Trabajo integrador - Parte 3

Aprendizaje No Supervisado

Nombre:

Ejercio 8

Para este ejercicio vamos a utilizar el dataset de *digits* MNIST:

- MNIST (Ejercicio 4)

1. Aplicar PCA (validar que se cumplan las condiciones), ¿Cuántas componentes necesitamos para explicar el 80% de la varianza?
2. Gráficar la variación acumulada para cada caso.
3. Utilizando KMeans. Agrupar el dataset transformado (ejercicio de PCA) y agrupar en clusters de $k=10$ y $k=2$.
4. Graficar los resultados con los distintos k's usando las primeras dos componentes principales como ejes x,y.
5. Explique. ¿Cuál fue la ganancia de usar PCA en conjunto con k-means?

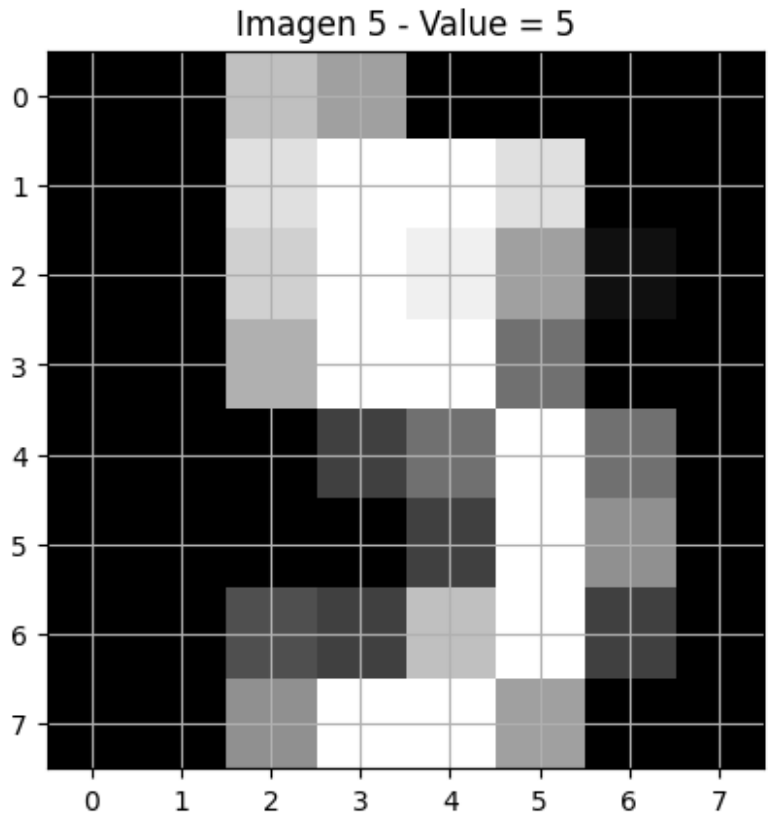
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.datasets import load_digits
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans

# Entrenamiento de modelos de prueba
from sklearn.linear_model import LinearRegression

# Evaluación de modelos de prueba
from sklearn.metrics import mean_squared_error

# Crear datasets
from sklearn.datasets import make_regression

# Cargamos los datos de training
X, y = load_digits(return_X_y=True)
print('X_shape: {}, y_shape: {}'.format(X.shape,y.shape))
```

```
X_shape: (1797, 64), y_shape: (1797,)
```

```
# Cargamos una muestra
plt.grid(True)
```

```
plt.title("Imagen 5 - Value = {}".format(y[5]))
plt.imshow(X[5, :].reshape((8, 8)), cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f11003a36d0>
```

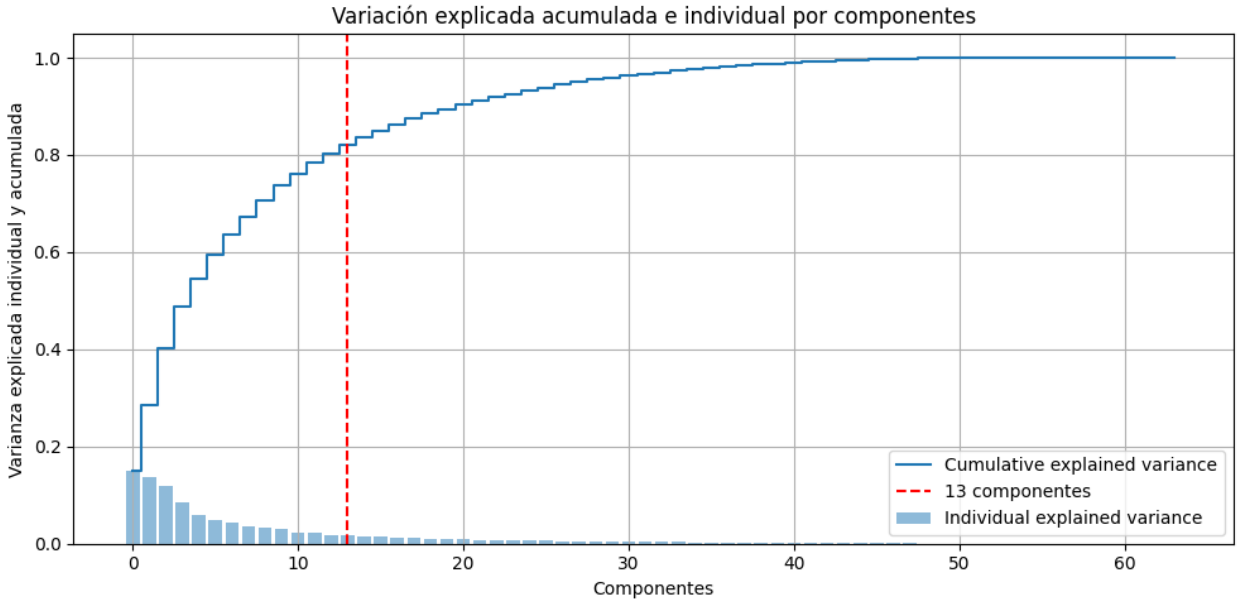



```
# Transformacion PCA
pca = PCA()
X_pca = pca.fit_transform(X)

exp_var_pca = pca.explained_variance_ratio_                    # Varianca
explicada
variance_cumsum = np.cumsum(exp_var_pca)                       #
Variación explicada acumulada
qty_components_80 = np.argmax(variance_cumsum >= 0.8) + 1   # Cantidad
de componentes que explican el 80% de la varianza

print(f"El numero de componentes que explican el 80% de la varianza es
{qty_components_80}")
```

```
El numero de componentes que explican el 80% de la varianza es 13
```

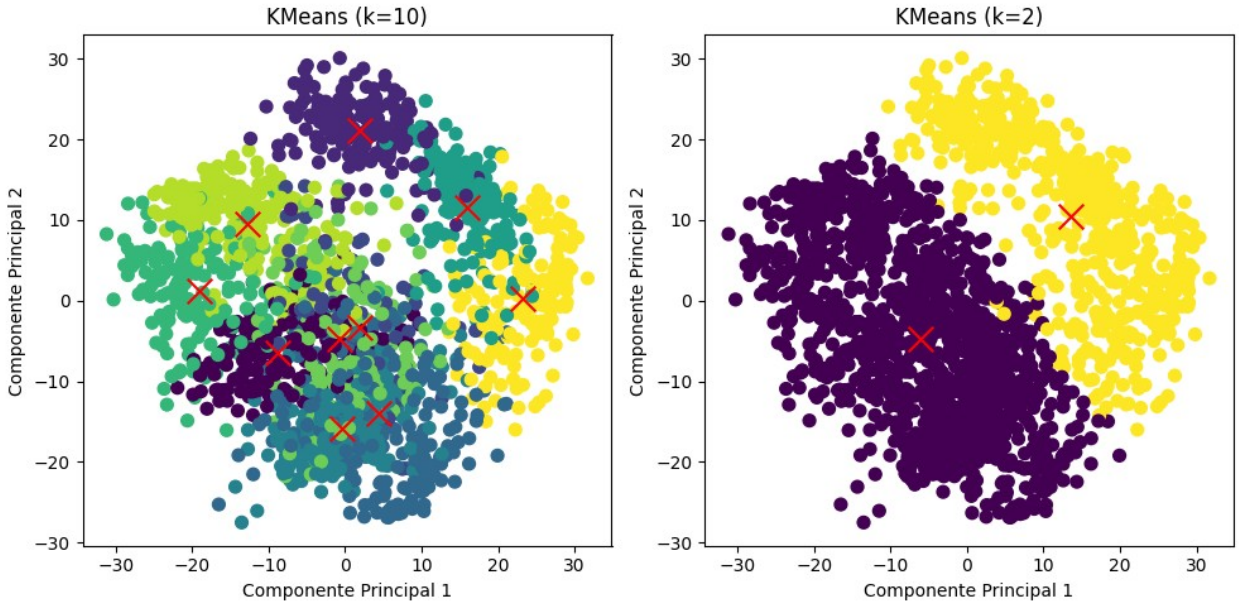
```
plt.figure(figsize=(10, 5))
plt.title('Variación explicada acumulada e individual por
componentes')
```

```
plt.bar(range(0,len(exp_var_pca)), exp_var_pca, alpha=0.5,
align='center', label='Individual explained variance')
plt.step(range(0,len(variance_cumsum)), variance_cumsum,
where='mid',label='Cumulative explained variance')
# plt.plot(range(0, len(variance_cumsum)), variance_cumsum,
marker='o', linestyle='--', color='b')
plt.ylabel('Varianza explicada individual y acumulada')
plt.xlabel('Componentes')
plt.grid(True)
plt.axvline(x=qty_components_80, color='red', linestyle='--',
label=f'{qty_components_80} componentes')
plt.legend()
plt.tight_layout()
plt.show()
```




```
kmeans_models_list=[]
kmeans_pred_list=[]
n_clusters_list=[10,2]
for n_clusters in n_clusters_list:
    kmeans_model = KMeans(n_clusters=n_clusters, n_init=10,
random_state=42)
    pred_kmeans = kmeans_model.fit_predict(X_pca)
    kmeans_models_list.append(kmeans_model)
    kmeans_pred_list.append(pred_kmeans)

plt.figure(figsize=(10, 5))
for i, (kmeans_model, kmeans_pred) in
enumerate(zip(kmeans_models_list, kmeans_pred_list), start=1):
    plt.subplot(1, 2, i)
    plt.scatter(X_pca[:, 0], X_pca[:, 1], c=kmeans_pred,
```

```
cmap='viridis', s=50)
    plt.scatter(kmeans_model.cluster_centers_[:, 0],
kmeans_model.cluster_centers_[:, 1], c='red', marker='x', s=200)
    plt.title('KMeans (k={})'.format(kmeans_model.n_clusters))
    plt.xlabel('Componente Principal 1')
    plt.ylabel('Componente Principal 2')
plt.tight_layout()
plt.show()
```



La reduccion de dimensionalidad realizada por PCA antes de aplicar el algoritmo de K-means puede llevar a agrupaciones mas precisa porque PCA se encarga de identificar las direcciones principales de variación en los datos. Esto significa que permite retener las características mas importantes y descarta las que no influyen tanto en la variación de la variable target.