTPy782F02Vg1Apr8o0Tgc+T88BRtBKhQDQIV5IKwxdALAn5/
hH8PG1z82786gezX1v5IEzltxiAJAX3o903ZpsL9Ans76E4rnv7YnLAzeCf6/
pu0LF+kp7b+qPfYg/k0MwJ0nPqj94PQ/ulGqole2978S6+x+jcX2P0NosYIty/C/
rYhADF138r8QA+DGehf5v6BdZZUunAbAcYo3cj7X2r8NsqWUld7ev8ZrKyXHxdq/+Tvu/
D6I6z/a3urtCUX1P6qs4TXxY+i/OHnh0nZ19L+q0DNQJcDsP9n6srSHTbe/
sMZ83SSY7L9xdQ/MEZ4NQAcUvb0ZWhNA5MBgLT9U8T+K0Bsx0/Yv/
U4VZL75gHASdtRpmGT0z+wo17r95kcQEXcxLZ5JS1AGwhEjjcI0D8CL2TV+zzuPyhDzYig
vnY/B17PYFI+2L88k20+sJAIwIgRUFw4luW/
uc+dN2ap8b8Te0qmBpL8P1MIptLNSQfAisCjICckAMChP3lzo0L3P6D3TijPqRDAYjWw+y/
/pBcBcn3n1GyZ8P5K2KajzTwDAJG7g5PuP/
L+gPBZGBnz9vxtDM3+jP70/2Wl7IsD0+r92XCvI3D30P/DN6iNC6vk/p00tAHmIzL+w7/
mlmW3wPwb11pHVdhxCd39/wGb5j8EAtiT8hYLwNsE0fFL8P0/
S6WdwdpFsCImlx8zpf*i*P4YJb6eMvPm/QFzZcSdH4T8FtjSU36fvv5EFn2tCHeo//
HARIH+YDMBEfu6MyYTZv/r6jNVT78M/GQtsV00o8b8oDfkFuAUgQArv/qucYeG/
BczcAg6V3r/DD4JVczYGQA74gFSV+/g//
CPUUCUL779uDLI3VToCQMm3GVvLiwPAZ+du2Qg5AsAJByyZIXnYP50vABWF7L0/
Ao8Y+33RAMBr*s*R8ALffkv409qN5vsQhAtaPxICTk5D91r4VXUnXmv5ZVB5Q46fQ/
VT9LtzlNBsDH/
a4o2J4AQP+XtPZEyAPA2z4pVpo7CMBCbsWrz2Tzv27sJbdzLB7Aw21n0d2zBEAGnCzSz0o
EQAqqqb9fPAFAEg+M00E2AkAzgWTErGPxP2vdUlVeyQbAKv1hCkSqEsCUzWKnCHjnvyvT1
gRfD9w/Q19Ko7/5/D8GarC3l07FP62LYjd9sva/RB6LfgMB9L9/Lse2+DvhP/
fnvKFKqe0/pfcKsC7Syb8qp96C57kLQP6PXdmRBsC/aSpATf+q9b9/
yMo7g6r0v8uge0Jigw/Awg0c8smZ8j/
MZ4vFa5X1v7f+uDzNRAyNR7kACcD8C6rygJBS3yv6mpyzWPPAHA1BX4gqNw878q2152q
tv7v1u3soxsKBNAIkA3wpZY+b8TQTPrcmvwv4ZssNMZnwBafBm4HtDA8r+oLmFN3eMCQIo
Imb22Vec/2keLVwl63T+cs2viRpkAwMYq5cpyy0S/WzbsRzTn4D8D75vqJGj8v/
WwMo+2dQbA+qeYPQU62T+g4A/JZCv9v0twFN3/Jw9AkYFiWTA9CkA0Xwvk4gzxP/
t8PVRG+ArA2JHT1CwADMBvVuJImLPvv5Ax87i+
+QjAeVuFH03LA8C19qjkWgMYwMcPI6Rx/Po/j0xsheGn8b9gz/
0uZnGDP1D5Fblc0A1AgDuolqtaCcDeUrh1YkLaP6G+BjE6mgjANW5i2PJU+T8jCHhFrScJ
wHEUkSlxK+s/7Hb/2Umb1L+n0uosvKwLwHkCMSuZEgXAr0/vwrXd7L/mPMdwPXX1v/
j2qDnYqsG/Gl+0eDZu/r9tmPvADXjiPwJLWK93x3c/txEsgS0G9r/
9aoJpwh77P4y3URPhp/+/\tJlTxJfJDUD4bZ6oj6rWP/waUcgFGBFAS2gf+dvf6b/
rk3X+u032PxCBQb14MdG/NJ/lPt3X6b+i5MhhJxP3v8RQxIzxX+G/eEMlnidf6r/
SntJUp0MEwFNvNQV5I0g/9ArXIP/B978MZUErMvHiP3geBLoaPKw/
0ZXWDdmH7D880YHYdvDGvxqFvh52lt0/652FiEdl+l/12hFY2MzhP/
1CydjphQjAmeLipd9XBMCG2BcQWY2xvxo1vqsTHAnAZWbRUzz+cb932NufMc/
0vye2v97BlxFa2VcSPra2AsCnP2AmQcb8v1HGmq1NIwPAh0gkVFEJGkA5oBMLcG/

2v1wt5FE3xAtAhDLlUD1JEcCBo2TfdVvZP4CAGIaBR6K/LybNBcfC57/
oqTS2Kb7dvsWBI8IV/y/cfkDXLFe/b/RC1+waU7Wv0SwIP0a8/E/
A84lYu060b9ZIm2aAkH0P+6D7/6zp/E/gWxxNfZa6L9dlWqEDR7tv/5W4cKb0/m/z/
aFB1Wu7z+mmommtv7uv2C2ngNkLPI/
qF56M3L4FMAC9VAPNQzNP5nXPboiXARA80jC+oDn/D+wG7A0N1L1v367Wqpmr+4/
Idbznt+TC0BmVbZ2gwPiv6vKfhWdEvW/pBStEDZp3j/SdIonLmL3P3r0PRTvtfW/
fBIF+5huFECqouOsQ6ANwH8qQH74+9I/
ibPp4YYzzz8REsxzF4PpP3mvvELQBQFAJpEAfCCW/b9RpbS2Lhr6P1ckot7yieI/w/
G1zckWBsCTDjBT5QXzv/+VheAzDNC/WdxRxKf+AMBvfpdT8M73v/2weVy0jw/
Ao9z0q0M99r9C4hCZuNvWv6HjvuXG5BrA0g5YmWY+DcCbulk05NHyp73ftwH2JQDAbYPq+
5RN/79x0+lobd40wHh400K8o/G/
orroeNiewL8sbYpCELUZQGkyG5fudAHA7KKRgeW50j96uI1D8fAXQE3bV2J7awJA/
KAd45HEt8i2cyY/lrvP95i0eEpSgFA0hiZXpTT7b8AegzNtRL1v21R8GkCgtQ/
0UsF9WuQA8AwE2wKZZ8NQK2CQgMUFcm/PP7t2PQt6T8Nmnf3N9cEQFUF0cNEj+g/
pAu6DhaJ67+a8XekCIDbP/UzPaAiNfu/2DRG1sBSuz88pVn6N9jkv7LDhg0V39S//
9E6zAEezL+akJtc2UoAQOLXqR52w++/IxXSW9zF0r+w0VoVaXHVvwLgcK2h3P0/
d0l1UQTZ5T88/7/vDWX/v2lGq0sZp9c/65DaNRVC8b9skGE0NILGv0SvF3a5q9+/
yP0DMjYm2D8/lU4eJLfuP50k6K1TCNy/IquJAPrYAMDhrNRNFo/
sP90bY9pNBARABv4g7LBNT7/ojlkgGd6Cv6qd92kUwpa/xrmwcRTp6D/
Do78zRICQKn12bQdm0s/vAmBHbgVvT9ds9Y1cFQEEShf9DXqtu/
8sn0I0NoBEA5o7w74rrZP/yJ1maXC0m/
1CvaLXk9AMBB0aRZawwIQDE60NUE1QNAssibF60c6r/56jc4D8LvVz6QnR064vy/
ryn804f12z8dzTGb/hfdvywD2V3jHf0/GmKkmR5Q/b8gTIJ/
kuEBwCndGKv gegFAYLgiPUj/979edvQmeRcBwNWKn0mEYuc/
uYFn5fWuAcD8Ya6yQxXxP/YPYy4odPy/NawZdFmb8L/
exe044mv5vzR2jTKEmhDAQU0TuJ7b/D8FfGR4/zoEwHWqNGPSm54/
BHs7PMVj3r+iABZJPkrwv5UzfBxtJPm/qXH/t/
fd6D95SgbnK6n6v2b1UzoamwDA6F05bH0F37/+6mRsIsT6v2UFUyh5Gh/A6c1/
Gc7zA8DGPDVE95+xvyuAjCrks/y/F1VVoU3hAEAJeRMgPNS9PxcxGU0Sxuw/
w5ABbBqn8z/NsgWteGgLQFHRra7NGPc/HUWc+njn7L83K1DoCVv8v9MAQFc d8/0/
vgDz1zJk/j9LrRuRXmrKv7jG0kDUje+/
xJQdDWG2AMAg5fD2nZP0P3wh47wfcQJAVUoivv6L5b8mauw1Y+kCwKwNGh0PU0A/
LRz6XH7H9L9EIzMm5hrNv//2KD5jhFAK0GpARe8A8BGepvMrdT3v0Jl1KQS8e8/
C6sggsSdA8DcyYniUUvvv+fZBQQA0RLAj5P4lkjH87++jptEVvPhv12qh6AHP/G/
N+V8MR91+b8M5UPxI0H4P3CUJ27fcei/n54Y9mDw8L+smdhvVicAwKtDmt+e10q/
WNLM4qAVC0BIxIjnohf4P6Ez9ySi+xXAwbxmiF7rHkBHTsD07JYgQKIUHsvrh/k/
v5oSf93EECLmoB5rqb6wgRMR9lbAZAErXnH0zB+j+zc13z00kBwFwM7JvvEP8/
ISMBzHaT9r8XFcI3Yhn/v2FJq4nhS0k/srmfmW8Z/7/DNXHL3KoCwFlzQkTF0PE/
UwTYfpIu8D+3nma4XdsTQLiwKiD1rAvA3yJaCzUn6b9Jx/0k+VzkP3hmC0Mmk/Q/
cq8b0tiBUCaDz2VEMu4P1phM1ut4+u/A2i11duo+D/tGjaFJTrjP0YvMZgW79k/
tN3Joha86T9Qmb0eXsH3v9Bt04AMiAfAHdd0NhtM+D/sXV6RwA4DwC9k5g6jXdg/
kSN1AJ5GDECFe4Nda/ACQKNR9HDYZRZA9aZ/G/Ada0AX2rSRbXrxPyb3G1+EJP2/
pRNMfLzn+r/7QaGNIWPLv6M0LMUuNPK//osDvj2Wxz+a8jdjvoYCQEwxQ+Pwt8/
a5zxgeo5zT9uIMXnLY/ov3fMnfwLLv0/7eNcb0Y597+ivTq7NdD8P7SsmHvsj+q//
MS1ZATd+T/JacKAG1nxP5EI3DluRr8/fUavx01Vy79KvxtHga/0P68AsmdZyca/d6/
fW4EMAcBgJdSbcNwLQNmTVHkExeK/b270d1KdDMBzcFdJIawMwEIOn630b/6/
iQoQq43A+j/qgnlJ23Xlvw==", "dtype": "f8"}, "z": {"bdata": "WzylFX348b+S/
HE39+/gv0CFi5Hqp+G/4jo7xMDcCcDCQekIfDz2P4v1J6+gegfAwqrgVhU9+r8QIt/x/
ebrv/Z0SCy79QjA5NjPLR9eEcDLHYDC/Ejf v+/lGGRcbPi/

11bxn8HrFkDg1B7piHn7P43wZJVnhwbAyF2ku48ACcA33P2lPK7pv0eLDQadKQLAht9CdN
kk1L/7eIasLfziv2Pwzse50vy/oTGGUEHg3D+9GUi73b3v+bqWWmtE+i/
RDhHCj5p+7+v7ZGts7Sdv+PRVvMHKA/A6Ga3VmunBEC0e0EuzxcGwDnUQzlCtsI/
S2nZNfBZ1r92B76//gIIwAne3287I+i/CEdgl3ueBMDstARYOfMLwEgbPd/1wf6/
qRB9Nlk2+b9FcEpgiLf7P6f3R1no0BFAoVxdNqyjBcDyXV1bQ+Pyv/TTuXnCef6/
njjdPzshAUCyv1PtYXwCwFTpakhagrAAhfdBIrc+r/cNbuxzYqlPzvjzyztnwjAmwP/
Sevg979rYZCDTaDXv2Dw54yXSvA/P1BBukbD37+Igu1atYXDv+n3UeT0ceo/
Iorgh0pW9r/Anh+Jq3PbP0pFT/NmUue/IBJHpyoa/79/RSrWmn r0PxoPU1oo4/0/
YMMNDuVs8z/eSQwyjGf2P7Hfez9gR8W/zxcl0apeBkBaz32EKsz8v9gIFxzCdfe/
933XoCFB3j9BxXoui+ziP9pnkf4s4/c/
7MLeQ7NBbj+xL7/6i6vTP+eXVfN8jQ5Ajx9kgFmuAcCJlj+h4FMAw0WrwV6mr+K/
fNXrZYDK8T/uNDUu0UvpP8b1eC7s/rY/u6vSTwoMAEChsgYVXrvkvwq16gwJyts/
zgTM8tJ9/r+6iDDaS0KjP0Dh/qJtdQVALGT7yfVe7r9HR0anqc7CPzlz6f2vMvo/
I7VQHcdF+7+t1TSVVJCKv7w6sZa7obS/RACU6wbX0z8XH98HxtviP2CPjdtI5eA/
vfSF8w3KtL/ZT69iemLjv3b1lZJYi+I/kn21Yd2R8D/5LSE56tT4P5iQHPh6n/W/
CDvND/GWyb/PwV2eGi21v2oIe/fEjcu/f52jyJ22tr9109zyRdDhvyLQV3tIA+8/
uBxmfBu5BcBHGoR8DTDIV870zqLX2fe/qKE5M3kp4z+ZSML5B3fxvxyijiYyt/o/
5FrAdGLgAECwNGgQRWAAQMvaYjM0t/s/vL0Qb4IA+78tE0ToNBXhv8JW9R+/
Dg1AJ0DoV1fpBMDaRvzaMvb+vzWSe+WlKOI/
W26EiUKIAMBRAAsVP40vnP097wvKVUB1AMku/1okm8r9mFhwg1xfRP0KbUsapc+W/
MDDckEwkAcBR7tW9KxT0P5FzZqCVhuQ/1V8riBu78r/0z8T3CyLxv/d0fsgyMfS/
QPgeL1Bb87/7ATNyl3Dlv/q4MNrI/0G/h6k89Fdb2L8lgTwWPfx9P0GxX45FI+a/
sisDnDVeDUCwpwIv9inyP2QY9NhA9eq/wtkxaHjJ4T9bxvUqvZnWP+k7FdTeH/e/
oJBjEc105D/86qIderLLP2Zea8tUc/C/qDNtmBj7AUBLW/0/
ebvvw0xIUYWEC96/3+zMrUem4T9SNg6CoLmu9Ia3ytRbx1A+0dX2xvuuL+6uX2dhoHsv5
DzyARVj0m/zX842h0k9z8J0aIg2S/3P9A2D3oHBu2/4q8JuLx30L/
xxzjGMLX3P98Z06rSDghAdjUat+w0+b9S196WRCbPzTJ3TeHPe0/
UZSFjKuhxL91NKsV51bu6Pv2PLcYe8/aqz7JULH8j/zkpcmQ8DRvx4vi0CqyN+/
R15qR9Z51r8Tlf2vozkBwIyrwX5cZwVAdJN4ESLz+d/
RAkvDkczUP5Iicv1iFhFA65iv7Wet8L92sQMAUCbcPy42E1gfYuI/
dMGB40Nk5L92a92P5RMJwM+0AAHPfi/v7/cpu5P+z8kpwE2ba4EwMmTer9gQPA/
AonWQbXKAcCkGS1TijzWv5pdiqXLU+E/
jr6MwK13wz9JAzw4I0TUv3rel5ASYwpAo4UWmIWCEEA0gZb+HCuvv241Lt0M7cu/
pG0TzWB82L/zegyR1Rz1P8nIa2YEuwNAQkX7YabR879ENBpR4bQ0wAwdopA8esu/
A9LZxGw1779Ub9HS51LMP4L0o7vJKQLAKpg7j9aY5L+jXM1JhSoCwD3X4YiE952/
pRIOVcJw3D8IKMxpn6oAwIVDbEUzY/e/
395HEMDzBUDf96q8Pxjbjv2gh+pqP0SRA4K4D3EheHEAfZdK6q2rKP5ePrQoMlv/
9674thoT5D+LZe/12FD7P/o/ksh+Pu0/N4GmM1EFFL94R6TMm2r4v/qbtY/DKfe/
60NG4oaZzL8AqGU1XwcFw00VnqUUQtS/lxJZv0Qm0T9UgASkujXsv7jdEDlapfW/
NPcr3gGV8b9juu0s28oDwJX8jyQ1KwfAVXb3JfwR17+giky10CzbP94EfGN0TMq/
0MMXIahJyD/ik9MwsMjNP3UPHKRUDKg/dod81znz9T9kH7INGb3zPzPJVMU7vvG/
rvqKvA0hyL8bZ2FFMWzsv6wdcNt8M+w/
oCzXRAE59D+1RR0Fn8HsP9v6ZB74cQNAhKIwqELw0D9N7X9tw975v1YYFr5iiuG/r8ZK/
V0q47/tzzDXCGrxP+Hk54054tM/x0efsdN69L+AKP2GGcP6v/1EY3heGfy/
nY45yD4T8L9bjLGg27XeP1DrbGztruq/l9zoyMY6B0A305Qm7R8MwNKHT8C4RPy/
Ndq21Z9bxz+0/P4hYm8EQLB7jSBHAvg/YvW2JbDb/7/uKbSL+6jvP5jkyzfvt/I/
iSj674to9T+fG31fANTcv8cKf6mJhPc/Bcr78WYSyL/jd+sbt73xv1EVnxIbtvc/FD/
1oMPr5j+jLYZM7J+9v7/HueR/jwhA7Ski/sTc2D837Ssll6TTP85/T6vjerK/W99/
KZ3g679A8H0oF1D3v78J2abZt/w/uoehMmro+79c1gBdSur/v6XtEX3jyfg/

m5SIYfkY7D8dG7QoXmvwPw4SZM1nFdC/
sXJvVItYEkB7kxB9hr7zP5y0JfoRdxdae1ACNh9+8b+bUeacmMjqv/QS6SLjA96/
DNrANZjT6b9Ak7VHHq3iv+UhsxCRmuw/WOU7f1C+/z8hs0KBAkrsv/gTPzREWfw/
aIr9YzEg1z9NofMDaVfyP07lU7F0GgBAAn00xqzw80D8kCQBtkexjP7CswinnJf0/
iuxqW0lP8r+PkJONG0zhP1jBTEFvb0/KQrIBByb1j/VNNjK2ErRv2/wnbvnDsM/
6cM7Yiw7rl+NoZ7VAUD1vwh0rpEGmg9A5o90V6+Kxb+tCgk1ZLnFP3g0rYro0vq/
UxGdclWAUCFSVap830JQCnXxmigwPE/iy2/i5JE1D+L3VPVcq/zv/Fy6i0BugbAZ/
8JMGKY679KBXQvlHLLev/Nv1wjLWKK/yzyg5HBxsD8yFCw10Anxv3GKp04MZ/w/
0YLIPWyp+79tjILTQHTtv6KDx80syvk/SKrsXW0j2b+sEvj321zcP6Imtz+LxfU/
4cmDgIU96L8PuRNkNKTsv1F2H9fQ+dc/EQS+DQFJ1D/dUBYYQG7xv4n1bq7UBfc/
itM0eFus8r8sJh1aDN/rP32nAypk/M2/MN0ystKnc0De0JTmQKbIvyJ+RfGgy02/
QhcKvsG+7L8ku0Mhtc05v9EFqu4Y0uE/aulmBC+t4D/igpkBwojsvzKKbrpwus/
s9h12l69/z+91S0uk/QCQHloFgGmqeY/0CUQmityuT/z78WEKHQEQQPoSagdMwn8/
I3NC75pg7z97h7nhW1LWP4mh2mbT79S/2VeKhC6H1D9JF/U1my7zv/t+vcDglJG/X/
aHv7ffBUCRaVGmfaDxv1jbsIFIxfc/5F6yCQvuw7+C6MLotnD5v8J0qPuZZue/
yC4qb10F6r+cUt9uR1zwvxFWr/4N2v0/CT4ZJ1G++r9t+n/2zGL0P+F8qwp3NG/
mleK6y1h+b9Mg6RkfYUTwFfmgZasifm/UZ1RWJQW5L854M/iGyxZP8rZTXRlQu6/nY/
D30kU8r/JHqaWON3vPxGIWTlvWb+/p/M+NaEQpz+wDxfxs0L/P/
NbTUpnDg5API5ZLiaY8r8wKfPT6aH9Py69iupJauy/b10Q6qw6AcDQ8Y1/
kKW6vxAGxPsx7dI/qkQXaB18L/k1V00YTT3P684ZYD17fC/0jS8h9gg1L/
W1BuM53QJwJJ3VMkKaue/jluMrITQ4z8wjAs/wbf3vyV0YzWdL/Y/getQkq0f8j/vyo/
m/Nnzv7en8zS25ANA0ntZW73f8r8TwnBLQ+3aP4s7eiDsRfc/
MhZVeJ1g7L86c08b6inKP5M6MohTiN2/4X0UZzrt9j/z+Cz4Na6lP/
OueAThegNATPCeABVh+z89KrNmefsvwCxAh8LG+A/
IXN8RvcSwD88wnWk5sfzPxy0GM42KvC/5RkjK5jy77+GeUiw8woAQB7sGaLJunU/
Hp2oH993tb/xeeUqbZPRP8RANnxz18g/yZ0Q87Ii5j+hv+x4rRD8vz1A/6zyNNc/
8ttKRKSZ4L8yZZstvRP3P+Yv6B7V2Nu/mLH1xD1yBcCJEok5NMvdvx/hKzBBYt8/
ke2rQWN16D/RSVSU10bFv5J/e4pTHaQ/hnEBSy/t4j/
KH3tdoDzJPwCBfnKsFgFA+vaggd4y+7+vdK4pPpvQP+iwu0fBsPy/
1AcnX0bR9L+QYPUy9ePUv7bLBWQFpMi/W048giV//D+H+ByQ34rpP8vYTqXhXeQ/
qNA7zQ7S3r/iUNGKPMn3vy7Yx/CLcuo/Of5khK4b9D+hidKwl9XIv4lxp/qpLLM/
1sImSyI05D+geB0JIGvrP39KdMYQ8RhAatgeh83MxT+ohhhfv+jov21LKwUDQ0i/
NJ0KH61R67/LD2uEFCjhv5syrhYFa+Y/QYg2mp+L+z8hBygEEnfjP6H8QHXJuNU/
Q80yWvW+BkCuYz6+SKz3vyJS1G1SFQtAsgG963s/AMC8s4IU3kL6v+vsX1KivtS/
VcVOZmBS77+3CA4Kk2Xsvzm1zb8RNZw/73NiCkv14r8NWbwC9a/bv9zMFxh06QHAb+
+2MJLY7L/NE33KSwsCwPhyhDpfPw/JVinEy1H6r8tvYBRXQ/0vyIi0TyBirS/
nJc0s572/78Z425u1sfpv077pdMHE+g/uIrXowGZ9D9MVos42WkBQkq1Wi59Bus/
8pYFZb161D+MHaCV8roBwPdW33sdwPm/T5dm9dez4D/W93sTKLPmvxCQ1HA//0W//
10QhDMZ6L/NwEN706bWvxXoggQnk/4/FnzRv5YL5r8Q90kqz8nbPy7q37AetPW/
oNe1ICpq2r9wb95Wspznv5yr/wLlPOC/
UnFCeA6rvj8V0Cjz05qVv6p0nnqSALA0jTmSuoU4r+kP1tLeJ3Rv2HtbXYcZ9a/
xHFZp/G87r/CR6s8HTvzv3Cm6WpviuW/EoPYU87G5r/wAoAVqyPwPxpmpRD5V68a/
8wA+mH9M3r95P07AVknfvzyLMJ0J98u/kkxVAyR3/L/
pppXhKYMGwKDTIurSkAHA0tdtrYhFBkB9FCA6Ur7yvyJXCHHGv+M/
Gw6GC05AMAYzung+d37vz9N2I5Ajfo/qjypL1Xoxj9XBENKlwbbv8Ihg/xRXNy/
euHGhtDVST+4kPwc/Dv1Pys6IDEM0fQ/SPKltTYJ+D8ptNmsED/kv9cm9CFF5Nc/
eZQx6xuc2D/zBS2BCfTCvy3rJs/dmt4/Go08Bwrk1L88Wb/2/ZGrvyrNNPQt1+0/
QBN2H3mg9j8FXQR6FLT4PxNkfsRaK0A/K5WXmTv66D+hPsoEtCEGQL/JZJTQEuI/
X6yTwmBs/T9Vv4SLVML9Pw1lNqIB6QlABwJBaDjk2D9Qy/VkvN72P+V2fz2Dg/A/

```

WrUvNUJQC0AVbwSHMhoKwD0FDkv2t0w/vVMMSSgetA0CF+f0bM7UAQCZ3eHLjAeI/
eC4v6E8I/r/TvwvY2tj3Pw==", "dtype": "f8"}]}, "layout": {"coloraxis": [{"colorbar": {"title": {"text": "diagnosis"}}, "colorscale": [[0, "#0d0887"], [0.1111111111111111, "#46039f"], [0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"], [0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"], [0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"], [0.8888888888888888, "#fdca26"], [1, "#f0f921"]]}, "legend": {"tracegroupgap": 0}, "scene": {"domain": {"x": [0, 1], "y": [0, 1]}, "xaxis": {"title": {"text": "Componente Principal 1 (44.27% varianza)"}}, "yaxis": {"title": {"text": "Componente Principal 2 (18.97% varianza)"}}, "zaxis": {"title": {"text": "Componente Principal 3 (9.39% varianza)"}}, "template": {"data": {"bar": [{"error_x": {"color": "#2a3f5f"}, "error_y": {"color": "#2a3f5f"}, "marker": {"line": {"color": "#E5ECF6", "width": 0.5}, "pattern": {"fillmode": "overlay", "size": 10, "solidity": 0.2}}, "type": "bar"]}, "barpolar": [{"marker": {"line": {"color": "#E5ECF6", "width": 0.5}, "pattern": {"fillmode": "overlay", "size": 10, "solidity": 0.2}}, "type": "barpolar"]}, {"carpet": [{"aaxis": {"endlinecolor": "#2a3f5f", "gridcolor": "white", "linecolor": "white", "minorgridcolor": "white", "startlinecolor": "#2a3f5f"}, "baxis": {"endlinecolor": "#2a3f5f", "gridcolor": "white", "linecolor": "white", "minorgridcolor": "white", "startlinecolor": "#2a3f5f"}, "type": "carpet"}]}, {"choropleth": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "type": "choropleth"}]}, {"contour": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "colorscale": [[0, "#0d0887"], [0.1111111111111111, "#46039f"], [0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"], [0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"], [0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"], [0.8888888888888888, "#fdca26"], [1, "#f0f921"]]}, {"type": "contour"}]}, {"contourcarpet": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "type": "contourcarpet"}]}, {"heatmap": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "colorscale": [[0, "#0d0887"], [0.1111111111111111, "#46039f"], [0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"], [0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"], [0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"], [0.8888888888888888, "#fdca26"], [1, "#f0f921"]]}, {"type": "heatmap"}]}, {"histogram": [{"marker": {"pattern": {"fillmode": "overlay", "size": 10, "solidity": 0.2}}, "type": "histogram"}]}, {"histogram2d": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "colorscale": [[0, "#0d0887"], [0.1111111111111111, "#46039f"], [0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"], [0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"], [0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"], [0.8888888888888888, "#fdca26"], [1, "#f0f921"]]}, {"type": "histogram2d"}]}, {"histogram2dcontour": [{"colorbar": {"outlinewidth": 0, "ticks": ""}, "colorscale": [[0, "#0d0887"], [0.1111111111111111, "#46039f"], [0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"], [0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"], [0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"], [0.8888888888888888, "#fdca26"], [1, "#f0f921"]]}]}]

```

```

[[0, "#0d0887"], [0.1111111111111111, "#46039f"],  

[0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"],  

[0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"],  

[0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"],  

[0.8888888888888888, "#fdca26"],  

[1, "#f0f921]], "type": "histogram2dcontour"}], "mesh3d": [{"colorbar":  

{"outlineWidth": 0, "ticks": ""}, {"type": "mesh3d"}], "parcoords": [{"line":  

{"colorbar": {"outlineWidth": 0, "ticks": ""}}, {"type": "parcoords"}], "pie":  

[{"autoMargin": true, "type": "pie"}], "scatter": [{"fillPattern":  

{"fillMode": "overlay", "size": 10, "solidity": 0.2}, {"type": "scatter"}]], "sc  

atter3d": [{"line": {"colorbar": {"outlineWidth": 0, "ticks": ""}}}, {"marker":  

{"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scatter3d"}], "scattercarpet":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scattercarpet"}], "scattergeo":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scattergeo"}], "scattergl":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scattergl"}], "scattermap":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scattermap"}], "scattermapbox":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scattermapbox"}], "scatterpolar":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scatterpolar"}], "scatterpolargl":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scatterpolargl"}], "scatterternary":  

[{"marker": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}}, {"type": "scatterternary"}], "surface":  

[{"colorbar": {"outlineWidth": 0, "ticks": ""}}, {"colorscale":  

[[0, "#0d0887"], [0.1111111111111111, "#46039f"],  

[0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"],  

[0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"],  

[0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"],  

[0.8888888888888888, "#fdca26"]],  

[1, "#f0f921]], "type": "surface"}], "table": [{"cells": {"fill":  

{"color": "#EBF0F8"}, "line": {"color": "white"}}, {"header": {"fill":  

{"color": "#C8D4E3"}, "line":  

{"color": "white"}}, {"type": "table"}]}, {"layout": {"annotationDefaults":  

{"arrowcolor": "#2a3f5f", "arrowhead": 0, "arrowwidth": 1}, "autoTypeNumbers":  

"strict", "colorAxis": {"colorbar":  

{"outlineWidth": 0, "ticks": ""}}, "colorscale": {"diverging":  

[[0, "#8e0152"], [0.1, "#c51b7d"], [0.2, "#de77ae"], [0.3, "#f1b6da"],  

[0.4, "#fde0ef"], [0.5, "#f7f7f7"], [0.6, "#e6f5d0"], [0.7, "#b8e186"],  

[0.8, "#7fbc41"], [0.9, "#4d9221"], [1, "#276419"]]}, "sequential":  

[[0, "#0d0887"], [0.1111111111111111, "#46039f"],  

[0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"],  

[0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"],  

[0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"],  

[0.8888888888888888, "#fdca26"]]}]]]

```

```

[0.8888888888888888, "#fdca26"], [1, "#f0f921"]], "sequentialminus":  

[[0, "#0d0887"], [0.1111111111111111, "#46039f"],  

[0.2222222222222222, "#7201a8"], [0.3333333333333333, "#9c179e"],  

[0.4444444444444444, "#bd3786"], [0.5555555555555556, "#d8576b"],  

[0.6666666666666666, "#ed7953"], [0.7777777777777778, "#fb9f3a"],  

[0.8888888888888888, "#fdca26"], [1, "#f0f921"]]], "colorway":  

["#636efa", "#EF553B", "#00cc96", "#ab63fa", "#FFA15A", "#19d3f3", "#FF6692",  

 "#B6E880", "#FF97FF", "#FECB52"], "font": {"color": "#2a3f5f"}, "geo":  

{"bgcolor": "white", "lakecolor": "white", "landcolor": "#E5ECF6", "showlakes": true, "showland": true, "subunitcolor": "white"}, "hoverlabel":  

{"align": "left"}, "hovermode": "closest", "mapbox":  

{"style": "light"}, "paper_bgcolor": "white", "plot_bgcolor": "#E5ECF6", "polar": {"angularaxis":  

 {"gridcolor": "white", "linecolor": "white", "ticks": ""}, "bgcolor": "#E5ECF6", "radialaxis":  

 {"gridcolor": "white", "linecolor": "white", "ticks": ""}}, "scene":  

 {"xaxis":  

 {"backgroundcolor": "#E5ECF6", "gridcolor": "white", "gridwidth": 2, "linecolor": "white", "showbackground": true, "ticks": "", "zerolinecolor": "white"}, "yaxis":  

 {"backgroundcolor": "#E5ECF6", "gridcolor": "white", "gridwidth": 2, "linecolor": "white", "showbackground": true, "ticks": "", "zerolinecolor": "white"}, "zaxis":  

 {"backgroundcolor": "#E5ECF6", "gridcolor": "white", "gridwidth": 2, "linecolor": "white", "showbackground": true, "ticks": "", "zerolinecolor": "white"}, "shapedefaults": {"line": {"color": "#2a3f5f"}}, "ternary": {"aaxis":  

 {"gridcolor": "white", "linecolor": "white", "ticks": ""}, "baxis":  

 {"gridcolor": "white", "linecolor": "white", "ticks": ""}, "bgcolor": "#E5ECF6"}, "caxis":  

 {"gridcolor": "white", "linecolor": "white", "ticks": ""}}, "title":  

 {"x": 5.0e-2}, "xaxis":  

 {"automargin": true, "gridcolor": "white", "linecolor": "white", "ticks": ""}, "title":  

 {"standoff": 15}, "zerolinecolor": "white", "zerolinewidth": 2}, "yaxis":  

 {"automargin": true, "gridcolor": "white", "linecolor": "white", "ticks": ""}, "title":  

 {"standoff": 15}, "zerolinecolor": "white", "zerolinewidth": 2}}}, "title":  

 {"text": "Visualización PCA 3D"}]}

```

Al proyectarlo en un subespacio de 3 dimensiones se aprecia todavía más la clusterización antes mencionada.

Conclusiones

La técnica fue muy efectiva en este dataset, ya que permitió reducir las 30 dimensiones originales a solo 6 componentes principales, manteniendo el 88.76% de la varianza total de los datos originales. Demostrando así que gran parte de la información médica era redundante.

El análisis de los autovalores permitió identificar que los primeros 6 componentes capturan patrones significativos, mientras que la visualización 2D y 3D mostraba una separación clara de las clasificaciones, validando así que el PCA extrajo características esenciales para el diagnóstico.

Al trabajar con 6 componentes ortogonales en lugar de 30 correlacionados, trabajar con modelos de neurona tendrá una convergencia más rápida, un menor costo computacional y un riesgo reducido de overfitting.

<https://github.com/Carlos-HC/02-Principal-Component-Analysis>

Referencias

Wolberg, W., Mangasarian, O., Street, N., & Street, W. (1993). Breast Cancer Wisconsin (Diagnostic) [Dataset]. UCI Machine Learning Repository. <https://doi.org/10.24432/C5DW2B>.