Ingeniería en Inteligencia Artificial, Machine Learning
Semestre: 2025-1, 5BM1, Ejercicio de Laboratorio 3.
PCA y SOM
Fecha de entrega: 30 de Septiembre de 2024

Informe de Lab 3: Aplicación de PCA y SOM en Machine Learning

Machine Learning

Grupo: 5BM1

Profesor: Andrés Floriano García

Integrantes:
Juan Manuel Alvarado Sandoval
Alexander Iain Crombie Esquinca
Herrera Saavedra Jorge Luis
Quiñones Mayorga Rodrigo

Contents

1 Introducción

En este informe se presenta la aplicación de dos técnicas de reducción de dimensionalidad en el ámbito de Machine Learning: el Análisis de Componentes Principales (PCA) y los Mapas Autoorganizados (SOM). El objetivo es analizar y comparar los resultados obtenidos mediante ambos métodos en distintos conjuntos de datos, explorando sus diferencias y evaluando la utilidad de cada enfoque para la práctica de ML.

2 Conceptos Básicos

- **Machine Learning (ML)**: Es un campo de la inteligencia artificial enfocado en el desarrollo de algoritmos que permiten a las máquinas aprender a partir de datos sin ser explícitamente programadas para cada tarea. El objetivo principal es que los modelos puedan identificar patrones y realizar predicciones basadas en datos históricos.

- **Algoritmos supervisados**: Son aquellos en los que el modelo es entrenado utilizando un conjunto de datos etiquetados. Estos algoritmos buscan predecir una salida a partir de entradas basadas en ejemplos previos. Ejemplos comunes son la clasificación y la regresión.

- **Algoritmos no supervisados**: A diferencia de los supervisados, estos algoritmos no cuentan con etiquetas en los datos de entrenamiento. El objetivo es encontrar patrones o estructuras ocultas en los datos, como es el caso del agrupamiento (clustering) y los mapas autoorganizados (SOM).

- **Reducción de dimensionalidad**: Es una técnica clave que permite simplificar un conjunto de datos al reducir el número de variables, manteniendo la mayor cantidad de información relevante posible. Esto es útil para mejorar el rendimiento de los modelos, facilitar la visualización y reducir el ruido en los datos. Los métodos más comunes son el **Análisis de Componentes Principales (PCA)** y el uso de **Mapas Autoorganizados (SOM)**.

- **Visualización de datos**: Es un aspecto crucial en Machine Learning, ya que permite explorar y comprender los datos de manera intuitiva. La reducción de dimensionalidad juega un papel importante aquí, ya que permite transformar datos de alta dimensionalidad en representaciones bidimensionales o tridimensionales que se pueden visualizar fácilmente.

3 Análisis de Componentes Principales (PCA)

3.1 Definición Matemática

El Análisis de Componentes Principales (PCA) es una técnica estadística que reduce la dimensionalidad de un conjunto de datos transformándolos en un nuevo espacio donde las componentes capturan la máxima varianza posible. Dado un conjunto de datos $\mathbf{X}$, PCA proyecta los datos en un espacio de menor dimensión:

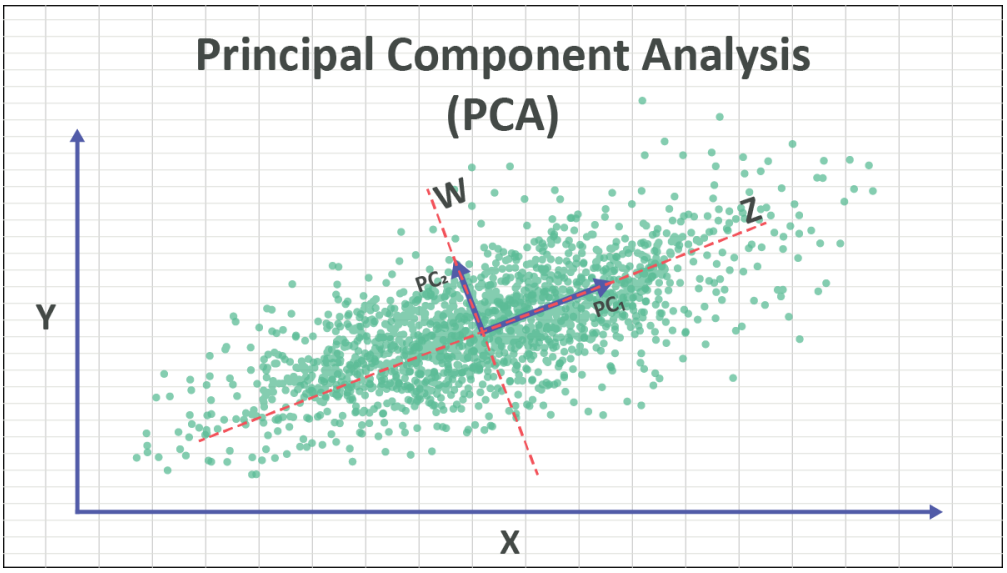
$$\mathbf{Z} = \mathbf{X}\mathbf{W} \tag{1}$$

Donde $\mathbf{W}$ es la matriz de autovectores obtenidos de la descomposición en valores propios de la matriz de covarianza de los datos.

3.2 Algoritmo de PCA

El procedimiento para realizar PCA incluye los siguientes pasos:

1. Centrar los datos restando la media de cada característica (estandarizar).
2. Calcular la matriz de covarianza de los datos centrados.
3. Obtener los eigenvalores y eigenvectores de la matriz de covarianza.
4. Seleccionar las k principales componentes (menores a las actuales).
5. Proyectar los datos originales en las componentes seleccionadas.



4 Mapas Autoorganizados (SOM)

4.1 Definición Matemática

Los Mapas Autoorganizados (Self-Organizing Maps, SOM) son una técnica no supervisada de reducción de dimensionalidad que preserva la topología de los datos en una rejilla bidimensional. El algoritmo ajusta los pesos de las neuronas en la red competitiva usando la siguiente fórmula de actualización:

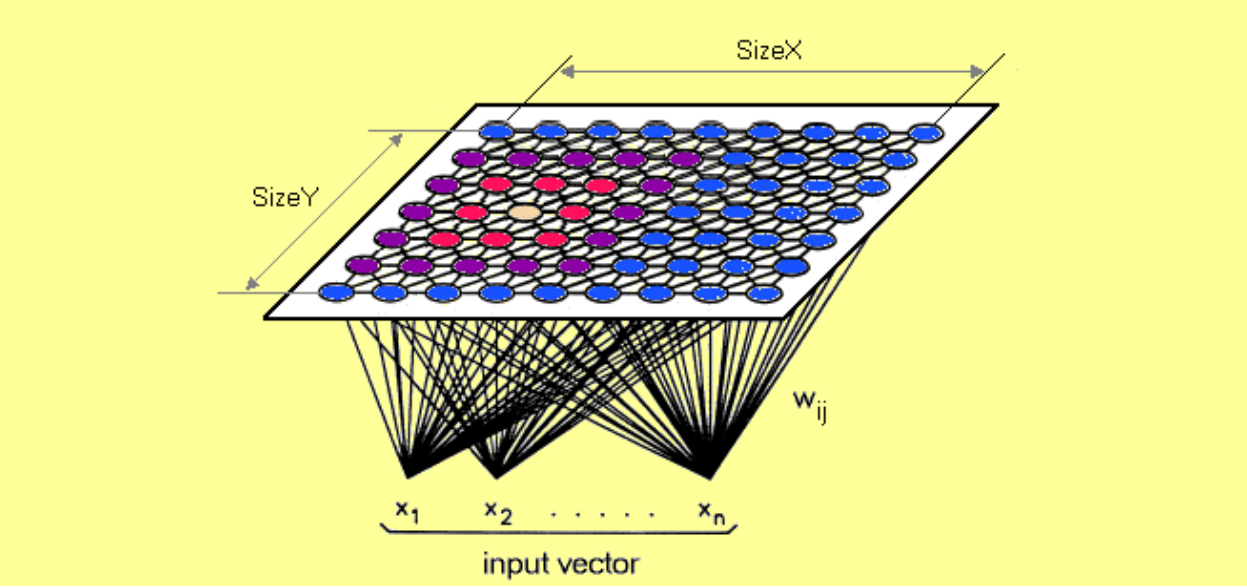
$$\mathbf{W}(t+1) = \mathbf{W}(t) + \eta(t)h_{ci}(t)(\mathbf{X} - \mathbf{W}(t)) \tag{2}$$

Donde $\eta(t)$ es la tasa de aprendizaje y $h_{ci}(t)$ es la función de vecindad que ajusta los pesos de las neuronas cercanas al nodo ganador.

4.2 Algoritmo de SOM

El algoritmo SOM sigue estos pasos:

1. Inicializar los pesos de la red con valores aleatorios.
2. Para cada entrada, encontrar la neurona ganadora, aquella con el peso más cercano al vector de entrada.
3. Actualizar los pesos de la neurona ganadora y las neuronas vecinas.
4. Repetir hasta que los pesos converjan.



5 Comparación entre PCA y SOM

5.1 Resultados y Diferencias Clave

PCA y SOM ofrecen enfoques distintos para la reducción de dimensionalidad, con ventajas y desventajas particulares en cada caso:

- **PCA (Análisis de Componentes Principales)**: Busca transformar los datos a un espacio de menor dimensionalidad seleccionando las direcciones de máxima varianza. Este proceso es lineal y eficiente en términos de tiempo, lo que lo hace adecuado para conjuntos de datos grandes. Sin embargo, una limitación clave de PCA es que no captura relaciones no lineales entre las características. Además, en el peor de los casos, podría no representar adecuadamente la mayoría de la estructura del conjunto de datos, provocando traslapes o una representación deficiente de los grupos.
- **SOM (Mapas Autoorganizados)**: Aporta más información visual que PCA al preservar la topología de los datos, lo que le permite capturar relaciones no lineales y organizar los datos de forma más intuitiva en una rejilla bidimensional. SOM es particularmente útil en el agrupamiento de datos complejos, pero puede ser menos eficiente en términos computacionales en comparación con PCA. La interpretación de los resultados de SOM también puede ser menos directa debido a su naturaleza no lineal.

5.2 Aplicaciones Relevantes

- **PCA**: Se utiliza principalmente como un paso previo en otras técnicas de Machine Learning para reducir la dimensionalidad, lo que mejora la eficiencia y elimina redundancias en los datos. Es especialmente útil en problemas donde se espera que las principales variaciones en los datos sean lineales.
- **SOM**: Se aplica en tareas de agrupamiento, visualización y análisis exploratorio de datos de alta dimensionalidad, especialmente cuando se busca preservar relaciones no lineales. SOM es muy valioso en aplicaciones donde la estructura topológica de los datos tiene un significado relevante.

Mientras que PCA es más rápido y eficiente, SOM proporciona un nivel adicional de información al visualizar y organizar datos de manera que revelen patrones más complejos, lo que puede ser determinante en tareas de exploración y descubrimiento de datos.

6 Evidencia de la Práctica

6.1 Resultados de PCA en Wine Dataset

```
In [47]: #Cargamos el dataset
wine_data= load_wine()
wine_df= pd.DataFrame(data=wine_data.data, columns= wine_data.feature_names)
print("-----------------------------------------------------------------------------------------------------------------")
print("Resumen Estadistico del dataframe")
print(wine_df.describe())
print("-----------------------------------------------------------------------------------------------------------------\n\n

#Estandarización de Datos
scaler= StandardScaler()
wine_scaled= scaler.fit_transform(wine_df)
print("Dataset estandarizado \n\n", wine_scaled) #Ahora es un arreglo de numpy
```

```
-----------------------------------------------------------------------------------------------------------------
Resumen Estadistico del dataframe
          alcohol  malic_acid         ash  alcalinity_of_ash   magnesium  \
count  178.000000  178.000000  178.000000         178.000000  178.000000
mean    13.000618    2.336348    2.366517          19.494944   99.741573
std      0.811827    1.117146    0.274344           3.339564   14.282484
min     11.030000    0.740000    1.360000          10.600000   70.000000
25%     12.362500    1.602500    2.210000          17.200000   88.000000
50%     13.050000    1.865000    2.360000          19.500000   98.000000
75%     13.677500    3.082500    2.557500          21.500000  107.000000
max     14.830000    5.800000    3.230000          30.000000  162.000000

       total_phenols  flavanoids  nonflavanoid_phenols  proanthocyanins  \
count     178.000000  178.000000            178.000000       178.000000
mean        2.295112    2.029270              0.361854         1.590899
std         0.625851    0.998859              0.124453         0.572359
min         0.980000    0.340000              0.130000         0.410000
25%         1.742500    1.205000              0.270000         1.250000
50%         2.355000    2.135000              0.340000         1.555000
75%         2.800000    2.875000              0.437500         1.950000
max         3.880000    5.080000              0.660000         3.580000

       color_intensity         hue  od280/od315_of_diluted_wines      proline
count       178.000000  178.000000                    178.000000   178.000000
mean          5.058090    0.957449                      2.611685   746.893258
std           2.318286    0.228572                      0.709990   314.907474
min           1.280000    0.480000                      1.270000   278.000000
25%           3.220000    0.782500                      1.937500   500.500000
50%           4.690000    0.965000                      2.780000   673.500000
75%           6.200000    1.120000                      3.170000   985.000000
max          13.000000    1.710000                      4.000000  1680.000000
-----------------------------------------------------------------------------------------------------------------
```

Figure 1: Carga y análisis del Dataset wine de sklearn

```
Dataset estandarizado

[[ 1.51861254 -0.5622498   0.23205254 ...  0.36217728  1.84791957
   1.01300893]
 [ 0.24628963 -0.49941338 -0.82799632 ...  0.40605066  1.1134493
   0.96524152]
 [ 0.19687903  0.02123125  1.10933436 ...  0.31830389  0.78858745
   1.39514818]
 ...
 [ 0.33275817  1.74474449 -0.38935541 ... -1.61212515 -1.48544548
   0.28057537]
 [ 0.20923168  0.22769377  0.01273209 ... -1.56825176 -1.40069891
   0.29649784]
 [ 1.39508604  1.58316512  1.36520822 ... -1.52437837 -1.42894777
  -0.59516041]]
```

Figure 2: Estandarización de los valores del dataset

6.2 Resultados de SOM en Digits Dataset

	PC1	PC2	PC3	target
0	3.316751	1.443463	-0.165739	0
1	2.209465	-0.333393	-2.026457	0
2	2.516740	1.031151	0.982819	0
3	3.757066	2.756372	-0.176192	0
4	1.008908	0.869831	2.026688	0

Figure 3: Resultados del Dataset transformado con PCA

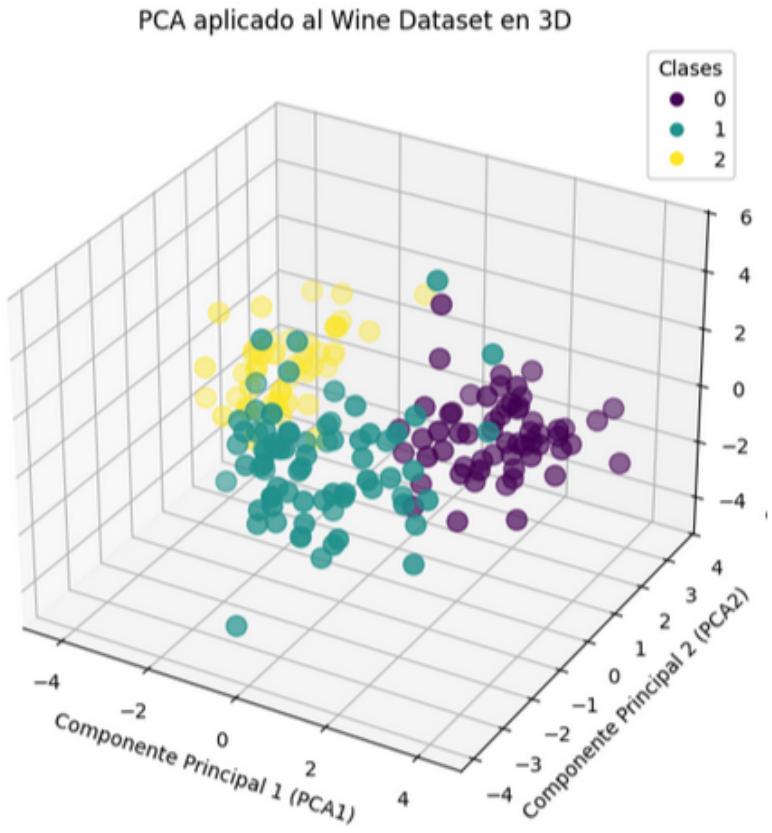


Figure 4: Gráfico de los componentes principales del Dataset

```
Varianza explicada por los componentes principales: [0.36198848 0.1920749  0.11123631]
Varianza Acumulada explicada por los componentes principales: 0.6652996889318523
```

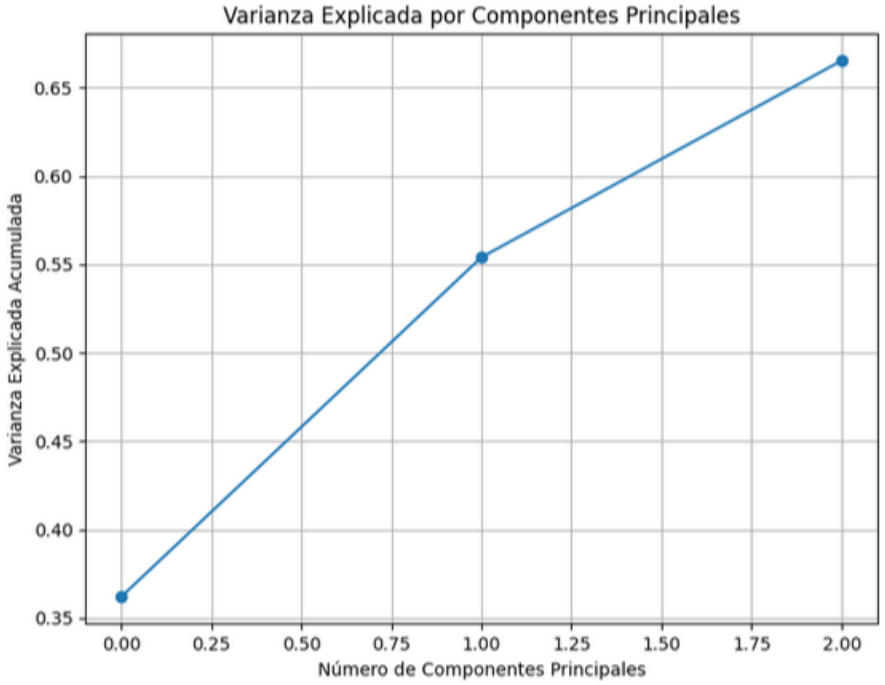


Figure 5: Gráfico de Varianza por componentes principales

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	\
PC1	0.144329	-0.245188	-0.002051	-0.239320	0.141992	
PC2	0.483652	0.224931	0.316069	-0.010591	0.299634	
PC3	-0.207383	0.089013	0.626224	0.612080	0.130757	

	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	\
PC1	0.394661	0.422934	-0.298533	0.313429	
PC2	0.065040	-0.003360	0.028779	0.039302	
PC3	0.146179	0.150682	0.170368	0.149454	

	color_intensity	hue	od280/od315_of_diluted_wines	proline
PC1	-0.088617	0.296715	0.376167	0.286752
PC2	0.529996	-0.279235	-0.164496	0.364903
PC3	-0.137306	0.085222	0.166005	-0.126746

Figure 6: Contribución de cada clase a cada PC

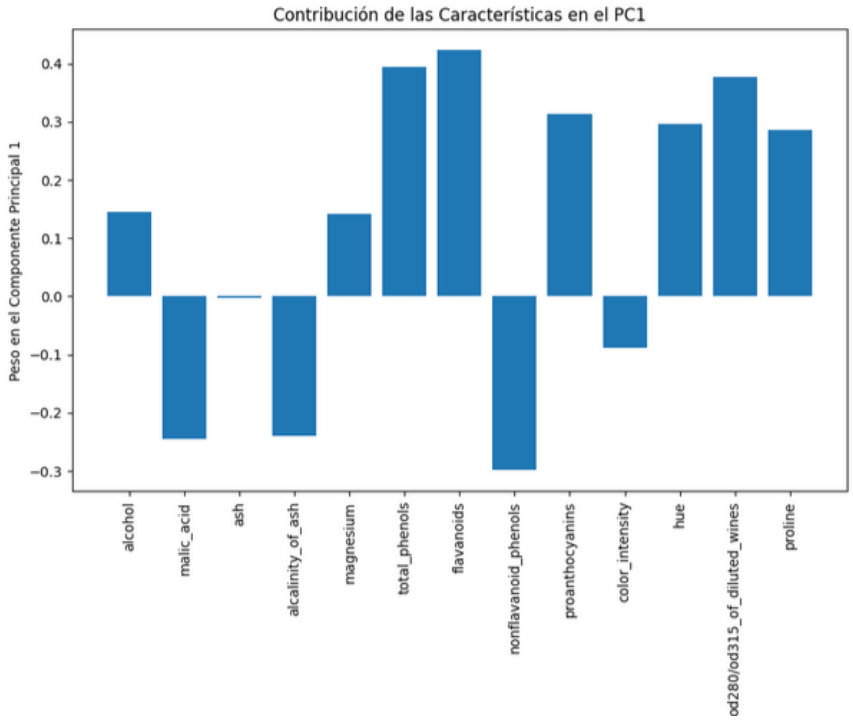


Figure 7: Contribución de características por clase al PC1

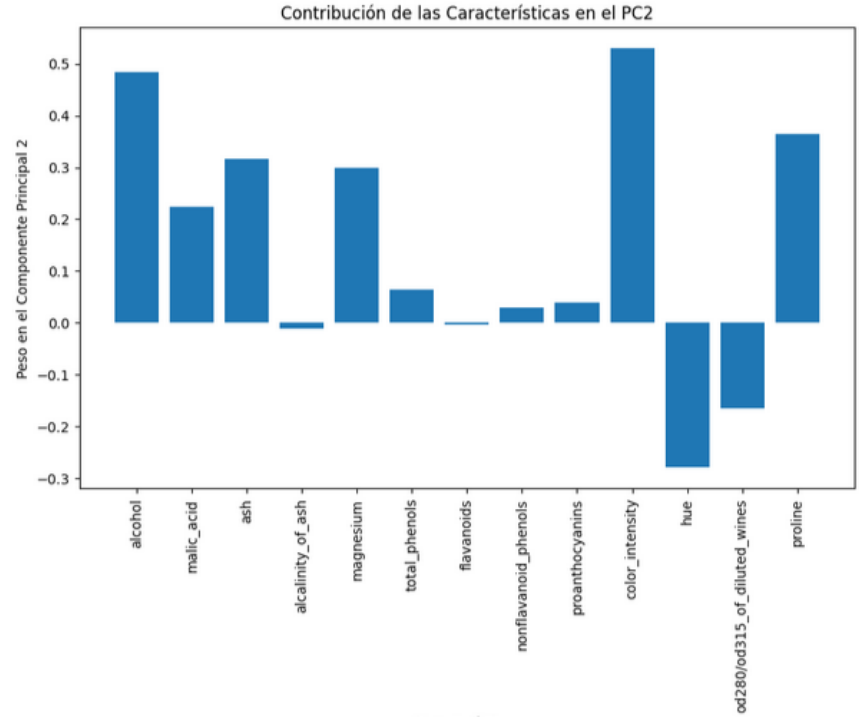


Figure 8: Contribución de características por clase al PC2

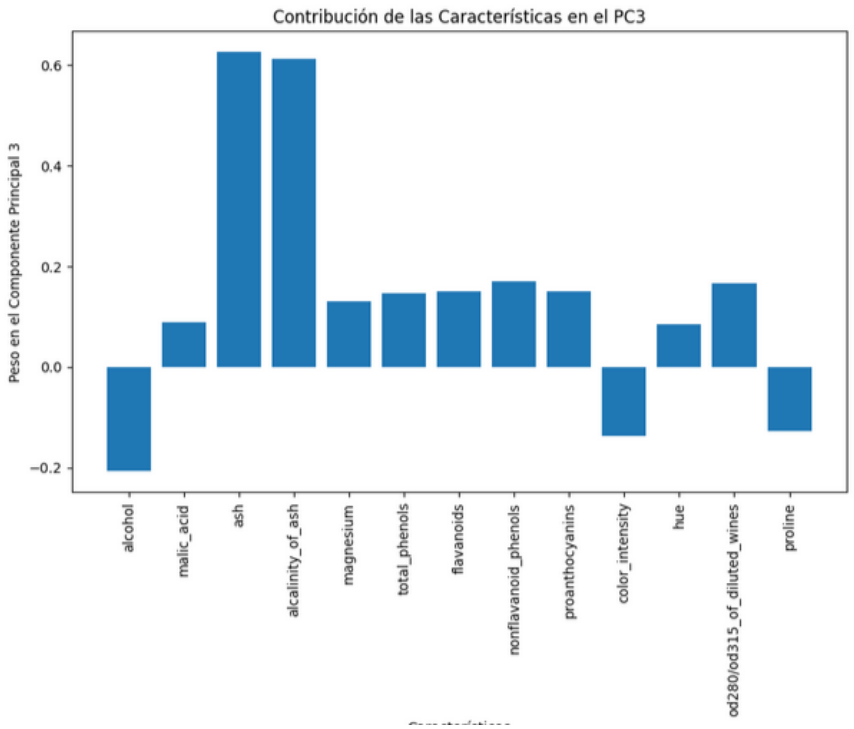


Figure 9: Contribución de características por clase al PC3

```python
# Cargar el dataset de dígitos
digits = load_digits()
X = digits.data
y = digits.target

# Número de dígitos
n_digits = 10

plt.figure(figsize=(10, 2))

# Mostrar imágenes de los dígitos
for i in range(n_digits):
    plt.subplot(1, n_digits, i + 1)
    plt.imshow(digits.images[i], cmap='gray')  # Mostrar la imagen en escala de grises
    plt.title(f'Dígito: {digits.target[i]}')  # Título con el dígito correspondiente
    plt.axis('off')

plt.show()
```

Figure 10: Carga de Digits Dataset

Figure 11: Impresión de Digits Dataset

```python
# Normalizar los datos
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)

# Inicializar SOM
som = MiniSom(x=10, y=10, input_len=X_scaled.shape[1], sigma=1.0, learning_rate=0.5)
som.random_weights_init(X_scaled)

# Entrenar SOM
som.train_random(X_scaled, 100)

# Visualizar resultados del SOM
plt.figure(figsize=(8, 8))
for i, x in enumerate(X_scaled):
    w = som.winner(x)  # Obtener la neurona ganadora
    plt.text(w[0] + 0.5, w[1] + 0.5, str(digits.target[i]),
             color=plt.cm.rainbow(digits.target[i] / 10.),
             fontdict={'weight': 'bold', 'size': 11})

plt.xlim([0, 10])
plt.ylim([0, 10])
plt.title("SOM Mapa Autoorganizado")
plt.show()
```

Figure 12: Se estandariza el Dataset y se entrena con SOM

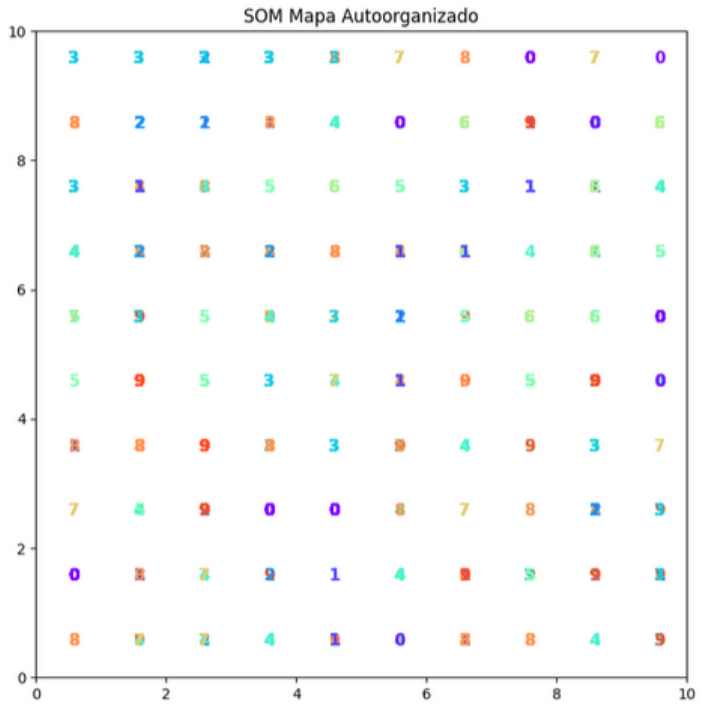
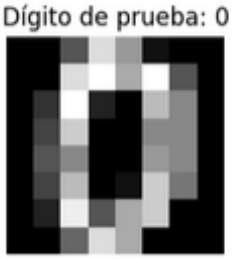


Figure 13: Mapa autoorganizado de SOM 10x10

In [16]:
```python
# Comprobamos la clasificación

# Por ejemplo, usamos el 0
test_digit = digits.data[0].reshape(1, -1)

# Normalizar el dígito de prueba
test_digit_scaled = scaler.transform(test_digit)

# Obtener la neurona ganadora para el dígito de prueba
winner_neuron = som.winner(test_digit_scaled[0])
print(f"La neurona ganadora para el dígito de prueba es: {winner_neuron}")

# Visualizar el dígito de objetivo
plt.figure(figsize=(2, 2))
plt.imshow(test_digit.reshape(8, 8), cmap='gray')
plt.title(f'Dígito de prueba: {digits.target[0]}')
plt.axis('off')  # No mostrar los ejes
plt.show()

print("\n\nLa neurona corresponde al dígito\n")
```

```
La neurona ganadora para el dígito de prueba es: (3, 2)
```




```
La neurona corresponde al dígito
```

Figure 14: Testing del modelo de clasificación y reducción de dimensionalidad con un dato de prueba

7 Conclusión

La práctica realizada resultó fructífera para comprender y aplicar dos técnicas clave de reducción de dimensionalidad: PCA y SOM. Ambas son fundamentales en el análisis de datos de alta dimensionalidad, y a través de su implementación en diferentes conjuntos de datos, pudimos observar sus fortalezas y limitaciones.

Al aplicar PCA al *wine dataset*, observamos cómo esta técnica permite identificar las principales componentes que explican la mayor parte de la variabilidad en los datos. Esto nos ayudó a reducir las dimensiones del conjunto de datos de manera eficiente, permitiendo una mejor interpretación y simplificación sin perder información crítica.

Por otro lado, al utilizar SOM en el *digits dataset*, pudimos apreciar la capacidad de esta técnica para preservar las relaciones topológicas entre los datos. SOM fue capaz de organizar visualmente los dígitos de una forma que refleja las similitudes y diferencias entre ellos, lo que resultó en una representación más intuitiva de los patrones en los datos. Las propiedades topológicas de SOM resultaron especialmente interesantes.

8 Referencias

- Scikit-learn. (n.d.). *Wine recognition dataset*. Scikit-learn. Recuperado el 29 de septiembre de 2024, de https://scikit-learn.org/dev/modules/generated/sklearn.datasets.load_wine.html
- Scikit-learn. (n.d.). *Digits classification example using SVM*. Scikit-learn. Recuperado el 29 de septiembre de 2024, de https://scikit-learn.org/dev/auto_examples/classification/plot_digits_classification.html

Alpaydin, E., Kaynak, C. (1998). Optical recognition of handwritten digits [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C50P49

Aeberhard, S., Forina, M. (1992). Wine [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5PC7J