Cuestionario Final

Santiago Neiszer, Marco Osornio

27 de mayo de 2024

1. Ejercicio 1

El análisis por correspondencias (AC) es una técnica estadística utilizada para analizar tablas de contingencia. El objetivo es representar las filas y las columnas de la tabla en un espacio de menor dimensión, preservando la mayor cantidad posible de la inercia (varianza) original. Esto permite visualizar las relaciones y patrones entre las categorías de las filas y columnas.

Característica	**Análisis por Correspondencias (AC)**
Datos de entrada	Tablas de contingencia
Objetivo	Visualizar relaciones entre categorías
Tipo de datos	Categóricos
Transformación de datos	Normalización de perfiles de filas y columnas
Método matemático	Descomposición en valores singulares (SVD) de una matriz de frecuencias relativas
Resultado	Representación gráfica de filas y columnas en un espacio de menor dimensión
Aplicaciones típicas	Análisis de encuestas, estudios de mercado, análisis de textos
Interpretación	Relación entre categorías de filas y columnas

2. Ejercicio 2

Paso 1: Discretización de las Variables Continuas

Para aplicar el análisis de correspondencias múltiples (ACM), primero discretizamos las variables continuas. En el conjunto de datos Iris, tenemos las siguientes variables continuas: longitud del sépalo, ancho del sépalo, longitud del pétalo, y ancho del pétalo. Discretizamos cada una de estas variables en categorías, por ejemplo, utilizando percentiles.

Paso 2: Construcción de la Tabla de Contingencia

Construimos una tabla de contingencia que resume las frecuencias de cada combinación de las categorías discretizadas y la variable categórica original (especie de la flor).

Paso 3: Aplicación del Análisis de Correspondencias Múltiple

Aplicamos el ACM a la tabla de contingencia para reducir la dimensionalidad y visualizar las relaciones entre las categorías.

Conclusiones del ACM sobre los Datos de Iris

Al realizar el ACM sobre los datos de Iris, podemos esperar obtener las siguientes conclusiones:

- **Relaciones entre las Especies y las Variables Discretizadas:** Podemos identificar patrones específicos que distinguen las especies de Iris en términos de las categorías de las variables discretizadas. Por ejemplo, una especie particular puede estar asociada con ciertas longitudes y anchos de sépalos y pétalos.
- **Dimensiones Principales:** Las primeras dos dimensiones obtenidas del ACM explicarán la mayor parte de la inercia (varianza) de los datos, permitiendo una visualización bidimensional que captura las relaciones más importantes entre las categorías.
- **Asociación entre Categorías:** Las representaciones gráficas resultantes (biplots) mostrarán cómo las diferentes categorías de las variables continuas discretizadas se asocian con las diferentes especies de Iris, proporcionando una visión clara de las características distintivas de cada especie.

Resultados

En general, el ACM nos permitirá visualizar y entender mejor las relaciones complejas entre las múltiples variables del conjunto de datos Iris, facilitando la identificación de patrones y asociaciones que no son evidentes en el análisis unidimensional o bidimensional simple.

3. Ejercicio 3

Concepto del Análisis Factorial

El análisis factorial es una técnica estadística utilizada para identificar la estructura subyacente en un conjunto de datos multivariado. Este método se emplea para descubrir las relaciones entre variables observadas y un menor número de variables latentes llamadas *factores*. La idea principal es que estas variables observadas están correlacionadas porque comparten uno o más factores comunes.

Modelo Matemático

En notación matricial, el modelo del análisis factorial se expresa como:

$$\Sigma = \mathbf{L}\mathbf{L}' + \mathbf{\Psi}$$

donde:

- Σ es la matriz de varianza-covarianza de las variables observadas.
- $\mathbf{L}$ es la matriz de cargas factoriales.
- $\mathbf{\Psi}$ es una matriz diagonal que contiene las varianzas específicas (errores).

Estimación de Parámetros

Los métodos comunes para estimar los parámetros del modelo incluyen el método de componentes principales y la estimación de máxima verosimilitud. El objetivo es encontrar las cargas factoriales $\mathbf{L}$ y las varianzas específicas $\mathbf{\Psi}$ que mejor ajusten el modelo a los datos observados.

Rotación de Factores

Una vez obtenidos los factores, se suele aplicar una rotación para facilitar la interpretación. La rotación Varimax, por ejemplo, redistribuye la varianza explicada entre los factores para obtener una estructura más clara y sencilla de interpretar. Aunque la cantidad total de varianza explicada por los factores permanece constante, la rotación cambia la distribución de esta varianza entre los factores.

Diferencias con el Análisis de Componentes Principales (PCA)

Característica	**Análisis Factorial (FA)**	**Anális**
Objetivo Principal	Identificar factores latentes que explican las correlaciones	Reducir la di
Modelo	Basado en la varianza compartida ($\Sigma = \mathbf{LL}' + \mathbf{\Psi}$)	
Variables	Factores latentes y errores específicos	Com
Estimación de Parámetros	Métodos como máxima verosimilitud o componentes principales	Descomposició
Rotación de Factores	Común para simplificar la interpretación	No se
Interpretación	Más adecuada para teorías subyacentes	M

Cuadro 1: Comparación entre Análisis Factorial (FA) y Análisis de Componentes Principales (PCA)

El análisis factorial proporciona una forma de reducir la dimensionalidad similar al PCA, pero se enfoca en explicar las correlaciones entre las variables observadas mediante factores latentes, proporcionando una estructura más interpretable para la comprensión de los datos subyacentes.

4. Ejercicio 4

5. Ejercicio 5

Calculo de la matriz de disimilaridades

La matriz de disimilaridades se calcula usualmente utilizando la distancia euclidiana entre cada par de puntos. Vamos a calcular la distancia euclidiana entre todos los pares de puntos dados: (0,0),(0,1),(-1,2),(2,0),(3,0) y (4,-1).

La formula de la distancia euclidiana entre dos puntos (x_1,y_1) y (x_2,y_2) es:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

Puntos:

- (0,0)
- (0,1)
- (-1,2)
- (2,0)
- (3,0)
- (4,-1)

Calculamos la distancia entre cada par de puntos:

Distancia entre (0, 0) y (0, 1):

$$d = \sqrt{(0-0)^2 + (1-0)^2} = \sqrt{1} = 1 \tag{2}$$

Distancia entre (0, 0) y (-1, 2):

$$d = \sqrt{(-1-0)^2 + (2-0)^2} = \sqrt{1+4} = \sqrt{5} \tag{3}$$

Distancia entre (0, 0) y (2, 0):

$$d = \sqrt{(2-0)^2 + (0-0)^2} = \sqrt{4} = 2 \tag{4}$$

Distancia entre (0, 0) y (3, 0):

$$d = \sqrt{(3-0)^2 + (1-0)^2} = \sqrt{9} = 3 \tag{5}$$

Distancia entre (0, 0) y (4, -1):

$$d = \sqrt{(4-0)^2 + (-1-0)^2} = \sqrt{16+1} = \sqrt{17} \tag{6}$$

Distancia entre (0, 1) y (-1, 2):

$$d = \sqrt{(-1-0)^2 + (2-1)^2} = \sqrt{1+1} = \sqrt{2} \tag{7}$$

Distancia entre (0, 1) y (2, 0):

$$d = \sqrt{(2-0)^2 + (0-1)^2} = \sqrt{4+1} = \sqrt{5} \tag{8}$$

Distancia entre (0, 1) y (3, 0):

$$d = \sqrt{(3-0)^2 + (0-1)^2} = \sqrt{9+1} = \sqrt{10} \tag{9}$$

Distancia entre (0, 1) y (4, -1):

$$d = \sqrt{(4-0)^2 + (-1-1)^2} = \sqrt{16+4} = \sqrt{20} \tag{10}$$

Distancia entre (-1, 2) y (2, 0):

$$d = \sqrt{(2-(-1))^2 + (0-2)^2} = \sqrt{9+4} = \sqrt{13} \tag{11}$$

Distancia entre (-1, 2) y (3, 0):

$$d = \sqrt{(3-(-1))^2 + (0-2)^2} = \sqrt{16+4} = \sqrt{20} \tag{12}$$

Distancia entre (-1, 2) y (4, -1):

$$d = \sqrt{(4-(-1))^2 + (-1-2)^2} = \sqrt{25+9} = \sqrt{34} \tag{13}$$

Distancia entre (2, 0) y (3, 0):

$$d = \sqrt{(3-2)^2 + (0-0)^2} = \sqrt{1} = 1 \tag{14}$$

Distancia entre (2, 0) y (4, -1):

$$d = \sqrt{(4-2)^2 + (-1-0)^2} = \sqrt{4+1} = \sqrt{5} \tag{15}$$

Distancia entre (3, 0) y (4, -1):

$$d = \sqrt{(4-3)^2 + (-1-0)^2} = \sqrt{1+1} = \sqrt{2} \tag{16}$$

La matriz de distancias es:

Puntos	(0,0)	(0,1)	(-1,2)	(2,0)	(3,0)	(4,-1)
(0,0)	0	1	$\sqrt{5}$	2	3	$\sqrt{17}$
(0,1)	1	0	$\sqrt{2}$	$\sqrt{5}$	$\sqrt{10}$	$\sqrt{20}$
(-1,2)	$\sqrt{5}$	$\sqrt{2}$	0	$\sqrt{13}$	$\sqrt{20}$	$\sqrt{34}$
(2,0)	2	$\sqrt{5}$	$\sqrt{13}$	0	1	$\sqrt{5}$
(3,0)	3	$\sqrt{10}$	$\sqrt{20}$	1	0	$\sqrt{2}$
(4,-1)	$\sqrt{17}$	$\sqrt{20}$	$\sqrt{34}$	$\sqrt{5}$	$\sqrt{2}$	0

5.1. K-means con K=2

Usamos (0,0) y (4,-1) como centroides iniciales.El algoritmo K-means se realiza iterativamente. Tenemos que:

Distancia de (0,0) a:

$$(0,0) : 0 \tag{17}$$

$$(4,-1) : \sqrt{17} \tag{18}$$

Distancia de (0,1) a:

$$(0,0) : 1 \tag{19}$$

$$(4,-1) : \sqrt{20} \tag{20}$$

Distancia de (-1,2) a:

$$(0,0) : \sqrt{5} \tag{21}$$

$$(4,-1) : \sqrt{34} \tag{22}$$

Distancia de (2,0) a:

$$(0,0) : 2 \tag{23}$$

$$(4,-1) : \sqrt{5} \tag{24}$$

Distancia de (3,0) a:

$$(0,0) : 3 \tag{25}$$

$$(4,-1) : \sqrt{2} \tag{26}$$

Distancia de (4,-1) a:

$$(0,0) : \sqrt{17} \tag{27}$$

$$(4,-1) : 0 \tag{28}$$

Asignacion inicial de puntos a clusters: Cluster 1 (centroide((0,0)):(0,0),(0,1),(-1,2) Cluster 2 (centroide((4,-1)):(2,0),(3,0),(4,-1)

Recalulamos nuevamente los clusters: Nuevo centroide de Cluster 1 :

$$(\frac{0+0-1}{3}, \frac{0+1+2}{3}) = (-\frac{1}{3}, 1) \tag{29}$$

Nuevo centroide de Cluster 2 :

$$(\frac{2+3+4}{3}, \frac{0+0-1}{3}) = (3, -\frac{1}{3}) \tag{30}$$

Calculamos las distancias a los nuevos centroides:

Distancia de (0,0) a:

$$(-\frac{1}{3}, 1) : \frac{\sqrt{10}}{3} \tag{31}$$

$$(3, -\frac{1}{3}) : \frac{\sqrt{82}}{3} \tag{32}$$

Distancia de (0,1) a:

$$(-\frac{1}{3}, 1) : \frac{\sqrt{1}}{3} \tag{33}$$

$$(3, -\frac{1}{3}) : \frac{\sqrt{97}}{3} \tag{34}$$

Distancia de (-1,2) a:

$$(-\frac{1}{3},1) : \frac{\sqrt{13}}{3} \tag{35}$$

$$(3,-\frac{1}{3}) : \frac{\sqrt{193}}{3} \tag{36}$$

Distancia de (2,0) a:

$$(-\frac{1}{3},1) : \frac{\sqrt{52}}{9} \tag{37}$$

$$(3,-\frac{1}{3}) : \sqrt{15} \tag{38}$$

Distancia de (3,0) a:

$$(-\frac{1}{3},1) : \frac{\sqrt{109}}{3} \tag{39}$$

$$(3,-\frac{1}{3}) : \sqrt{15} \tag{40}$$

Distancia de (4,-1) a:

$$(-\frac{1}{3},1) : \sqrt{1} \tag{41}$$

$$(3,-\frac{1}{3}) : \frac{\sqrt{130}}{3} \tag{42}$$

Pertenencia del punto (1,1)

Finalmente, asignamos (1,1) a los centroides nuevos:

- Distancia a:

$$(-\frac{1}{3},1) : \frac{1}{2} \tag{43}$$

- Distancia a:

$$(3,-\frac{1}{3}) : \sqrt{1+1} \tag{44}$$

Conclusión: El punto (1,1) esta mas cerca del centroide (-1/3,1)

Por lo tanto, pertenece al cluster inicializado por (0,0).

6. Ejercicio 6

Demostración: La distancia euclidiana entre $\mathbf{x}_i$ y μ se define como:

$$d_2(\mathbf{x}_i, \mu) = \|\mathbf{x}_i - \mu\|_2$$

Por lo tanto, queremos minimizar la suma de los cuadrados de estas distancias:

$$\hat{\mu} = \arg\min_{\mu} \sum_{i=1}^{n} \|\mathbf{x}_i - \mu\|_2^2$$

Expandimos el término cuadrado:

$$\sum_{i=1}^{n} \|\mathbf{x}_i - \mu\|_2^2 = \sum_{i=1}^{n} (\mathbf{x}_i - \mu)^\top (\mathbf{x}_i - \mu)$$

$$= \sum_{i=1}^{n} (\mathbf{x}_i^\top \mathbf{x}_i - 2\mathbf{x}_i^\top \mu + \mu^\top \mu)$$

$$= \sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - 2\sum_{i=1}^{n} \mathbf{x}_i^\top \mu + \sum_{i=1}^{n} \mu^\top \mu$$

$$= \sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - 2\mu^\top \sum_{i=1}^{n} \mathbf{x}_i + n\mu^\top \mu$$

Para encontrar el mínimo, derivamos con respecto a μ y establecemos la derivada igual a cero:

$$\frac{\partial}{\partial \mu} \left(\sum_{i=1}^{n} \mathbf{x}_i^\top \mathbf{x}_i - 2\mu^\top \sum_{i=1}^{n} \mathbf{x}_i + n\mu^\top \mu \right) = 0$$

$$-2\sum_{i=1}^{n} \mathbf{x}_i + 2n\mu = 0$$

$$n\mu = \sum_{i=1}^{n} \mathbf{x}_i$$

$$\mu = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

Por lo tanto, la media muestral $\hat{\mu}$ es:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

Esto demuestra que la media muestral $\hat{\mu}$ es la solución al problema de optimización.

7. Ejercicio 7

Supongamos que se tienen n observaciones, cada una con p características. Determina cuáles de los siguientes enunciados son verdaderos con respecto al análisis de clústers:

1. Podemos agrupar las n observaciones sobre la base de las p características para identificar subgrupos entre las observaciones.
2. Podemos agrupar las p características sobre la base de las n observaciones para descubrir subgrupos entre las características.
3. El análisis por clústers es parte del aprendizaje supervisado y es parte del análisis exploratorio de datos.

Análisis y Justificación

Enunciado 1: Verdadero. El análisis de clústers es una técnica utilizada para agrupar un conjunto de objetos (en este caso, observaciones) en subgrupos o clústers basados en las características que poseen. Al analizar n observaciones cada una con p características, el objetivo es identificar similitudes y diferencias entre las observaciones para agruparlas en clústers de manera que las observaciones dentro de un mismo clúster sean más similares entre sí que con las de otros clústers. Esto es un uso estándar del análisis de clústers.

Enunciado 2: Verdadero. Aunque menos común que agrupar observaciones, también es posible agrupar características en base a las observaciones. Este método se conoce como clúster de características (*feature clustering*) y se utiliza para identificar grupos de características que tienen patrones similares de valores a través de las observaciones. Esto puede ser útil, por ejemplo, en la reducción de dimensionalidad y en la interpretación de datos, donde descubrir grupos de características relacionadas puede proporcionar información sobre la estructura interna de los datos.

Enunciado 3: Parcialmente verdadero. El análisis de clústers se considera principalmente una técnica de aprendizaje no supervisado, ya que no se requiere una variable de respuesta o etiquetas conocidas durante el proceso de agrupamiento. Sin embargo, el análisis de clústers es una herramienta fundamental en el análisis exploratorio de datos (*Exploratory Data Analysis*, EDA), que es el proceso de resumir las principales características de un conjunto de datos, a menudo con métodos visuales. EDA se utiliza para descubrir patrones, detectar anomalías y formar hipótesis basadas en los datos disponibles.

8. Ejercicio 8

(*) Realiza un algoritmo de K-means (K = 2) dadas las asignaciones siguientes:

Observación	X_1	X_2	Clúster inicial
1	1	3	2
2	0	4	1
3	6	2	2
4	5	2	1
5	6	21	2

Cuadro 2: Asignaciones iniciales para K-means

Determina las asignaciones finales de los agrupamientos.

Solución: Para resolver el problema de K-means con $K = 2$, seguimos estos pasos:

1. **Inicialización:** Usamos las asignaciones iniciales dadas en la tabla. 2. **Cálculo de los centroides:** Calculamos los centroides de los dos clústeres iniciales. 3. **Reasignación:** Asignamos cada observación al clúster cuyo centroide esté más cercano. 4. **Repetición:** Repetimos los pasos 2 y 3 hasta que las asignaciones no cambien.

Paso 1: Inicialización Dadas las asignaciones iniciales:

$$\begin{aligned} &\text{Clúster 1:} \quad (0,4), (5,2) \\ &\text{Clúster 2:} \quad (1,3), (6,2), (6,21) \end{aligned}$$

Paso 2: Cálculo de los centroides iniciales

$$\text{Centroide del Clúster 1:} \quad \mu_1 = \left(\frac{0+5}{2}, \frac{4+2}{2} \right) = (2.5, 3)$$

$$\text{Centroide del Clúster 2:} \quad \mu_2 = \left(\frac{1+6+6}{3}, \frac{3+2+21}{3} \right) = (4.33, 8.67)$$

Paso 3: Reasignación Calculamos las distancias euclidianas de cada punto a los centroides y reasignamos las observaciones:

Distancias al centroide μ_1 (Clúster 1):

$$\begin{aligned} d((1,3), \mu_1) &= \sqrt{(1-2.5)^2 + (3-3)^2} = 1.5 \\ d((0,4), \mu_1) &= \sqrt{(0-2.5)^2 + (4-3)^2} = \sqrt{6.25+1} = 2.69 \\ d((6,2), \mu_1) &= \sqrt{(6-2.5)^2 + (2-3)^2} = \sqrt{12.25+1} = 3.61 \\ d((5,2), \mu_1) &= \sqrt{(5-2.5)^2 + (2-3)^2} = \sqrt{6.25+1} = 2.69 \\ d((6,21), \mu_1) &= \sqrt{(6-2.5)^2 + (21-3)^2} = \sqrt{12.25+324} = 18.33 \end{aligned}$$

Distancias al centroide μ_2 (Clúster 2):

$$d((1,3),\mu_2) = \sqrt{(1-4.33)^2 + (3-8.67)^2} = \sqrt{11.09 + 32.22} = 6.17$$
$$d((0,4),\mu_2) = \sqrt{(0-4.33)^2 + (4-8.67)^2} = \sqrt{18.75 + 21.81} = 6.61$$
$$d((6,2),\mu_2) = \sqrt{(6-4.33)^2 + (2-8.67)^2} = \sqrt{2.79 + 44.83} = 6.86$$
$$d((5,2),\mu_2) = \sqrt{(5-4.33)^2 + (2-8.67)^2} = \sqrt{0.45 + 44.83} = 6.73$$
$$d((6,21),\mu_2) = \sqrt{(6-4.33)^2 + (21-8.67)^2} = \sqrt{2.79 + 154.67} = 12.58$$

Reasignamos cada observación al clúster más cercano:

$$(1,3) \rightarrow \text{Clúster 1}$$
$$(0,4) \rightarrow \text{Clúster 1}$$
$$(6,2) \rightarrow \text{Clúster 1}$$
$$(5,2) \rightarrow \text{Clúster 1}$$
$$(6,21) \rightarrow \text{Clúster 2}$$

Paso 4: Repetición Calculamos los nuevos centroides con las nuevas asignaciones y repetimos hasta que las asignaciones no cambien.

9. Ejercicio 9

Considera los siguientes puntos:

$$(2,10), (2,5), (8,4), (5,8), (7,5), (6,4), (1,2), (4,9)$$

Usando el algoritmo de K-means, agrupa los puntos en 3 clústers.

9.1. Solución

Usamos Python para ejecutar el algoritmo de K-means. El código es el siguiente:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import linkage, dendrogram

# Definir los puntos
points = np.array([[2, 10], [2, 5], [8, 4], [5, 8], [7, 5], [6, 4], [1, 2], [4, 9]])

# (1) Algoritmo de K-means
kmeans = KMeans(n_clusters=3)
kmeans.fit(points)
labels = kmeans.labels_
centroids = kmeans.cluster_centers_

# Visualizaci n de los cl sters K-means
plt.figure(figsize=(12, 6))

plt.subplot(121)
plt.scatter(points[:, 0], points[:, 1], c=labels, cmap='viridis')
plt.scatter(centroids[:, 0], centroids[:, 1], s=300, c='red', marker='X')
plt.title('K-means Clustering (K=3)')
plt.xlabel('X1')
plt.ylabel('X2')

# (2) An lisis de cl ster: Single-Link, Complete-Link, Average-Link
linkage_methods = ['single', 'complete', 'average']
```

```
plt.subplot(122)
for method in linkage_methods:
    linked = linkage(points, method=method)
    dendrogram(linked, labels=range(1, len(points) + 1), orientation='top')
    plt.title(f'{method.capitalize()} Linkage')
    plt.xlabel('Sample index')
    plt.ylabel('Distance')
    plt.show()
```

9.2. Análisis de Clúster

Usando single-link, complete-link y average-link, agrupa los puntos dados.

9.3. Single-Link

Single-link clustering se basa en la distancia mínima entre puntos de diferentes clústers. La implementación en Python es la siguiente:

```
linked = linkage(points, method='single')
dendrogram(linked, labels=range(1, len(points) + 1), orientation='top')
plt.title('Single Linkage')
plt.xlabel('Sample index')
plt.ylabel('Distance')
plt.show()
```

9.4. Complete-Link

Complete-link clustering se basa en la distancia máxima entre puntos de diferentes clústers. La implementación en Python es la siguiente:

```
linked = linkage(points, method='complete')
dendrogram(linked, labels=range(1, len(points) + 1), orientation='top')
plt.title('Complete Linkage')
plt.xlabel('Sample index')
plt.ylabel('Distance')
plt.show()
```

9.5. Average-Link

Average-link clustering se basa en la distancia promedio entre puntos de diferentes clústers. La implementación en Python es la siguiente:

```
linked = linkage(points, method='average')
dendrogram(linked, labels=range(1, len(points) + 1), orientation='top')
plt.title('Average Linkage')
plt.xlabel('Sample index')
plt.ylabel('Distance')
plt.show()
```

9.6. Conclusión

Este documento presenta la implementación y resultados del algoritmo de K-means y los métodos de single-link, complete-link y average-link para el análisis de clúster.

9.7. Codigo

Definir los puntos points = np.array([[2, 10], [2, 5], [8, 4], [5, 8], [7, 5], [6, 4], [1, 2], [4, 9]])

(1) Algoritmo de K-means kmeans = KMeans(n$_c lusters = 3) kmeans.fit(points) labels = kmeans.labels_c entroids = kmeans.cluster_c enters$

Visualización de los clústers K-means plt.figure(figsize=(12, 6))

plt.subplot(121) plt.scatter(points[:, 0], points[:, 1], c=labels, cmap='viridis') plt.scatter(centroids[:, 0], centroids[:, 1], s=300, c='red', marker='X') plt.title('K-means Clustering (K=3)') plt.xlabel('X1') plt.ylabel('X2')

(2) Análisis de clúster: Single-Link, Complete-Link, Average-Link linkage$_m ethods = ['single','complete','average']$

plt.subplot(122) for method in linkage$_m ethods : linked = linkage(points, method = method) dendrogram(linked, lab range(1, len(points)+1), orientation =' top') plt.title(f'method.capitalize() Linkage') plt.xlabel('Sampleindex') plt.y$

10. Ejercicio 10

En este estudio, construiremos una base de datos de 20 tickers (incluyendo 10 empresas con tickers terminados en .MX) en un umbral de tiempo de 6 años. Consideraremos las siguientes estadísticas:

1. Movimientos finales: Se obtienen a través de los precios diarios, tomando el precio más alto menos el precio más bajo en ese día.
2. Rendimientos: Se obtienen como el precio de la fecha corriente entre el precio del día anterior menos 1.

Realizaremos un análisis de clúster con respecto a tales estadísticas y extraeremos conclusiones. Adicionalmente, realizaremos una simulación de Monte Carlo para obtener la frontera eficiente de Markowitz, determinando el índice de Sharpe y los pesos óptimos en el portafolio.

10.1. Construcción de la Base de Datos

Para la construcción de la base de datos, utilizaremos Python. El código a continuación ilustra el proceso.

```
import numpy as np
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from scipy.optimize import minimize

# Definir los tickers
tickers = ['AAPL', 'MSFT', 'GOOGL', 'AMZN', 'FB', 'BRK-B', 'JNJ', 'JPM', 'V', 'PG',
           'WALMEX.MX', 'FEMSAUBD.MX', 'GMEXICOB.MX', 'AMXL.MX', 'BIMBOA.MX',
           'CEMEXCPO.MX', 'TLEVISACPO.MX', 'GFNORTEO.MX', 'ALSEA.MX', 'BSMXB.MX']

# Descargar datos
data = yf.download(tickers, start="2015-01-01", end="2021-12-31")

# Calcular movimientos finales
data['Movement'] = data['High'] - data['Low']

# Calcular rendimientos
data['Returns'] = data['Adj Close'].pct_change()

# Resumir estad sticas
movements = data['Movement'].mean()
returns = data['Returns'].mean()
```

```
stats = pd.DataFrame({'Movements': movements, 'Returns': returns})
print(stats)

# An lisis de cl ster
kmeans = KMeans(n_clusters=3)
kmeans.fit(stats)
labels = kmeans.labels_

# Visualizaci n de los cl sters
plt.scatter(stats['Movements'], stats['Returns'], c=labels, cmap='viridis')
plt.xlabel('Movements')
plt.ylabel('Returns')
plt.title('Cluster Analysis')
plt.show()

# Simulaci n de Monte Carlo para la frontera eficiente de Markowitz
returns = data['Returns'].dropna()
mean_returns = returns.mean()
cov_matrix = returns.cov()

def portfolio_performance(weights, mean_returns, cov_matrix):
    returns = np.sum(mean_returns * weights) * 252
    std = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights))) * np.sqrt(252)
    return std, returns

def neg_sharpe_ratio(weights, mean_returns, cov_matrix, risk_free_rate):
    p_var, p_ret = portfolio_performance(weights, mean_returns, cov_matrix)
    return -(p_ret - risk_free_rate) / p_var

def get_ef_weights(mean_returns, cov_matrix, risk_free_rate):
    num_assets = len(mean_returns)
    args = (mean_returns, cov_matrix, risk_free_rate)
    constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})
    bounds = tuple((0, 1) for asset in range(num_assets))
    result = minimize(neg_sharpe_ratio, num_assets*[1./num_assets,], args=args,
                      method='SLSQP', bounds=bounds, constraints=constraints)
    return result

risk_free_rate = 0.01
optimal_weights = get_ef_weights(mean_returns, cov_matrix, risk_free_rate)
sharpe_ratio = -neg_sharpe_ratio(optimal_weights.x, mean_returns, cov_matrix, risk_f

print('Optimal Weights:', optimal_weights.x)
print('Sharpe Ratio:', sharpe_ratio)
```

10.2. Resultados

10.3. Movimientos Finales y Rendimientos

La tabla a continuación muestra las estadísticas resumidas de los movimientos finales y rendimientos de cada ticker:

```
print(stats)
```

10.4. Análisis de Clúster

El análisis de clúster se muestra en la Figura 1, agrupando los tickers en 3 clústers con base en sus movimientos finales y rendimientos.

10.5. Simulación de Monte Carlo y Frontera Eficiente de Markowitz

La simulación de Monte Carlo y la obtención de la frontera eficiente de Markowitz muestran que el índice de Sharpe es **sharpe_ratio** y los pesos óptimos en el portafolio son:

```
print('Optimal Weights:', optimal_weights.x)
```

10.6. Conclusiones

El análisis de clúster y la simulación de Monte Carlo nos proporcionan una visión clara de cómo se agrupan los tickers con base en sus movimientos finales y rendimientos, y nos permiten construir un portafolio óptimo de acuerdo a la teoría de Markowitz.