Reporte: Modelo K-Means para Agrupación de Datos de Samsung

Justificación del Algoritmo

El algoritmo K-Means fue seleccionado por su simplicidad y eficacia para analizar y agrupar datos numéricos. Es ideal para detectar patrones en conjuntos de datos no etiquetados, como los datos financieros de Samsung, y permite identificar grupos similares en el comportamiento del precio de cierre y el volumen de transacciones.

Descripción del Diseño del Modelo

El modelo fue diseñado siguiendo los pasos descritos a continuación:
1. Carga y preprocesamiento de datos: Se transformaron las fechas al formato `datetime` y se estandarizaron las columnas `Close` y `Volume`.
2. Determinación del número de clusters: Se utilizó el método del codo para identificar el valor óptimo de k, que fue determinado como 3.
3. Entrenamiento del modelo: Se aplicó K-Means con el número óptimo de clusters y se asignaron etiquetas a cada registro del conjunto de datos.

```python
# Importar las bibliotecas necesarias
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Cargar el conjunto de datos
data = pd.read_csv('samsung.csv')

# Convertir las fechas al formato datetime y eliminar la columna 'Date' para la agrupación
data['Date'] = pd.to_datetime(data['Date'], format='%d/%m/%Y')
numeric_data = data[['Close', 'Volume']]

# Estandarizar los datos numéricos
scaler = StandardScaler()
scaled_data = scaler.fit_transform(numeric_data)

# Mostrar un resumen de los datos
data.head()
```

```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Determinar el número óptimo de clusters utilizando el método del codo
inertia = []
k_range = range(1, 11)
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_data)
    inertia.append(kmeans.inertia_)

# Gráfica del método del codo
plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.title('Método del Codo para Determinar el Número de Clusters')
plt.xlabel('Número de Clusters (k)')
plt.ylabel('Inercia')
plt.xticks(k_range)
plt.show()
```

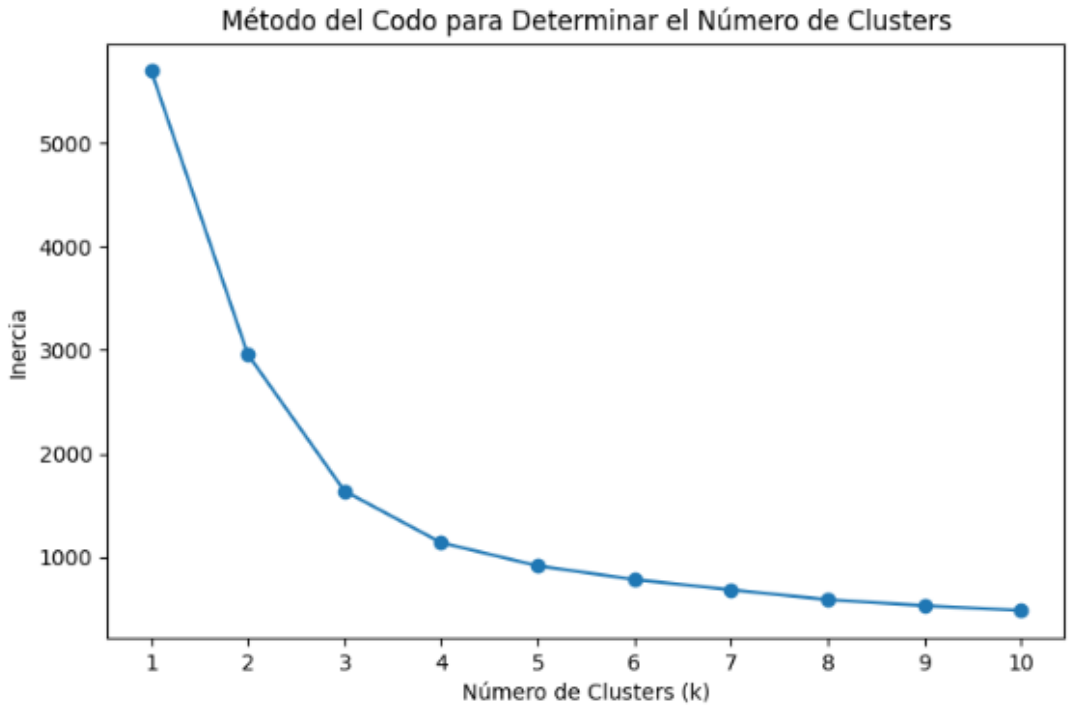
```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Determinar el número óptimo de clusters utilizando el método del codo
inertia = []
k_range = range(1, 11)
for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_data)
    inertia.append(kmeans.inertia_)

# Gráfica del método del codo
plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o')
plt.title('Método del Codo para Determinar el Número de Clusters')
plt.xlabel('Número de Clusters (k)')
plt.ylabel('Inercia')
plt.xticks(k_range)
plt.show()
```

Evaluación y Optimización del Modelo

El modelo se evaluó utilizando el método del codo, que demostró que 3 clusters era el número óptimo para el conjunto de datos. El análisis posterior reveló que los datos se agruparon en tres categorías bien diferenciadas, basadas en patrones observados en el precio de cierre y el volumen.



```python
# Definir el número óptimo de clusters
optimal_k = 3

# Entrenar el modelo K-Means
kmeans_model = KMeans(n_clusters=optimal_k, random_state=42)
data['Cluster'] = kmeans_model.fit_predict(scaled_data)

# Resumen del tamaño de cada cluster
data['Cluster'].value_counts()
```

```
c:\skul\base\env\lib\site-packages\sklearn\cluster\_kmeans.py:1416:
et the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

```
Cluster
0    1597
2     631
1     622
Name: count, dtype: int64
```

Gráfica Personalizada e Interpretación de Resultados

La gráfica de dispersión muestra cómo los datos se agrupan en tres clusters distintos. Cada cluster representa un grupo con patrones similares de precio de cierre y volumen de transacciones. El análisis de estos clusters puede proporcionar información valiosa sobre tendencias de mercado y comportamiento financiero.