TAREA MINERÍA DE DATOS 2

Adriana Acero Fernández-Villacañas

Máster en Big Data, Data Science e Inteligencia Artificial
Universidad Complutense de Madrid

Índice

0. Descripción del dataset

El dataset de *penguins* de la librería *seaborn* en Python muestra información detallada sobre tres especies de pingüinos en relación a su morfología y hábitat. Las variables son:

- **species**: Esta variable categórica identifica la especie del pingüino. El conjunto incluye tres especies distintas: *Adelie*, *Chinstrap*, y *Gentoo*.
- **island**: Indica la isla en la cual se recopilaron los datos, siendo estas *Biscoe*, *Dream*, y *Torgersen*.
- **bill length mm**: Longitud del pico del pingüino, medida en milímetros.
- **bill depth mm**: Profundidad del pico del pingüino, también medida en milímetros.
- **flipper length mm**: Longitud de las aletas del pingüino, medida en milímetros.
- **body mass g**: Masa corporal del pingüino, expresada en gramos.
- **sex**: Género del pingüino, que puede ser *Male* (macho), *Female* (hembra), o *NaN* en casos donde la información no está disponible.

En apartados posteriores se va a explorar el conjunto de datos y aplicar técnicas de reducción de dimensionalidad como el Análisis de Componentes Principales y agrupamiento (clustering) tanto jerárquico como no jerárquico.

0.1. Carga y exploración

1. Carga del conjunto de datos:

```
penguins = sns.load_dataset('penguins')
```

El conjunto contiene 344 filas inicialmente. Además, contiene 11 filas con *NaN* que serán eliminadas.

```
nans = penguins[penguins.isna().any(axis=1)]
print(nans)
```

```
penguins = penguins.dropna()
```

2. Estadísticas descriptivas básicas:

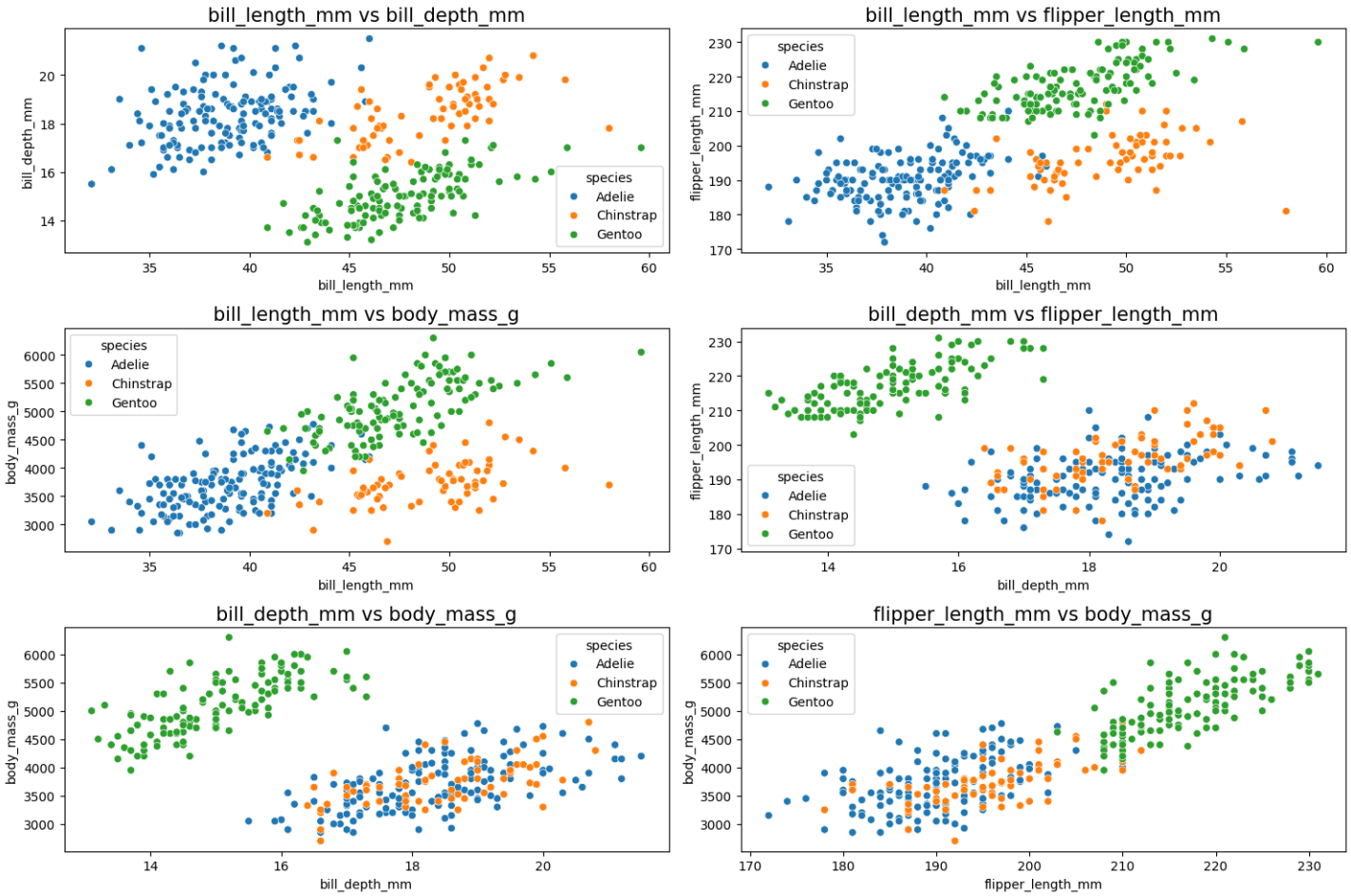
```
descriptivos_num = penguins.describe().T

for num in variables_numericas:
    descriptivos_num.loc[num, "Asimetria"] = penguins[num].skew()
    descriptivos_num.loc[num, "Kurtosis"] = penguins[num].kurtosis()
    descriptivos_num.loc[num, "Rango"] = np.ptp(penguins[num].dropna().values)

descriptivos_num
```

	count	mean	std	min	25%	50%	75%	max	Asimetria	Kurtosis	Rango
bill_length_mm	333.0	43.992793	5.468668	32.1	39.5	44.5	48.6	59.6	0.045340	-0.883418	27.5
bill_depth_mm	333.0	17.164865	1.969235	13.1	15.6	17.3	18.7	21.5	-0.149720	-0.891960	8.4
flipper_length_mm	333.0	200.966967	14.015765	172.0	190.0	197.0	213.0	231.0	0.360148	-0.961241	59.0
body_mass_g	333.0	4207.057057	805.215802	2700.0	3550.0	4050.0	4775.0	6300.0	0.472246	-0.733489	3600.0

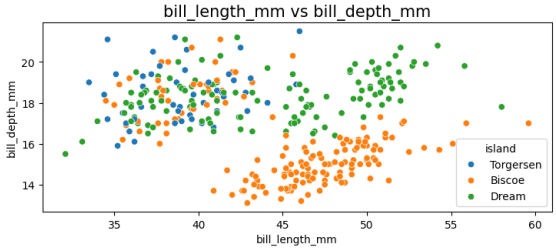
Se observan desviaciones estándar considerables en las variables **body_mass_g** y **flipper_length_mm** lo cual refleja una variabilidad significativa en estas características relacionadas con el peso y la longitud de la aleta.

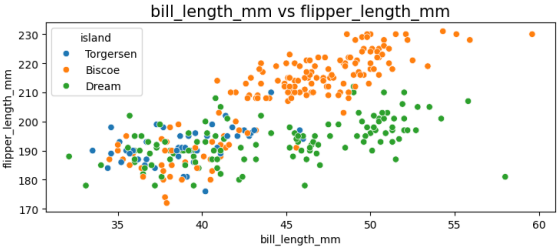
3. Gráficos de dispersión para visualizar las diferencias entre las especies en relación con las variables numéricas, generados con el siguiente código de Python:

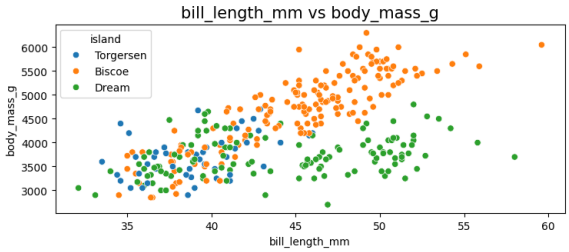
```
combinations = list(itertools.combinations(variables_numericas, 2))

plt.figure(figsize=(15, 10))

for i, (x_var, y_var) in enumerate(combinations, 1):
    plt.subplot(len(combinations)//2, 2, i)
    sns.scatterplot(x=x_var, y=y_var, data=penguins, hue="species")
    plt.title(f"{x_var} vs {y_var}", size=15)

plt.tight_layout()
plt.show()
```

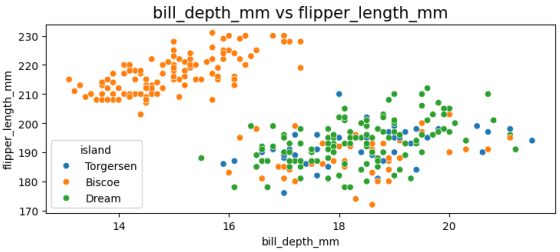


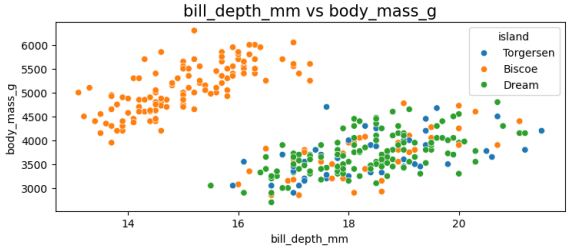
Las siguientes figuras ilustran los gráficos de dispersión, con los datos esta vez coloreados por isla y sexo respectivamente:

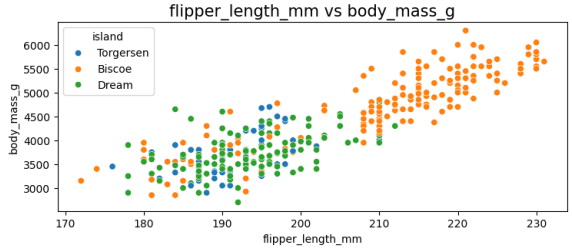

bill_length_mm vs bill_depth_mm
bill_depth_mm
bill_length_mm
island
Torgersen
Biscoe
Dream

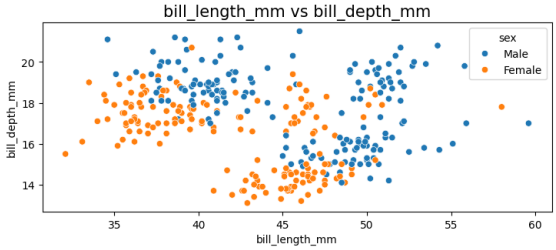

bill_length_mm vs flipper_length_mm
flipper_length_mm
bill_length_mm
island
Torgersen
Biscoe
Dream

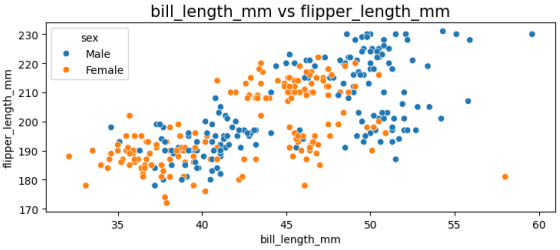

bill_length_mm vs body_mass_g
body_mass_g
bill_length_mm
island
Torgersen
Biscoe
Dream

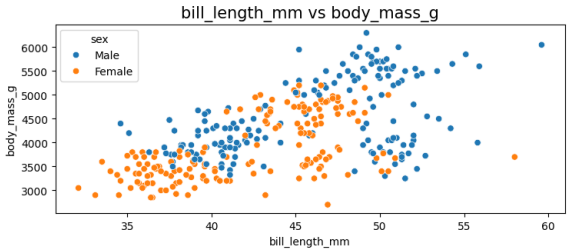

bill_depth_mm vs flipper_length_mm
flipper_length_mm
bill_depth_mm
island
Torgersen
Biscoe
Dream

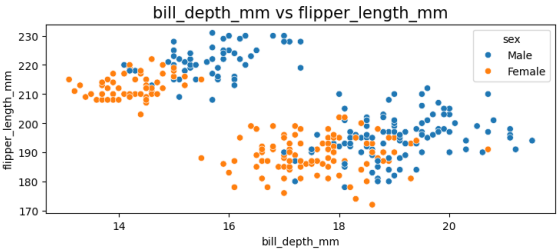

bill_depth_mm vs body_mass_g
body_mass_g
bill_depth_mm
island
Torgersen
Biscoe
Dream

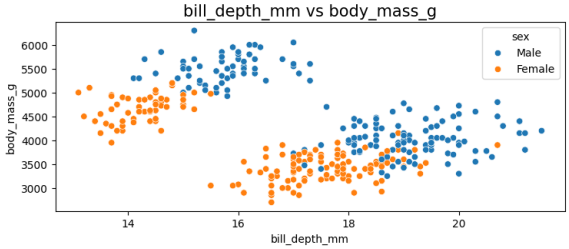

flipper_length_mm vs body_mass_g
body_mass_g
flipper_length_mm
island
Torgersen
Biscoe
Dream

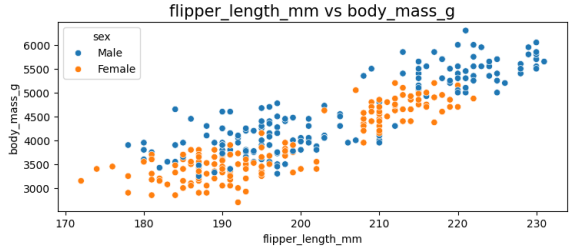

bill_length_mm vs bill_depth_mm
bill_depth_mm
bill_length_mm
sex
Male
Female


bill_length_mm vs flipper_length_mm
flipper_length_mm
bill_length_mm
sex
Male
Female


bill_length_mm vs body_mass_g
body_mass_g
bill_length_mm
sex
Male
Female

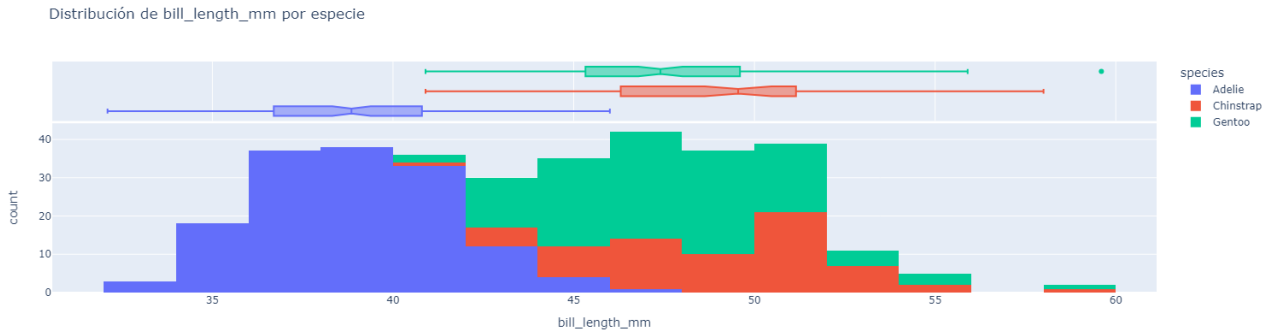

bill_depth_mm vs flipper_length_mm
flipper_length_mm
bill_depth_mm
sex
Male
Female

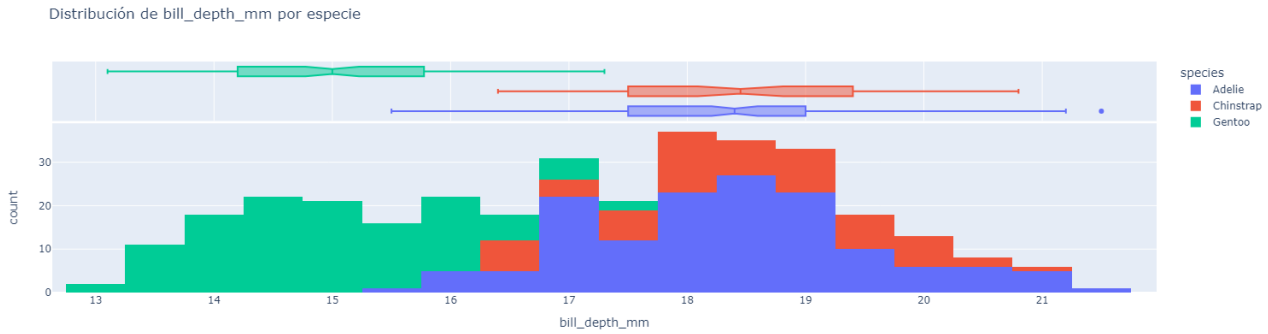

bill_depth_mm vs body_mass_g
body_mass_g
bill_depth_mm
sex
Male
Female

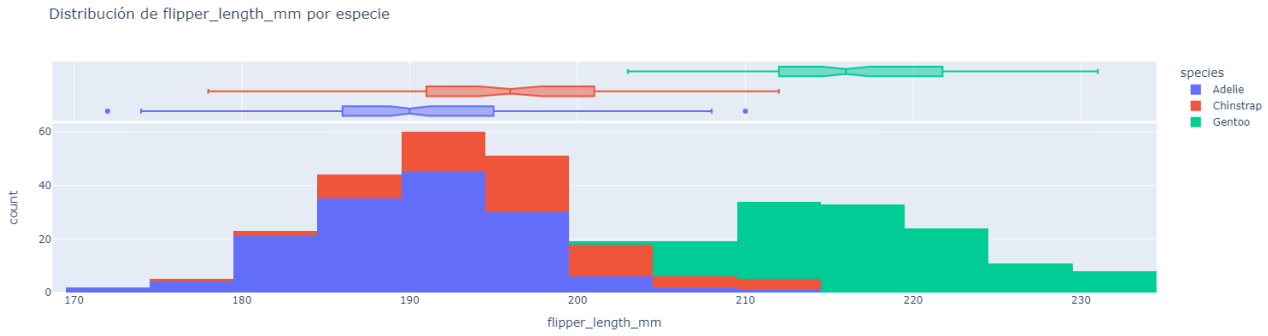

flipper_length_mm vs body_mass_g
body_mass_g
flipper_length_mm
sex
Male
Female

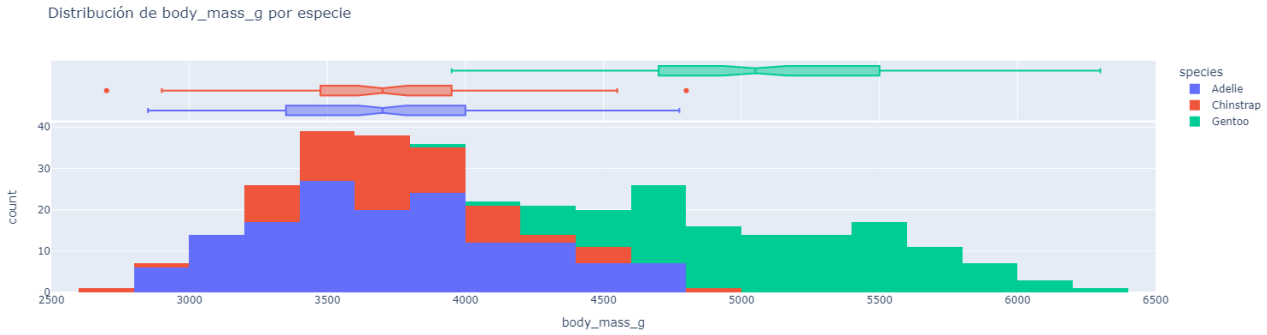
Adicionalmente, utilizando la librería Plotly Express se han generado para cada variable numérica un gráfico interactivo en el que se muestran histogramas de los datos y diagramas de cajas categorizadas por especie:

```
for var in variables_numericas:
    fig = px.histogram(penguins, x=var, color="species", marginal="box",
                       title=f'Distribución de {var} por especie')
    fig.show()
```







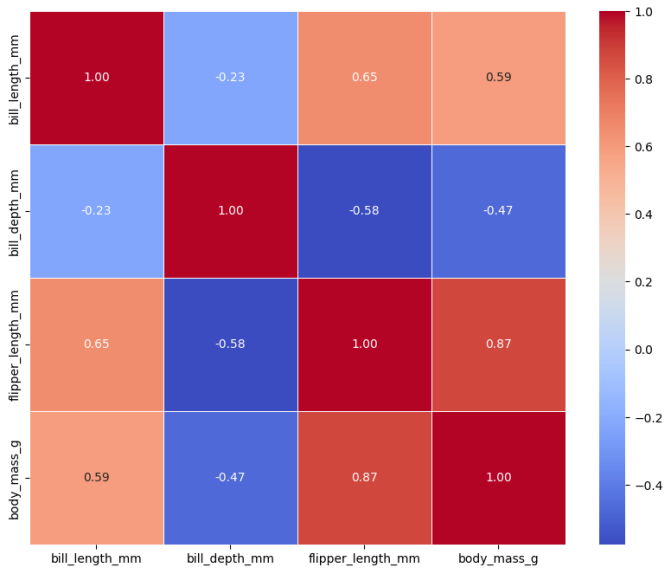


Estos gráficos han permitido una exploración dinámica de los datos, ya que pasando el cursor sobre los gráficos se pueden obtener detalles específicos, como conteos, cuartiles o rangos de datos.

1. Matriz de Correlaciones

Para responder a la pregunta planteada en la tarea se ha calculado la matriz de correlaciones entre las variables numéricas:

```
R = penguins.corr()

plt.figure(figsize=(10, 8))

sns.heatmap(R, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
```



Gracias a la matriz de correlación se pueden visualizar las asociaciones entre las características físicas de los pingüinos del estudio. Las variables más correlacionadas de forma inversa son **flipper_length_mm** (la longitud de la aleta) y **bill_depth_mm** (profundidad del pico), tal y como se muestra en la imagen, con un valor de la correlación de -0.58. Este hecho sugiere que los pingüinos con picos más profundos tienden a tener las aletas más cortas.

2. Análisis de Componentes Principales

En primer lugar, para poder llevar a cabo el análisis de componentes principales (PCA), se van a estandarizar los datos al ser un método sensible a la escala de los mismos.

```
scaler = StandardScaler()
penguins_numericas_estandarizadas = pd.DataFrame(
    scaler.fit_transform(penguins[variables_numericas]),
    columns=['{}_z'.format(variable) for variable in variables_numericas],
    index=penguins.index
)
```

Una vez estandarizadas, se aplica el PCA sin especificar el número de las componentes para obtenerlos todos. De esta manera se podrá estudiar la variabilidad explicada en todas las componentes.

```
pca = PCA()
fit = pca.fit(penguins_numericas_estandarizadas)
```

Adicionalmente, se examina la matriz de covarianzas (que es consistente con la matriz de correlaciones obtenida) y que va a ser utilizada en el PCA:

```
matriz_covarianzas = pca.get_covariance()
df_matriz_covarianzas = pd.DataFrame(
    matriz_covarianzas,
    index=penguins_numericas_estandarizadas.columns,
    columns=penguins_numericas_estandarizadas.columns
)

print(df_matriz_covarianzas)
```

```
                   bill_length_mm_z  bill_depth_mm_z  flipper_length_mm_z  \
bill_length_mm_z           1.003012        -0.229314             0.655063
bill_depth_mm_z           -0.229314         1.003012            -0.579532
flipper_length_mm_z        0.655063        -0.579532             1.003012
body_mass_g_z              0.591227        -0.473437             0.875608

                   body_mass_g_z
bill_length_mm_z        0.591227
bill_depth_mm_z        -0.473437
flipper_length_mm_z     0.875608
body_mass_g_z           1.003012
```

A continuación, se calculan los autovalores de la matriz de covarianzas anterior. Éstos miden la varianza que se captura en cada componente principal:

```
autovalores = fit.explained_variance_
print(autovalores)
```

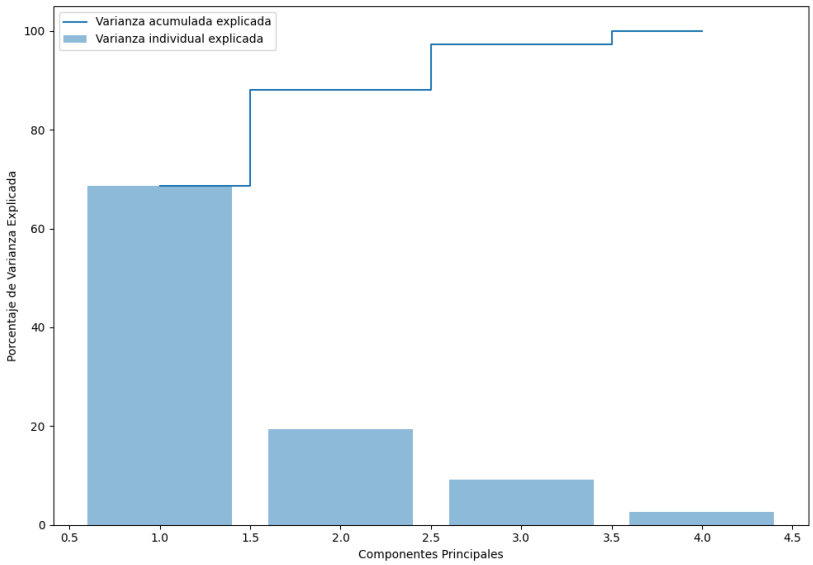
```
[2.75362487 0.7804609  0.36975289 0.10820954]
```

Se calcula la proporción de varianza explicada por componente como porcentaje de la varianza total:

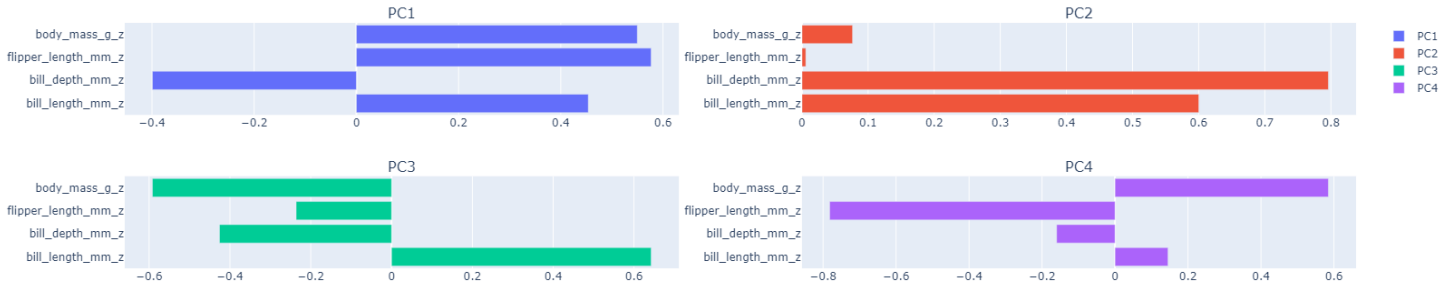
```
var_acumulada = np.cumsum(var_explicada)
print(var_acumulada)
```

```
[68.63389314 19.45292928  9.21606299  2.69711459]
```

De esta forma, se puede observar que entre las dos primeras componentes suman alrededor de un 89 % de varianza explicada. La siguiente figura muestra una gráfica por componentes, porcentaje de varianza explicada y su acumulación:



Asimismo, se pueden obtener las cargas de cada variable para cada componente:



Numéricamente:

```
cargas_pca = pd.DataFrame(
    pca.components_.T,
    index=penguins_numericas_estandarizadas.columns,
    columns=[f'PC{i+1}' for i in range(pca.n_components_)]
)

print(cargas_pca[['PC1', 'PC2','PC3','PC4']])
```

	PC1	PC2	PC3	PC4
bill_length_mm_z	0.453753	0.600195	0.642495	0.145170
bill_depth_mm_z	-0.399047	0.796170	-0.425800	-0.159904
flipper_length_mm_z	0.576825	0.005788	-0.236095	-0.781984
body_mass_g_z	0.549675	0.076464	-0.591737	0.584686

Estas cargas hacen referencia a los coeficientes de las variables originales en las componentes principales, indicando cómo contribuyen a la componente en cuestión. En particular, PC1 presenta cargas positivas significativas en **bill_length_mm_z**, **flipper_length_mm_z**, y **body_mass_g_z**, sugiriendo que esta componente está asociada con el tamaño de los pingüinos: picos más largos, aletas más largas y mayor masa corporal están correlacionados

con un valor más alto de PC1. Por otra parte, PC2 muestra cargas positivas significativas en **bill_length_mm_z** y **bill_depth_mm_z**, lo cual sugiere que está capturando variabilidad con respecto a las dimensiones de los picos.

Finalmente, se obtienen las coordenadas de los datos del dataset original en la nueva base formada por las componentes obtenidas.

```
resultados_pca = pd.DataFrame(fit.transform(penguins_numericas_estandarizadas),
                              columns=['Componente {}'.format(i) for i in range(1, len(autovalores)+1)],
                              index=penguins_numericas_estandarizadas.index)

print(resultados_pca.head())
```

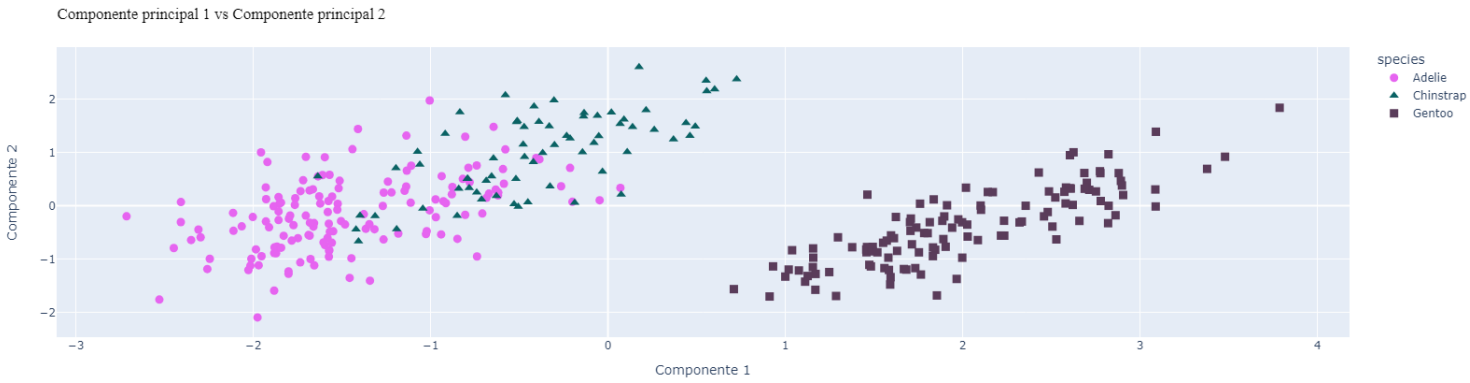
	Componente 1	Componente 2	Componente 3	Componente 4
0	-1.853593	0.032069	-0.234902	0.528397
1	-1.316254	-0.443527	-0.027470	0.401727
2	-1.376605	-0.161230	0.189689	-0.528662
4	-1.885288	-0.012351	-0.628873	-0.472893
5	-1.919981	0.817598	-0.701051	-0.196416

Para el análisis posterior, se ha decidido retener las dos primeras componentes, ya que entre las dos suman y explican aproximadamente un 89 % de la variabilidad. De esta forma, se reduce la complejidad del análisis del dataset, preservando información significativa sobre las distintas especies de pingüinos.

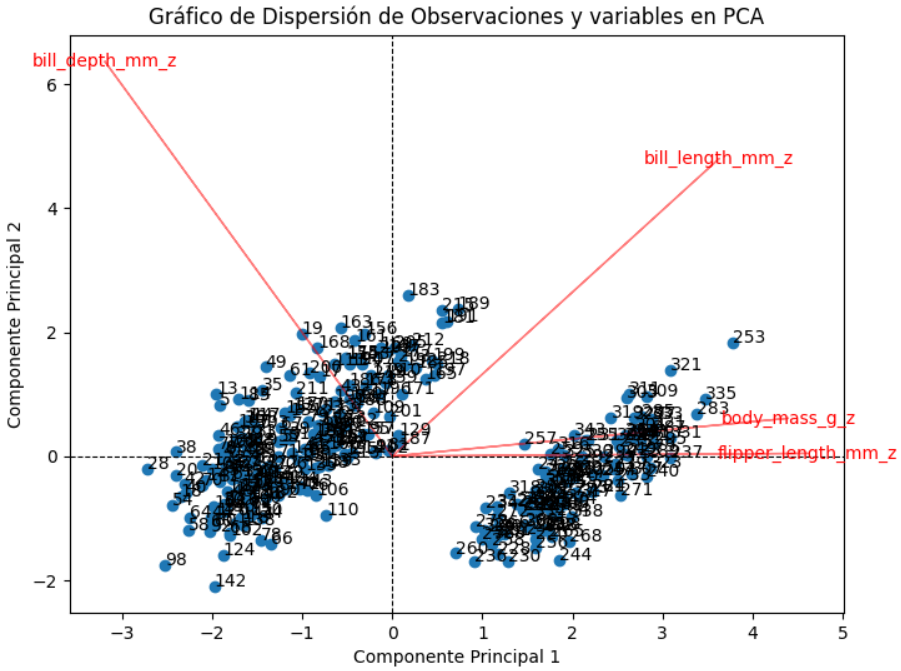
3. Análisis de (2) Componentes Principales

Se ha realizado nuevamente el análisis de componentes principales sobre los datos estandarizados, pero esta vez indicando dos componentes principales retenidas como se indicó anteriormente. A partir de ello se responderán a continuación a las preguntas propuestas en la tarea:

1. Gráficos que representan las variables en la nueva base de coordenadas formadas por las componentes principales PC1 y PC2. El primer gráfico de dispersión mostrado representa los datos en las nuevas coordenadas formadas por las componentes principales, diferenciando claramente y presentando los datos por especie:



En la siguiente imagen se puede observar, al igual que en la anterior, las observaciones en el nuevo espacio generado por las componentes princiales, y los vectores que representan las cargas de las variables:



Las proyecciones sobre la Componente Principal 1 más significativas son las de las variables **bill_length_mm_z**, **flipper_length_mm_z**, y **body_mass_g_z** (esto es, son las que tienen más carga en esta componente). En otras palabras, parece que la PC1 captura las medidas del tamaño de los pingüinos.
Análogamente, PC2 muestra cargas positivas significativas para **bill_length_mm_z** y **bill_depth_mm_z**, capturando así información independiente del tamaño general, como las formas de los picos de los pingüinos.

2. Según la imágen anterior coloreada por especie, se destacan los pingüinos *Gentoo* en la PC1, indicando que pueden tener un mayor tamaño que el resto, al representarse más a la derecha en el gráfico. Por este mismo motivo, la especie *Adelie* se podría decir que son los más pequeños, al situarse más hacia la izquierda. Con respecto a la forma de los picos (PC2), se destacan los *Chinstrap* cuyos picos son más profundos y largos.

3. Para construir un índice que valore de forma conjunta las características físicas se utilizará el resultado del análisis de componentes principales. Concretamente, se puede utilizar PC1 ya que es combinación lineal de las variables originales y captura la mayor varianza entre los atributos físicos de los pingüinos. De esta manera, el valor del índice para un pingüino se puede calcular multiplicando cada valor estandarizado de sus características por la carga correspondiente en la PC1 y sumando. Es decir, siendo Z la matriz de datos estandarizados

$$Z = \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,4} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,4} \\ \vdots & \vdots & \ddots & \vdots \\ z_{333,1} & z_{333,2} & \cdots & z_{333,4} \end{bmatrix}$$

, y P la matriz de cargas (autovectores) del PCA obtenida en la sección anterior

$$P = \begin{bmatrix} 0{,}453753 & 0{,}600195 & 0{,}642495 & 0{,}145170 \\ -0{,}399047 & 0{,}796170 & -0{,}425800 & -0{,}159904 \\ 0{,}576825 & 0{,}005788 & -0{,}236095 & -0{,}781984 \\ 0{,}549675 & 0{,}076464 & -0{,}591737 & 0{,}584686 \end{bmatrix}$$

. Se recuerda que la matriz de covarianza M se descompone como $M = PAP^{-1}$ por descomposición espectral, donde A es la matriz diagonal de autovalores.
Las nuevas coordenadas de las observaciones en el espacio de las componentes son las resultantes de ZP. De esta manera, el índice construido para un pingüino en específico es:
$\text{Índice}_{\text{PC1}} = z_{11}p_{11} + z_{12}p_{21} + \cdots + z_{1m}p_{m1}$, con $m = 4$ variables en este caso.

Sean
$\bar{z}_{\text{Adelie}} = \begin{bmatrix} \bar{z}_{\text{Adelie},1} & \bar{z}_{\text{Adelie},2} & \cdots & \bar{z}_{\text{Adelie},4} \end{bmatrix}$ el vector de las medias de las variables para *Adelie* y

$\bar{z}_{\text{Chinstrap}} = \begin{bmatrix} \bar{z}_{\text{Chinstrap},1} & \bar{z}_{\text{Chinstrap},2} & \cdots & \bar{z}_{\text{Chinstrap},4} \end{bmatrix}$ el vector de las medias de las variables para *Chinstrap*.

Cálculo del índice para las especies *Adelie* y *Chinstrap*:

$\text{Índice}_{\text{Adelie}} = \sum_{j=1}^{4} \bar{z}_{\text{Adelie},j} \cdot p_{j1}$
$\text{Índice}_{\text{Chinstrap}} = \sum_{j=1}^{4} \bar{z}_{\text{Chinstrap},j} \cdot p_{j1}$

donde p_{j1} son los valores de la primera columna de P. A continuación se muestra la implementación en código y el resultado de los dos índices para las especies *Adelie* y *Chinstrap*:

```
adelie_means = penguins_numericas_estandarizadas[penguins['species'] == 'Adelie'].mean()
chinstrap_means = penguins_numericas_estandarizadas[penguins['species'] == 'Chinstrap'].mean()

cargas_pc1 = pca.components_[0]

indice_adelie = np.dot(adelie_means, cargas_pc1)
indice_chinstrap = np.dot(chinstrap_means, cargas_pc1)

indice_adelie, indice_chinstrap
```

```
(-1.4597214389967106, -0.38860034552225986)
```

4. Clustering Jerárquico

Tal y como se mostró en la teoría, primero se han realizado los cálculos del clustering jerárquico con los datos sin estandarizar, para posteriormente replicarlos con los datos estandarizados. En las figuras 3 y 4 se muestran las matrices distancia en ambos contextos. La distancia entre observaciones seleccionada es la euclídea:

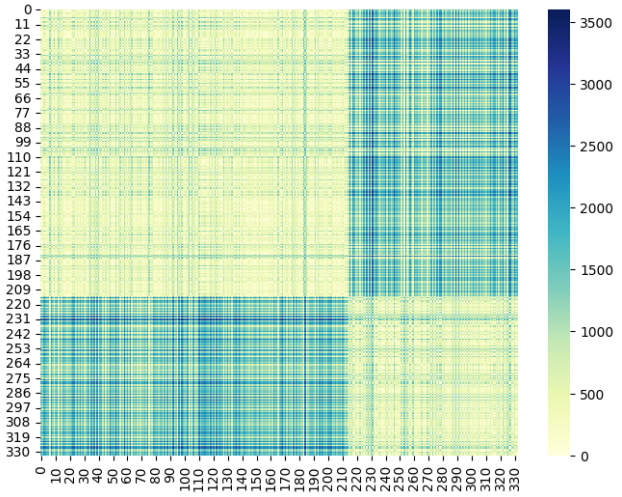


Figura 1: Matriz distancia con datos no estandarizados

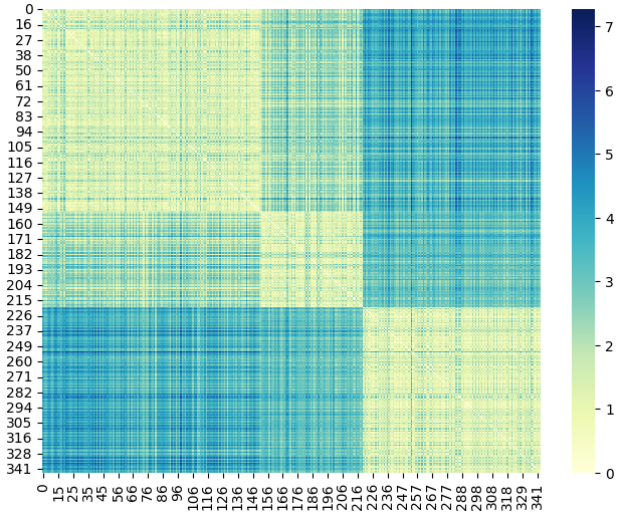


Figura 2: Matriz distancia con datos estandarizados

Estos mapas de calor revelan patrones tanto antes como después de la estandarización de los datos. En el conjunto de datos originales no estandarizados ya se empiezan a intuir dos patrones principales, indicando agrupaciones basadas en las características físicas de los pingüinos. Tras la estandarización, se revela otro patrón más. La estandarización de los datos mejora la detección de estructuras subyacientes, permitiendo una diferenciación más precisa. Esto es coherente con el conjunto de datos, ya que contenía tres posibles especies de pingüinos.

A continuación se presentan,como ejemplos,los mapas de calor y sus dendrogramas basados en la distancia *average* entre clusters:

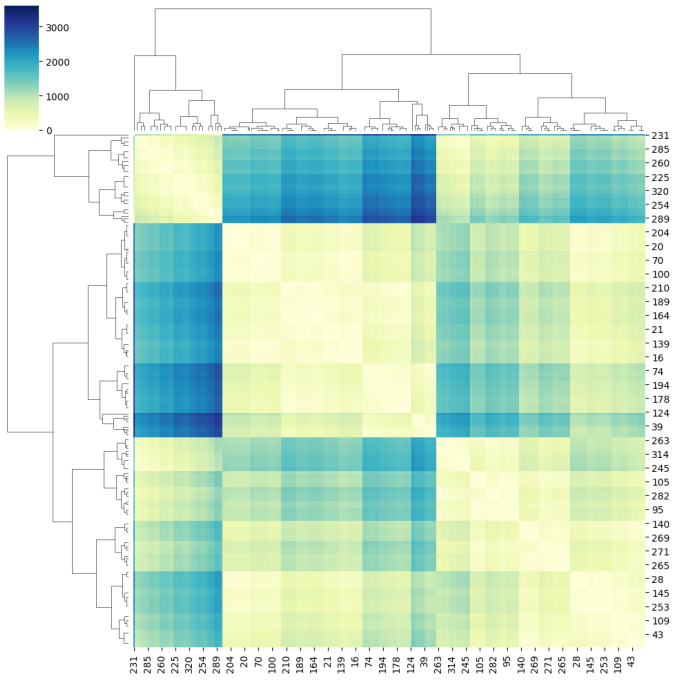


Figura 3: Matriz distancia con datos no estandarizados

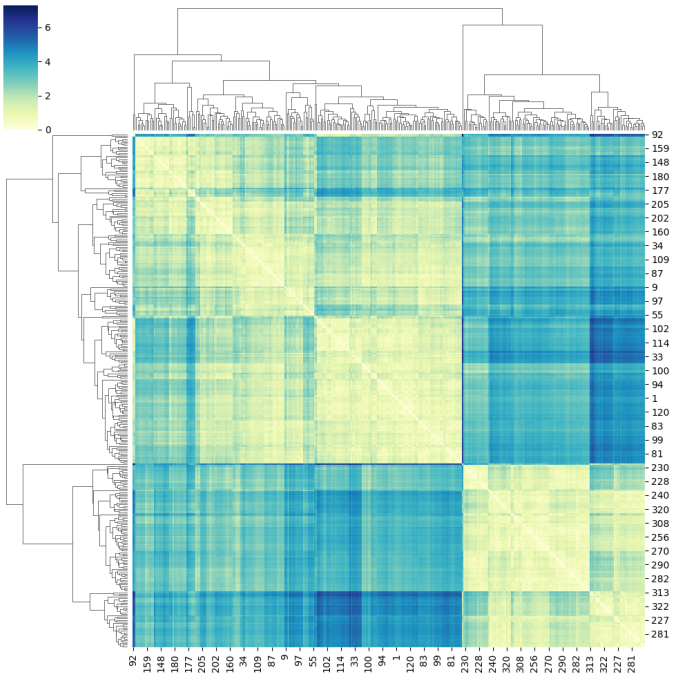


Figura 4: Matriz distancia con datos estandarizados

Una vez estandarizados los datos, se han generado dendrogramas para visualizar la relación entre los clústers, examinando las agrupaciones obtenidas para cada tipo de distancia vista en la parte teórica:

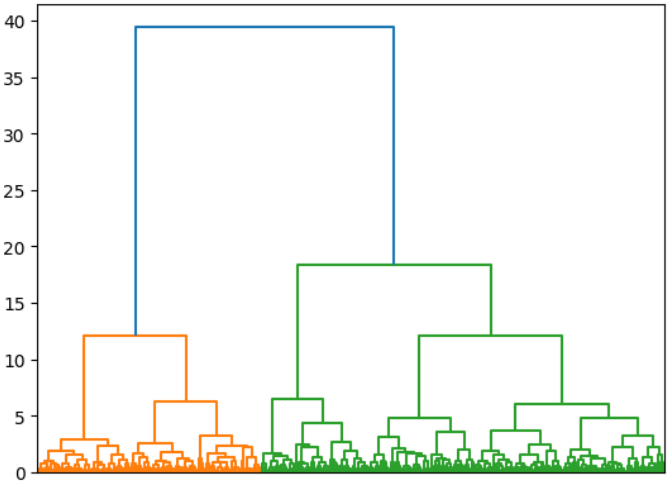


Figura 5: Distancia *Ward* entre clústers

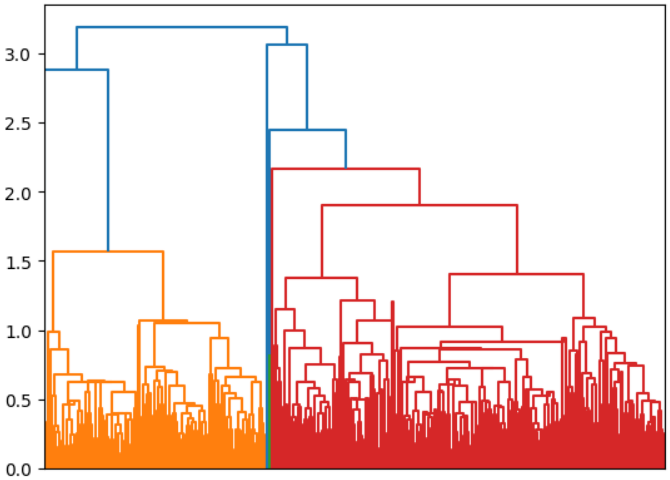


Figura 6: Distancia *Centroid* entre clústers

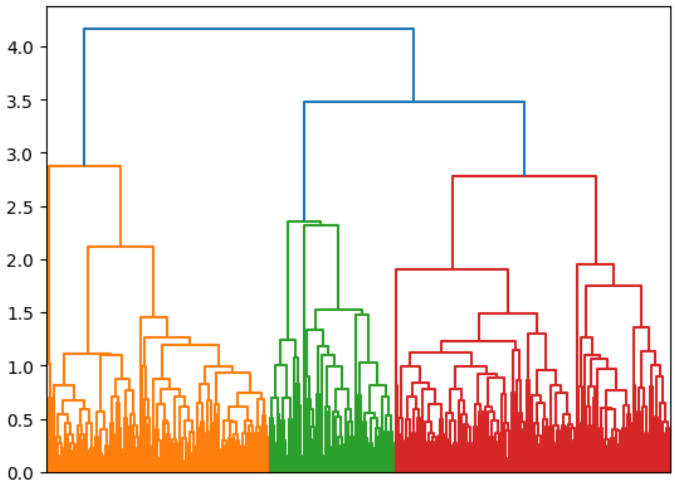


Figura 7: Distancia *Weighted* entre clústers

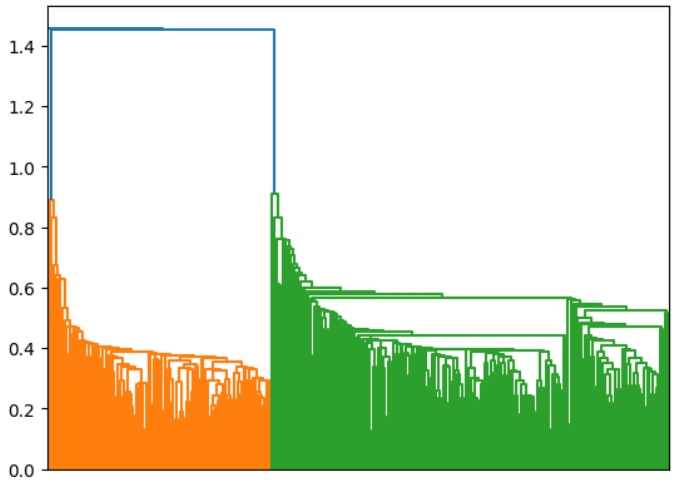


Figura 8: Distancia *Single* entre clústers

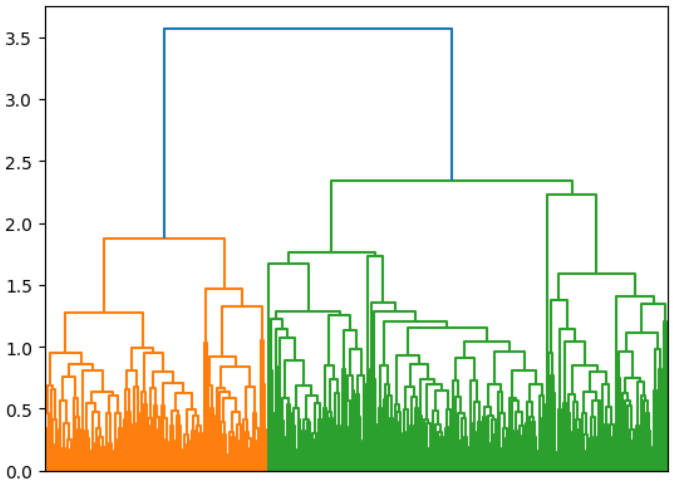


Figura 9: Distancia *Average* entre clústers

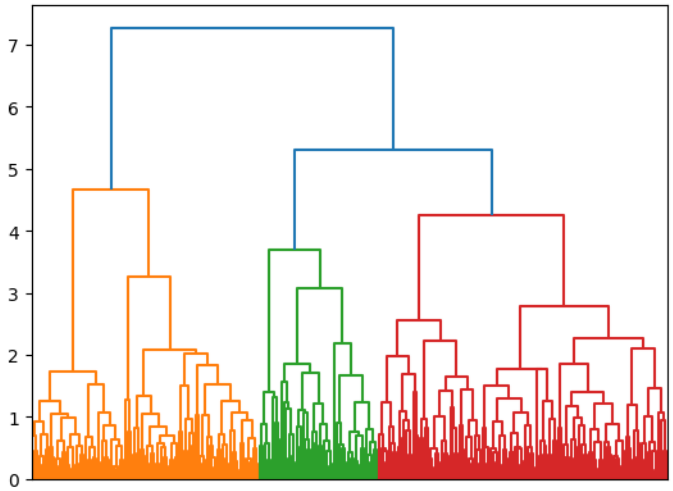


Figura 10: Distancia *Complete* entre clústers

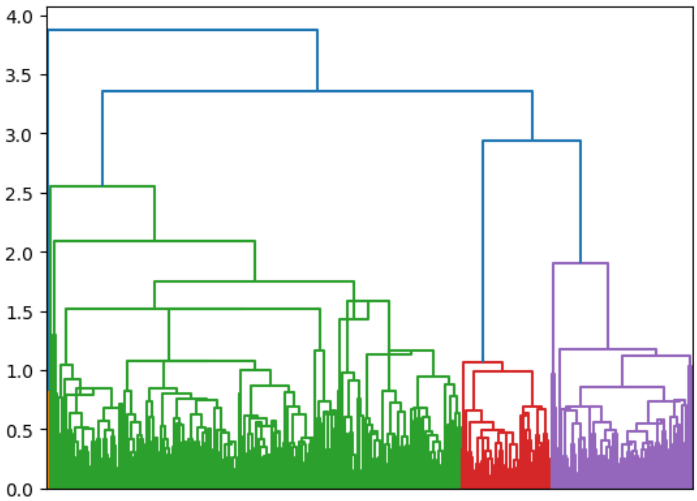


Figura 11: Distancia *Median* entre clústers

En vista de las figuras anteriores, se puede concluir la existencia de tres agrupaciones claramente definidas, coincidiendo así con el número de especies presentes en el dataset. Esta correspondencia puede sugerir que las diferencias entre los clústers pueden reflejar diferencias físicas a cada especie de pingüino.

Por último, se presenta la gráfica resultante del análisis jerárquico combinando con el PCA, mostrando la distribución de las observaciones en el espacio definido por las dos componentes principales. Los datos quedan agrupados en tres clústers diferentes, con características similares entre sí:

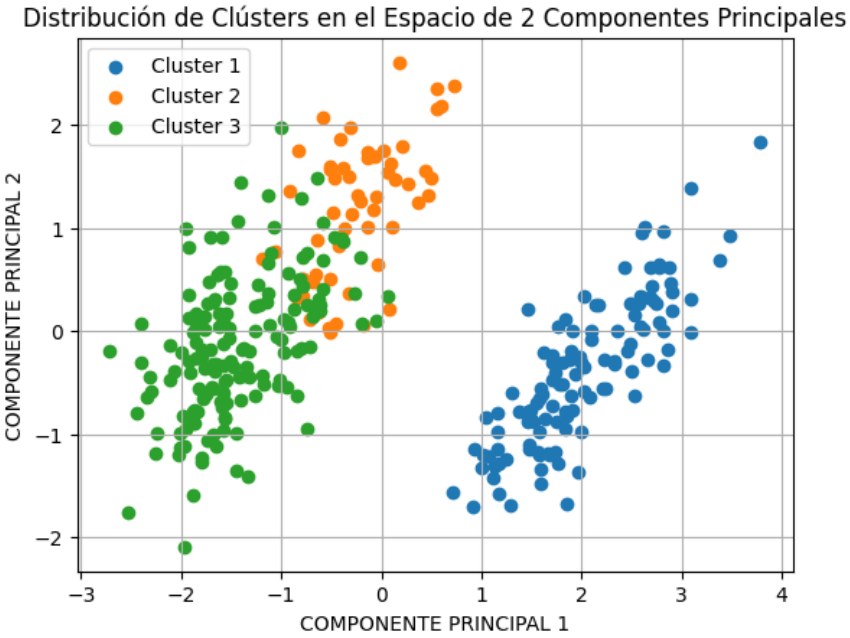


Figura 12: Distribución de clústers jerárquicos en el espacio de las dos primeras componentes princiales.

5. Agrupamiento KMeans

En primer lugar, se ha implementado el algoritmo de KMeans para diferentes números de clúster:

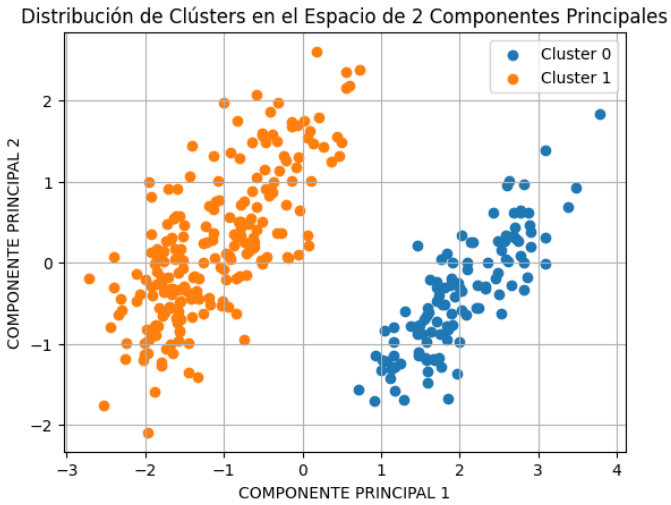


Figura 13: $k = 2$ clústers

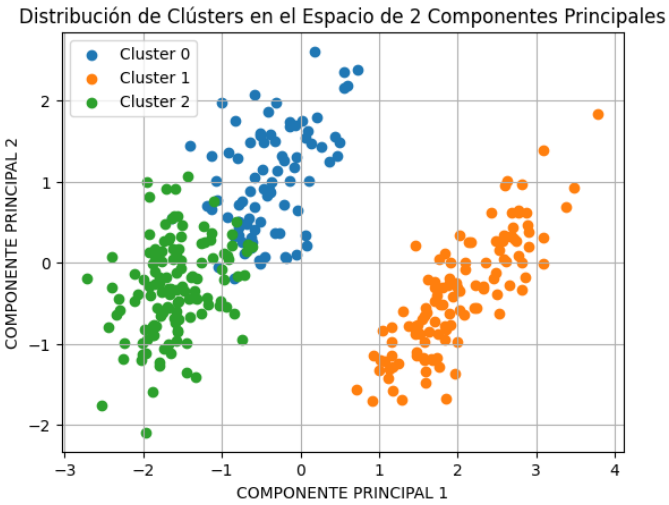


Figura 14: $k = 3$ clústers

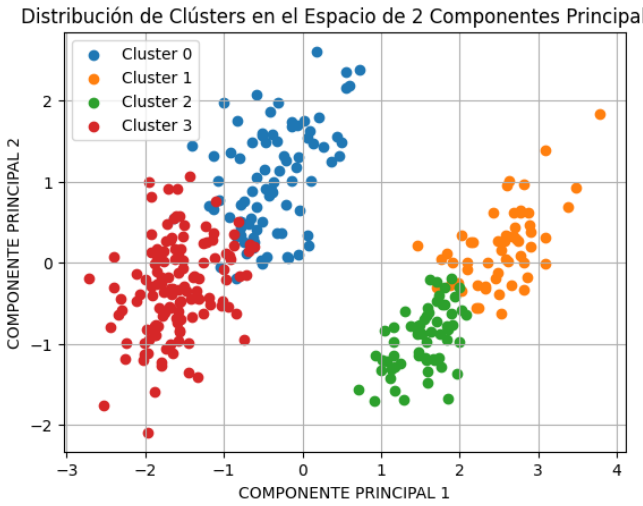


Figura 15: $k = 4$ clústers

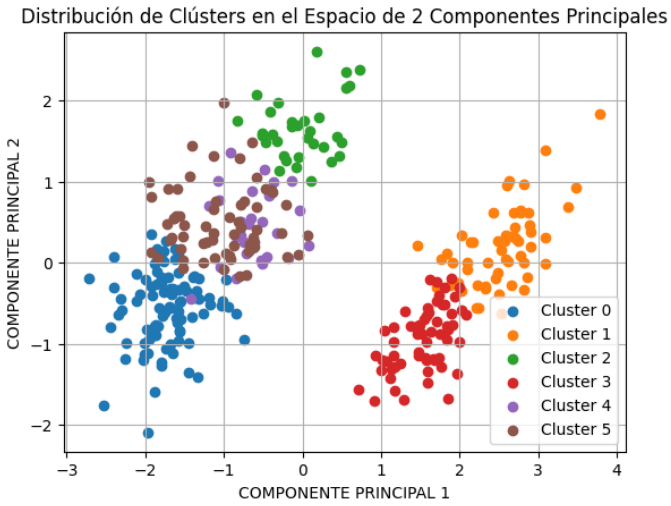


Figura 16: $k = 6$ clústers

A continuación, se ha generado la gráfica de la suma de las distancias al cuadrado de cada punto en un clúster a su centroide (WCSS) para diferentes valores de k para así buscar un punto donde el cambio comienza a disminuir más lentamente ("el codo", en el método del codo):

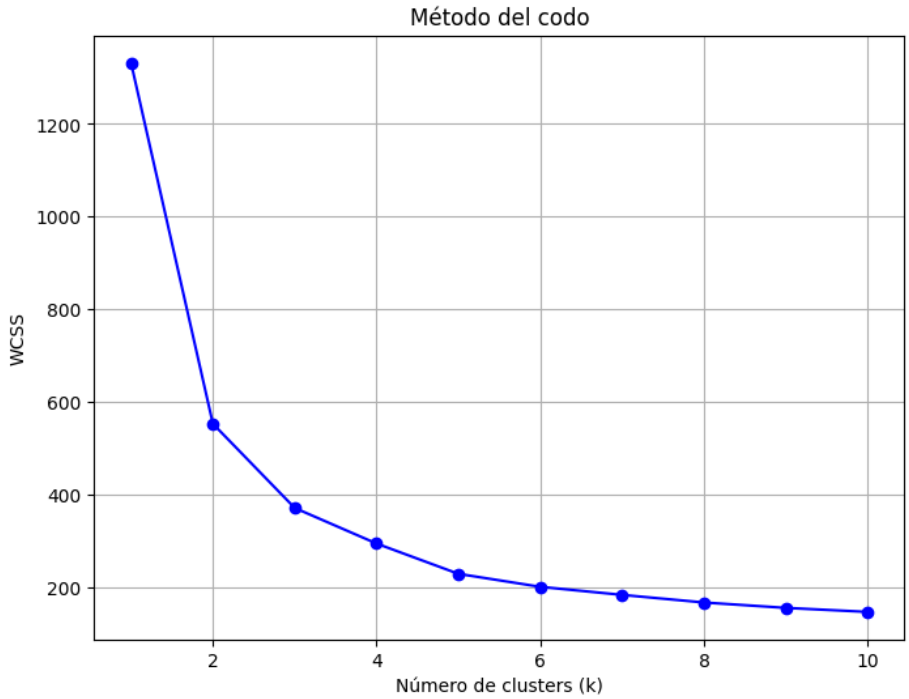


Figura 17: Número de clústers vs WCSS.

Se observa que el punto a partir del cual la curva comienza a aplanarse, reduciendo el beneficio de aumentar el número de clúster está entre $k = 3$ y $k = 4$. Es decir, con $k = 3$ clústers ya se puede capturar estructuras significativas en el conjunto de los datos. La elección de este número de clústers es coherente con el análisis por especie de la tarea.

6. Validación del agrupamiento

En este apartado se ha implementado el método de la puntuación de la silueta como métrica de validación del agrupamiento y así evaluar la calidad de los resultados anteriores. En la figura 18 se presentan las puntuaciones de silueta por número de clúster.

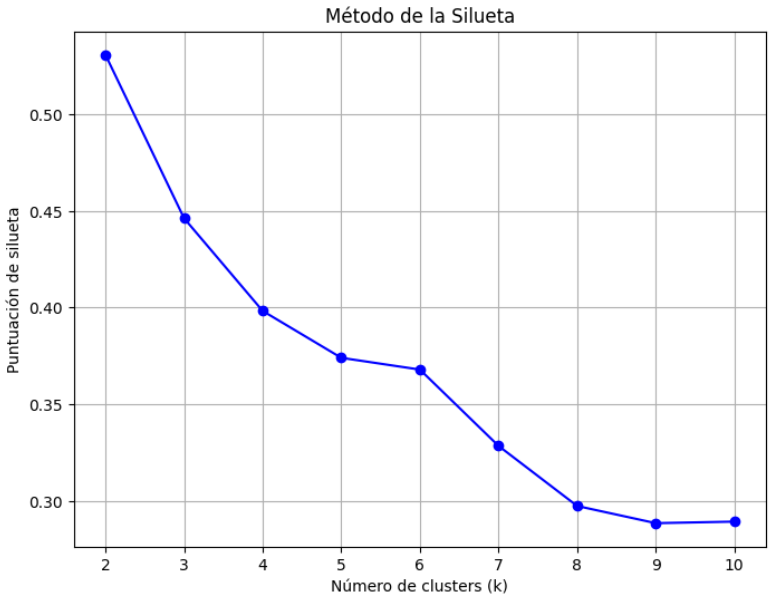


Figura 18: Puntuación de Silueta por número de clústers.

Parece que el valor más alto de silueta se obtiene cuando el número de clústers es $k = 2$, pero en el contexto de la tarea tiene más sentido $k = 3$ que es la segunda puntuación más alta. Por otra parte, la siguiente gráfica 19 mide qué tan similar es un punto a su propio clúster en comparación con otros clústeres.

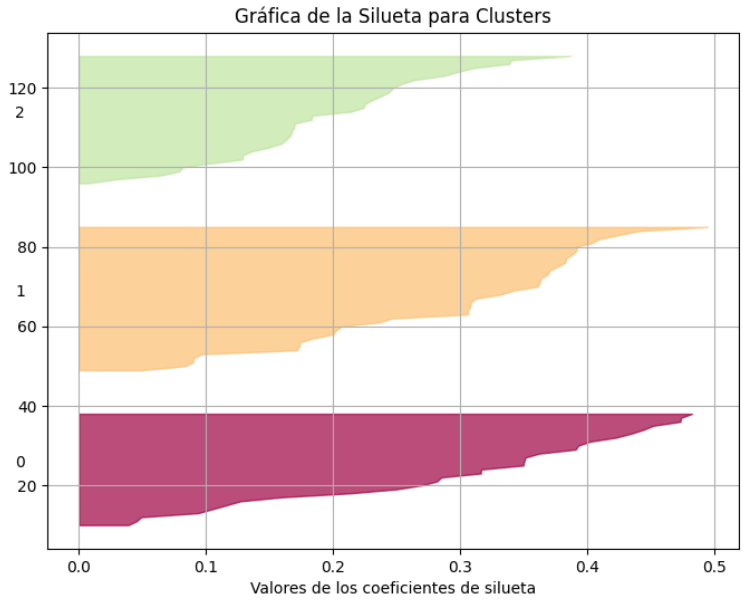


Figura 19: Puntuación de Silueta por número de clústers.

se refleja lo siguiente:

- Los clúster segundo y tercero (naranja y rojo en la figura 19) son los que tienen mayor puntuación de silueta, es decir, sus puntos son los que mejor emparejados están a su propio clúster y lejos del resto.
- Parece que se podría mejorar en el clúster 0 (tiene puntos de silueta más bajos) indicando que podría estar próximo a otro clúster.
- En general, parece que $k = 3$ clústers refleja buena cohesión y separación.

Adicionalmente, se ha calculado el índice de Calinski-Harabasz para distintos número de clústers:

```
from sklearn.metrics import calinski_harabasz_score

calinski_harabasz_scores = []

for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=1234)
    kmeans.fit(penguins_numericas_estandarizadas_sinespecie)
    labels = kmeans.labels_
    score = calinski_harabasz_score(penguins_numericas_estandarizadas_sinespecie, labels)
    calinski_harabasz_scores.append(score)

plt.figure(figsize=(8, 6))
plt.plot(range(2, 11), calinski_harabasz_scores, marker='o', linestyle='-')
plt.title('Índice Calinski-Harabasz por Número de Clústeres')
plt.xlabel('Número de clústeres (k)')
plt.ylabel('Calinski-Harabasz Score')
plt.grid(True)
plt.show()
```

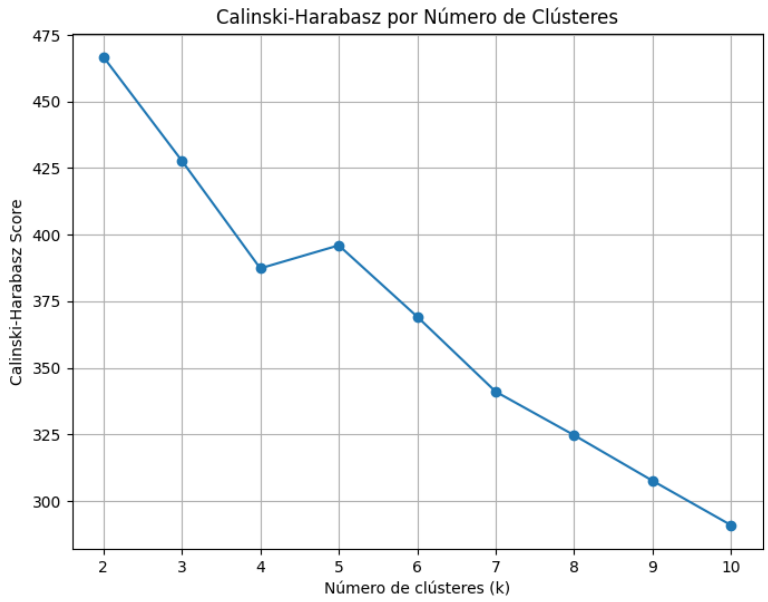


Figura 20: Puntuación de Calinski-Harabasz por número de clústers.

Matemáticamente, se define el índice de Calinski-Harabasz $C(k)$ como

$$C(k) = \frac{E(k)/(k-1)}{D(k)/(N-k)}$$

N es el número de datos y $E(k)$ es la suma de cuadrados ponderados de las distancias entre los centroides y el centroide global, midiendo la variabilidad de los centroides con respecto al centroide global de los datos (es decir, cuánto se

diferencian los clústers entre sí):

$$E(k) = \sum_{i=1}^{k} n_i ||\mu_i - \mu||^2$$

donde k es el número de clústers, n_i el número de puntos que hay en el clúster i, μ_i el centroide del clúster i y μ es el centroide global.

$D(k)$ es la suma de las distancias al cuadrado de todos los puntos hacia el centroide de su respectivo clúster, midiendo la variabilidad de las observaciones dentro de cada clúster:

$$D(k) = \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2$$

Por tanto, $C(k)$ compara la dispersión entre clústers con la dispersión dentro de los clústers. Un índice más grande implica o un mayor valor de $E(k)$ o un menor valor de $D(k)$, reflejando un mejor agrupamiento.

De esta forma, se observa en la figura 20 que para $k = 2$ clústers se tiene el máximo valor del índice, pero como ya se ha mencionado, en el contexto de las especies de pingüinos tiene más sentido considerar $k = 3$ clústers que es el segundo valor más alto para este índice. Estos resultados concuerdan también con lo obtenido por el método de la silueta y el método del codo.

7. Jerárquico vs KMeans

Se han comparado los resultados del agrupamiento jerárquico y KMeans usando las métricas de homogeneidad, plenitud, V-measure y pureza aprovechando que se tiene el conjunto de datos etiquetados por especie desde el principio:

- Homogeneidad: mide el grado en el que los clúster contienen miembros de una clase (especie) determinada.
- Plenitud: mide el grado por el que todos los miembros de una clase (especie) determinada han sido asignados al mismo clúster.
- V-measure: promedio armónico entre la homogeneidad y la plenitud.
- Pureza: Un clúster puro contendrá solo miembros de una sola clase. Para calcularlo, cada grupo se asigna a la clase (especie) que es más frecuente en el grupo, y luego la precisión de esta asignación se mide contando el número de observaciones asignadas correctamente y dividiendo por el número de elementos.

Estas métricas se han implementado con el siguiente código:

```
from sklearn.metrics import homogeneity_score, completeness_score, v_measure_score
from sklearn.preprocessing import LabelEncoder
from sklearn import metrics

le = LabelEncoder()
species_encoded = le.fit_transform(penguins['species'])

homogeneity_cluster3 = homogeneity_score(species_encoded, penguins['Cluster3'])
completeness_cluster3 = completeness_score(species_encoded, penguins['Cluster3'])
v_measure_cluster3 = v_measure_score(species_encoded, penguins['Cluster3'])

print("Métricas para Clustering Jerárquico:")
print(f"Homogeneidad: {homogeneity_cluster3}")
print(f"Plenitud: {completeness_cluster3}")
print(f"V-measure: {v_measure_cluster3}")
```

```python
homogeneity_label = homogeneity_score(species_encoded, penguins['label'])
completeness_label = completeness_score(species_encoded, penguins['label'])
v_measure_label = v_measure_score(species_encoded, penguins['label'])

print("\nMétricas para KMeans:")
print(f"Homogeneidad: {homogeneity_label}")
print(f"Plenitud: {completeness_label}")
print(f"V-measure: {v_measure_label}")

def purity_score(y_true, y_pred):
    contingency_matrix = metrics.cluster.contingency_matrix(y_true, y_pred)
    return np.sum(np.amax(contingency_matrix, axis=0)) / np.sum(contingency_matrix)

pureza_cluster3 = purity_score(species_encoded, penguins['Cluster3'])
pureza_label = purity_score(species_encoded, penguins['label'])

print("\nPureza para Clustering Jerárquico:", pureza_cluster3)
print("Pureza para KMeans:", pureza_label)
```

```
Métricas para Clustering Jerárquico:
Homogeneidad: 0.8864308434649981
Plenitud: 0.911771439415769
V-measure: 0.8989225892804699

Métricas para KMeans:
Homogeneidad: 0.8011796667633956
Plenitud: 0.7789952904660462
V-measure: 0.7899317532789121

Pureza para Clustering Jerárquico: 0.9669669669669669
Pureza para KMeans: 0.918918918918919
```

En base a los resultados se puede concluir que el clustering jerárquico supera al método de KMeans en todas las métricas evaluadas. Cabe destacar que ambos tienen una pureza alta, siendo ligeramente superior la del método de clustering jerárquico. Con respecto al método de KMeans, al obtener valores más bajos de homogeneidad, plenitud y V-measure puede indicar que lo clústers generados contienen más variación de clases y distribución más amplia de una sola clase en diferentes clústers.

No obstante, a pesar de estas diferencias e independientemente del método, todas las observaciones de la especie *Gentoo* se han asignado en un solo clúster (ver figuras 14 y 12): el clúster 1 en Kmeans y el clúster 1 en el jerárquico. Además, se ha demostrado que ambos métodos son eficientes en la agrupación por especie, ya que ambos métodos han generado una pureza bastante alta.

8. Interpretación de los grupos

La siguiente tabla muestra cómo las especies de pingüinos se agrupan en diferentes clústeres según los métodos vistos. De esta manera y por lo analizado en los apartados anteriores, se puede concluir que los métodos jerárquico y KMeans son capaces de agrupar las especies de los pingüinos de forma eficiente basándose en las características físicas recogidas en el dataset.

Cuadro 1: Asignación de Clústers por Especie

Especie	Clúster Jerárquico	Clúster KMeans
Gentoo	1	1
Adelie	3	2
Chinstrap	2	0

Las tendencias significativas incluyen una clara diferenciación de los pingüinos *Gentoo* del resto, teniendo mayor masa corporal, la longitud del pico y las aletas más grandes, y el pico menos profundo. Los pingüinos *Chinstrap* se distinguen gracias a que tienen el pico más significativamente profundo, mientras que los pingüinos *Adelie* se podrían considerar los más pequeños de tamaño general.

9. Resumen, conclusiones, limitaciones...

9.1. Resumen y conclusiones

1. Gracias al PCA se obtuvo una reducción de la dimensionalidad eficaz que capturó la varianza significativa en las dos compomentes principales primeras. Además, se reflejó cómo las diferentes características físicas contribuyen a la diferenciación de los pingüinos, mostrando una separación evidente entre las especies.

2. Ambos métodos de clústering, jerárquico y KMeans han sido eficientes a la hora de identificar grupos que correspondían a las tres especies. El clústering jerárquico mostró ser ligeramente más preciso que el Kmeans.

3. Las variables (atributos físicos) contribuyen a la separación de las especies, diferenciando a los *Gentoo* especialmente del resto.

9.2. Limitaciones y desafíos

1. El primer desafío ha sido determinar el número de clústers. $k = 3$ era el evidente pero también diferenciar por sexo ($k = 6$) se podría pensar que quizás sería interesante también.

2. La forma de dispersión de los datos hace que el método jerárquico o el Kmeans sean adecuados en este análisis. No obstante, si los clústers hubieran tenido una forma no esférica se hubiera necesitado un algoritmo de clústering basado en densidades (como DBSCAN).

3. No se han considerado las variables de isla y sexo. Este hecho podría haber generado resultados sesgados, y podría ser objeto de un análisis futuro.