UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE ESTUDIOS ESTADÍSTICOS

Máster big data, data science & inteligencia artificial

Tarea Minería de datos y modelización predictiva

Ejercicio de ACP + Clustering.

Alumno: Jesús David Navarro Rincón
Profesor: Dr. Pablo Flores Vidal

Madrid, 12 de junio de 2025

INDICE

ILUSTRACIONES

CÓDIGO

TABLAS

1. Introducción y carga del dataset

Comenzaremos la tarea haciendo la carga del dataset “penguins” que se encuentra en la librería *seaborn.* Para ello ejecutamos el siguiente codigo:

```
## CARGAR DATOS DE SEABORN
datos = sns.load_dataset("penguins")
datos.head()

species     island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
0  Adelie  Torgersen            39.1           18.7               181.0
1  Adelie  Torgersen            39.5           17.4               186.0
2  Adelie  Torgersen            40.3           18.0               195.0
3  Adelie  Torgersen             NaN            NaN                 NaN
4  Adelie  Torgersen            36.7           19.3               193.0

   body_mass_g     sex
0       3750.0    Male
1       3800.0  Female
2       3250.0  Female
3          NaN     NaN
4       3450.0  Female
```

Código 1 *Lectura de datos “seaborn penguins"*

Al explorar los datos nos encontramos que hay dos registros con mas del cincuenta por ciento de las variables nulas, por lo que las borraremos del conjunto de datos:

```
# LIMPIEZA DE DATOS MISSING
threshold = datos.shape[1] / 2

# Borrar datos con mas de 50% coloumnas missing.
datos_cleaned = datos.dropna(thresh=threshold)

# Verificar resultado
print(f"Original: {datos.shape}")
print(f"luego de borrado >50% missing: {datos_cleaned.shape}")

Original: (344, 7)
luego de borrado >50% missing: (342, 7)
```

Código 2 *Borrado registro con más del 50% perdidos.*

Adicionalmente, hay nueve registros con el valor de sexo perdido, por lo que vamos a imputarlos utilizando un clasificador:

```
#IMPUTACION VALORES PERDIDOS EN MISSING
missing_sex_rows = datos_cleaned[datos_cleaned['sex'].isnull()]
print(missing_sex_rows)

features = ['species', 'island', 'bill_length_mm', 'bill_depth_mm', 'flipper_length_mm',
'body_mass_g']
df_model = datos_cleaned.dropna(subset=features)

df_known = df_model[df_model['sex'].notnull()]
df_unknown = df_model[df_model['sex'].isnull()]

encoders = {}
df_encoded = df_known.copy()
for col in ['species', 'island', 'sex']:
    le = LabelEncoder()
    df_encoded[col] = le.fit_transform(df_encoded[col])
    encoders[col] = le
X_train = df_encoded[features]
y_train = df_encoded['sex']

clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

df unknown encoded = df unknown.copy()
for col in ['species', 'island']:
    le = encoders[col]
    df_unknown_encoded[col] = le.transform(df_unknown_encoded[col])

X_missing = df_unknown_encoded[features]
predicted_sex = clf.predict(X_missing)

datos_cleaned.loc[df['sex'].isnull(), 'sex'] = encoders['sex'].inverse_transform(predicted_sex)
print(datos_cleaned['sex'].isnull().sum())  # Debe ser  0

0
```

Código 3 *Imputación para valores perdidos de sexo.*

Ahora que tenemos los datos depurados y sin valores perdidos hacemos una descripción de las variables y gráficos para entender mejor los datos:

```
estadisticos = pd.DataFrame({
    'Mínimo': datos_cleaned[variables].min(),
    'Q1': datos_cleaned[variables].quantile(0.25),
    'Mediana': datos_cleaned[variables].median(),
    'Q3': datos cleaned[variables].quantile(0.75),
    'Media': datos_cleaned[variables].mean(),
    'Máximo': datos_cleaned[variables].max(),
    'Desv. típica': datos_cleaned[variables].std(),
    'Varianza': datos_cleaned[variables].var(),
    'Coef. de Variación': (datos_cleaned[variables].std() /
datos cleaned[variables].mean()),
    'Missing': datos_cleaned[variables].isna().sum()
})
display(estadisticos)
```

```
                    Mínimo      Q1  Mediana      Q3        Media      Máximo  \
bill_depth_mm         13.1  15.600    17.30    18.7    17.151170        21.5
bill_length_mm        32.1  39.225    44.45    48.5    43.921930        59.6
body_mass_g         2700.0  3550.000  4050.00  4750.0  4201.754386      6300.0
flipper_length_mm    172.0  190.000   197.00   213.0   200.915205       231.0
island              Biscoe      NaN      NaN     NaN          NaN  Torgersen
sex                 Female      NaN      NaN     NaN          NaN       Male
species             Adelie      NaN      NaN     NaN          NaN     Gentoo

                    Desv. típica       Varianza  Coef. de Variación  Missing
bill_depth_mm           1.974793       3.899808            0.115140        0
bill_length_mm          5.459584      29.807054            0.124302        0
body_mass_g           801.954536  643131.077327            0.190862        0
flipper_length_mm      14.061714     197.731792            0.069988        0
island                       NaN            NaN                 NaN        0
sex                          NaN            NaN                 NaN        0
species                      NaN            NaN                 NaN        0
```

Código 4 *Descriptivo de las variables del conjunto de datos.*

Analizamos la variable “sex” que es binaria como se comportan las características de los pingüinos en este grafico de cajas:

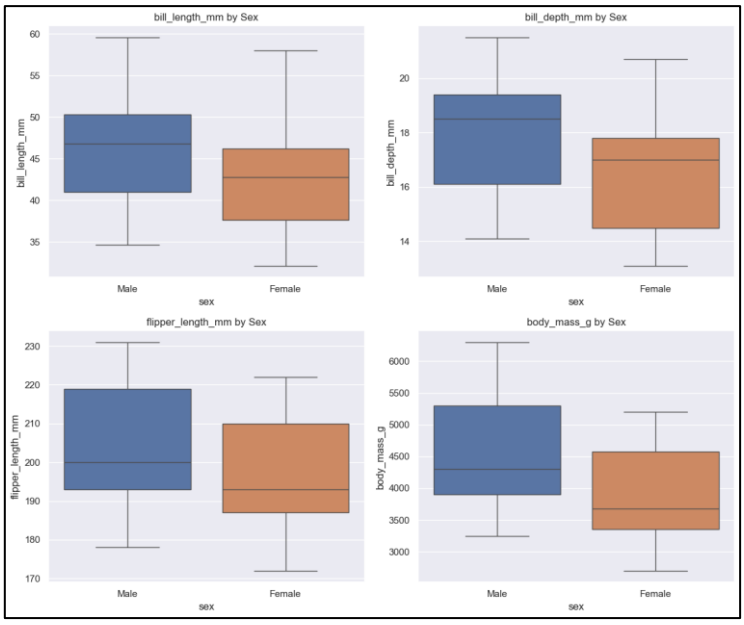

Ilustración 1 *Grafico de cajas de las variables continuas versus el sexo.*

De acuerdo con este grafico se puede observar una diferencia considerable en las características físicas de los pingüinos machos con respecto a las hembras, por tanto tomaremos en cuenta esta variable para el **análisis de componentes principales** y el *clustering*, para ello haremos el "One-Hot encoding" de la variable:

```
datos_encoded = pd.get_dummies(datos_cleaned, columns=['sex'],
drop_first=True)
datos_encoded['sex_Male'] = datos_encoded['sex_Male'].astype(int)
datos_encoded.head()

bill_length_mm  bill_depth_mm  flipper_length_mm  body_mass_g  sex_Male
0              39.1           18.7               181.0        3750.0          1
1              39.5           17.4               186.0        3800.0          0
2              40.3           18.0               195.0        3250.0          0
4              36.7           19.3               193.0        3450.0          0
5              39.3           20.6               190.0        3650.0          1
```

Código 5 *Borrado de las variables categóricas "island & species", y "one hot-encoding" de variable "sex".*

En la matriz podemos observar que hay una alta correlación entre la longitud de la aleta y la masa corporal del pingüino, por lo que el PCA nos ayudará bastante a reducir la multicolinealidad de estas variables. Adicionalmente, las variables de profundidad del pico y largo de la aleta están altamente correlacionadas de manera negativa.

En un grafico de dispersión podemos ver esto de una manera más clara:

Ilustración 2 *Grafico de dispersión variables altamente correlacionadas.*

2. Análisis de PCA

Para iniciar con el análisis de componentes principales (PCA), vamos a calcular primero la matriz de correlaciones y dibujarla en un gráfico:

```
R = datos_encoded[variables_num].corr()
print(R)
plt.figure(figsize=(10, 8))
sns.heatmap(R, annot=True, cmap='coolwarm', fmt='.2f',
linewidths=0.5)
```

Código 6 *Calculo matriz de correlaciones más el grafico de mapa de calor.*

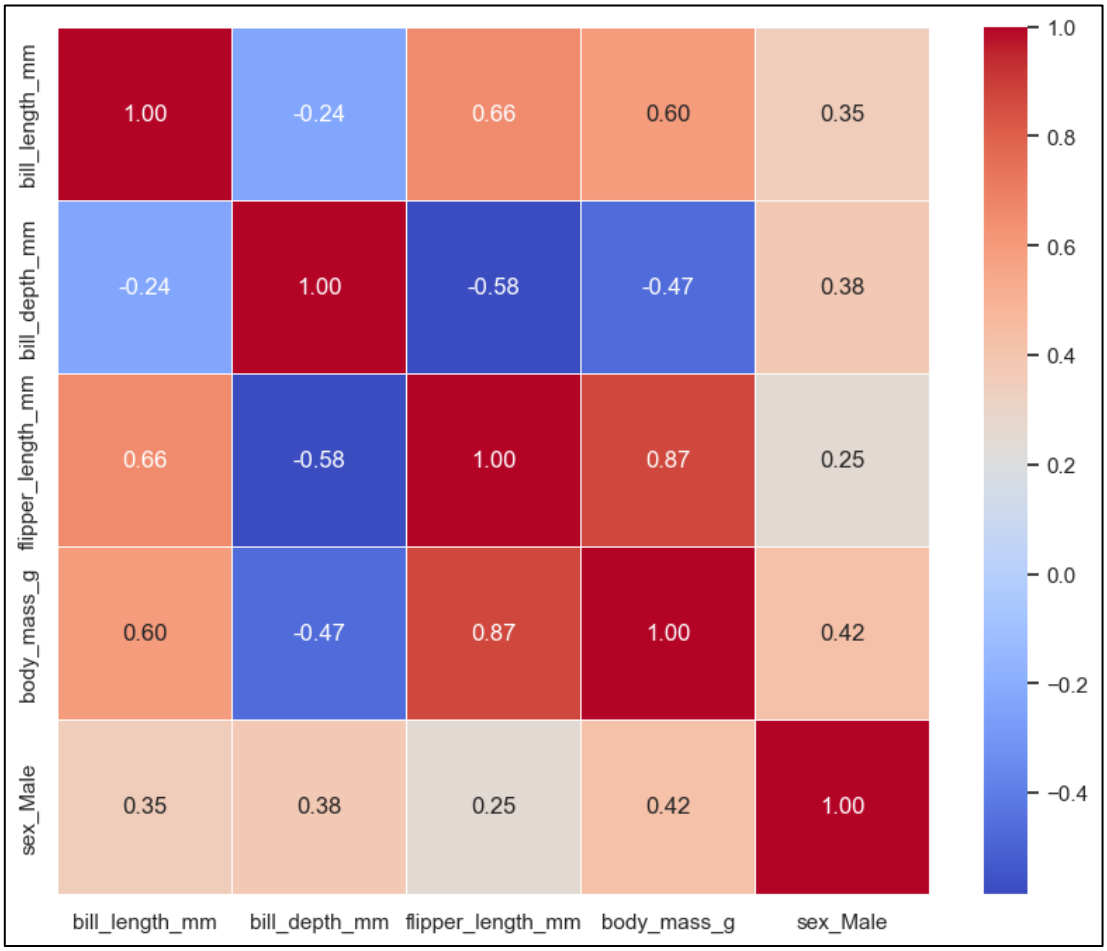

Ilustración 3 *Matriz de correlación variables continuas.*

En la matriz podemos observar que las variables de las características físicas mas correlacionadas de forma inversa son la profundidad del pico con el peso y el largo de la aleta. También, la profundidad del pico está ligeramente correlacionada de manera inversa con el largo del pico, por lo que podríamos decir que entre mas profundo es el pico, más pequeñas son las otras características físicas del pingüino.

Continuando con el PCA, hacemos la estandarización de los datos y calculamos de manera inicial cinco componentes (igual al numero de variables) para conocer los autovalores y la variabilidad explicada:

```
#Arrancamos con PCA
#Definimos X como el dataset donde ejecutaremos el PCA. (Variables
continuas)
X = datos_encoded[variables_num]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Crea una instancia de Análisis de Componentes Principales (ACP):
# - Utilizamos PCA(n_components=7) para crear un objeto PCA que realizará
un análisis de componentes principales.
# - Establecemos n_components en 7 para retener el maximo de las
componentes principales (maximo= numero de variables).
pca = PCA(n_components=5)

# Aplicar el Análisis de Componentes Principales (ACP) a los datos
estandarizados:
# - Usamos pca.fit(notas_std) para ajustar el modelo de ACP a los datos
estandarizados.
fit = pca.fit(X_scaled)

# Obtener los autovalores asociados a cada componente principal.
autovalores = fit.explained_variance_
autovalores

# Obtener la varianza explicada por cada componente principal como un
porcentaje de la varianza total.
var_explicada = fit.explained_variance_ratio_
# Por ejemplo podemos ver que la varianza explicada por tres CP es:
var_explicada_2 = np.sum(var_explicada[:2])
var_explicada_2

# Calcular la varianza explicada acumulada a medida que se agregan cada
componente principal.
var_acumulada = np.cumsum(var_explicada)
var_acumulada

# Crear un DataFrame de pandas con los datos anteriores y establecer
índice.
data = {'Autovalores': autovalores, 'Variabilidad Explicada':
var_explicada, 'Variabilidad Acumulada': var_acumulada}
tabla = pd.DataFrame(data, index=['Componente {}'.format(i) for i in
range(1, fit.n_components_+1)])

# Imprimir la tabla
print(tabla)
```

```
Autovalores  Variabilidad Explicada  Variabilidad Acumulada
Componente 1     2.848496                0.568034               0.568034
Componente 2     1.417004                0.282572               0.850606
Componente 3     0.477807                0.095282               0.945887
Componente 4     0.170808                0.034062               0.979949
Componente 5     0.100548                0.020051               1.000000
```

Código 7 *PCA inicial*

La tabla resumen del PCA inicial:

Componente	Autovalores	Variabilidad Explicada	Variabilidad Acumulada
Componente 1	2,848496	0,568034	0,568034
Componente 2	1,417004	0,282572	0,850606
Componente 3	0,477807	0,095282	0,945887
Componente 4	0,170808	0,034062	0,979949
Componente 5	0,100548	0,020051	1,000000

Tabla 1 *PCA inicial con cinco componentes.*

Antes de comentar los hallazgos hacemos el grafico de codo para los componentes:

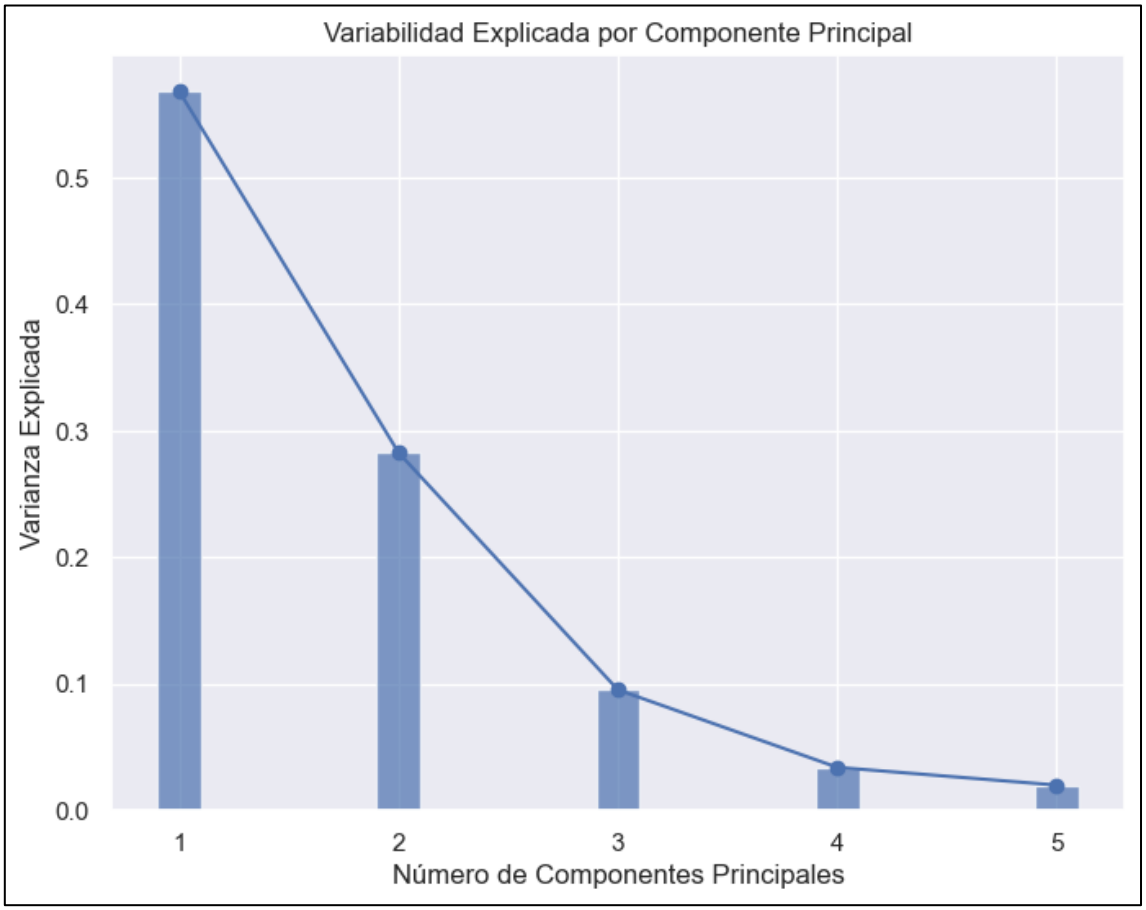

Ilustración 4 *Varianza explicada para los componentes PCA iniciales.*

Este análisis inicial nos muestra que el valor mas adecuado de componentes es el tres (3), esto se debe porque el salto entre dos y tres componentes es casi de un 10% en variabilidad explicada, lo cual podemos observar tanto en la tabla como en el gráfico. Con tres componentes lograremos explicar el 94,5% de la variabilidad de los datos, por lo que es un excelente número para reducir la dimensionalidad de los datos y no perder información importante.

Con la definición de los tres (3) componentes principales pasamos a contestar a las preguntas realizadas en el apartado con la ayuda de la tabla de autovalores, y del gráfico a continuación, podemos contestar la primera pregunta sobre que representa cada componente con respecto a las características físicas de los pingüinos:

```
# Renombrar columnas con sufijo _z para indicar estandarización
variables_num_z = [col + '_z' for col in variables_num]
X_scaled_df = pd.DataFrame(X_scaled, columns=variables_num_z,
index=datos_clean.index)

pca2 = PCA(n_components=3)
fit2 = pca2.fit(X_scaled_df)
resultados_pca = pd.DataFrame(
    fit2.transform(X_scaled_df),
    columns=['Componente {}'.format(i) for i in range(1, fit2.n_components_
+ 1)],
    index=datos_clean.index
)
datos_z = pd.concat([datos_clean[['species', 'island']], X_scaled_df],
axis=1)
datos_z_cp = pd.concat([datos_z, resultados_pca], axis=1)


# Seleccionamos solo las variables numéricas estandarizadas (sin
categóricas)
variables_numericas_estandarizadas = X_scaled_df  # Ya estandarizadas
componentes_pca = resultados_pca  # Ya generadas con PCA

# Concatenar para calcular correlación
df_corr = pd.concat([variables_numericas_estandarizadas, componentes_pca],
axis=1)

# Calcular la matriz de correlación
correlacion = df_corr.corr()

variables_num_z = [var + '_z' for var in variables_num]
# Extraer submatriz: correlación de variables vs componentes
correlaciones_var_comp = correlacion.loc[variables_num_z,
componentes_pca.columns]
##ELEVAMOS AL CUADRADO LAS CORRELACIONES Y MOSTRAMOS
cos2 = correlaciones_var_comp**2
display(cos2)
```

	Componente 1	Componente 2	Componente 3
bill_length_mm_z	0.612064	0.041112	0.341364
bill_depth_mm_z	0.317886	0.594401	0.012793
flipper_length_mm_z	0.901364	0.012928	0.005526
body_mass_g_z	0.869843	0.003986	0.065169
sex_Male_z	0.139009	0.760434	0.051557

Código 8 *Nuevo análisis de PCA = 3 componentes y autovalores.*

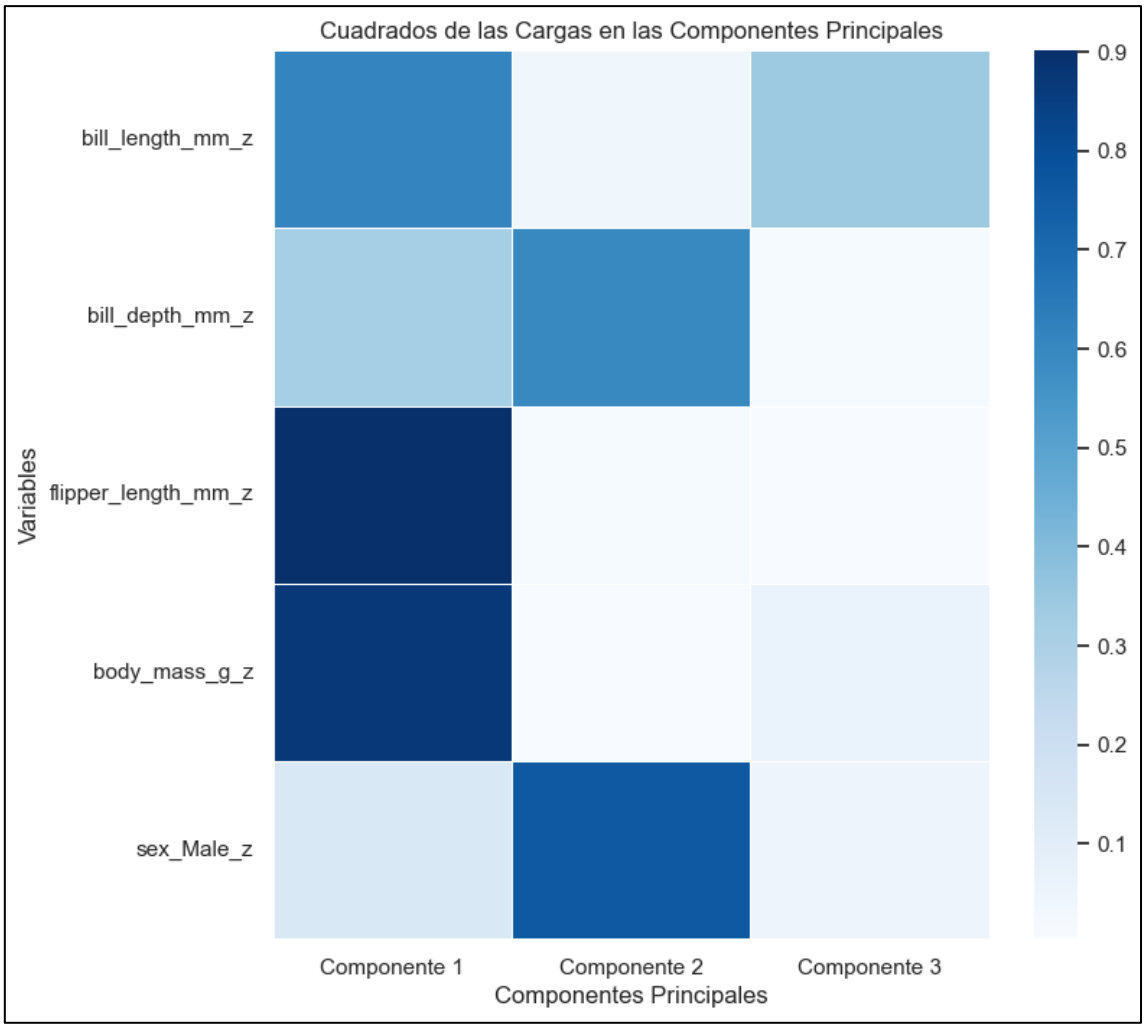

Ilustración 5 *Mapa de calor de componentes principales y variables estandarizadas.*

Con estos datos podemos describir que representa cada componente con respecto a las características físicas de los pingüinos:

- **PCA 1:**
  - Este componente parece representar una dimensión de tamaño general del pingüino.
  - Las variables físicas como longitud del pico, aleta y masa corporal tienen los pesos mas altos.
- **PCA 2:**
  - Este componente diferencia especialmente el sexo y la profundidad del pico que podrían estar asociados a algunas de las especies.
- **PCA 3:**
  - Este componente parece enfocarse en la longitud del pico en contraste de la masa y el sexo.

Para responder al apartado dos vamos a observar los gráficos de dispersión para cada componente y las especies:

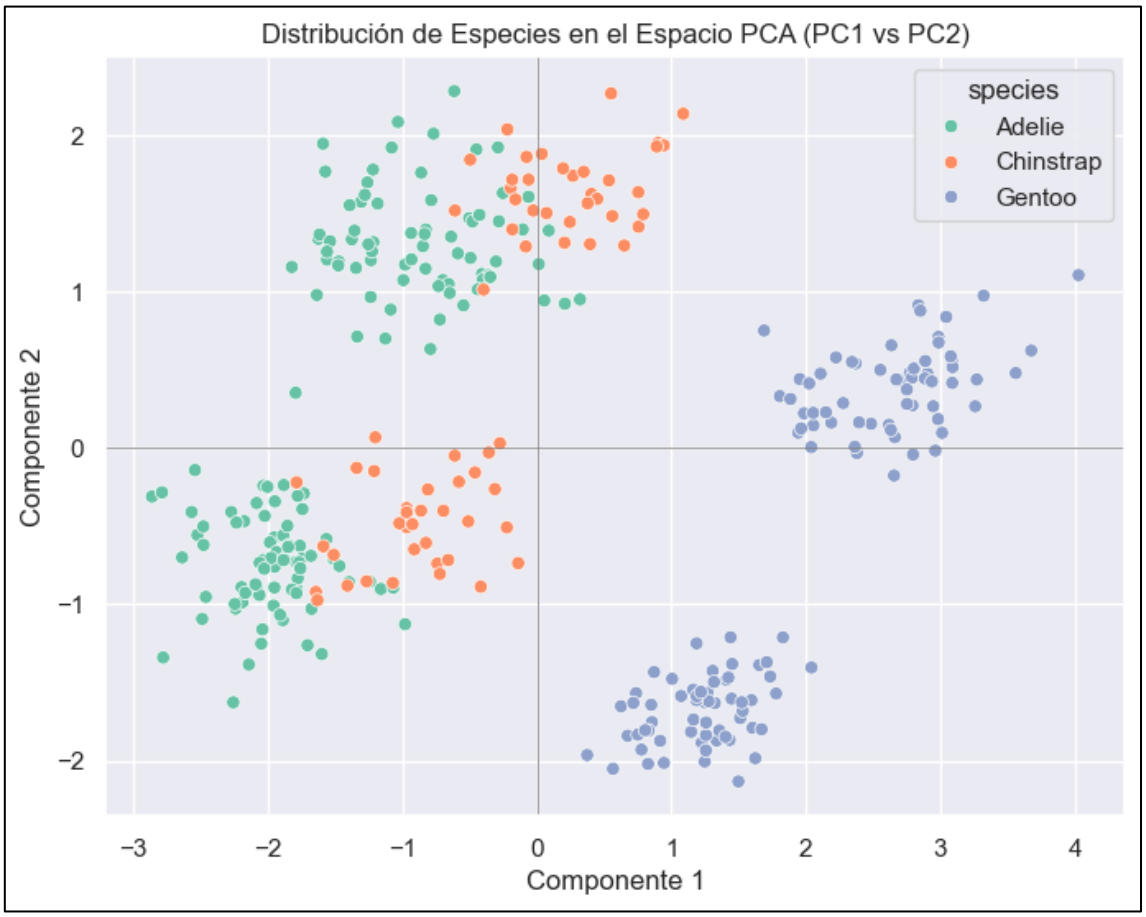

Ilustración 6 *Distribución de especies en el espacio PCA (PC1 vs PC2)*

Este grafico muestra claramente que la especie **Gentoo** se difirencia bastante por el PCA 1 que tiene que ver con el tamaño corporal de los pinguinos, por lo que podríamos decir que los Gentoo son mas grandes que las otras dos especies. El **PCA 2** al estar enfocado mas en el sexo, parece lograr una separación vertical entre los ejemplares femeninos y masculinos de cada especie.

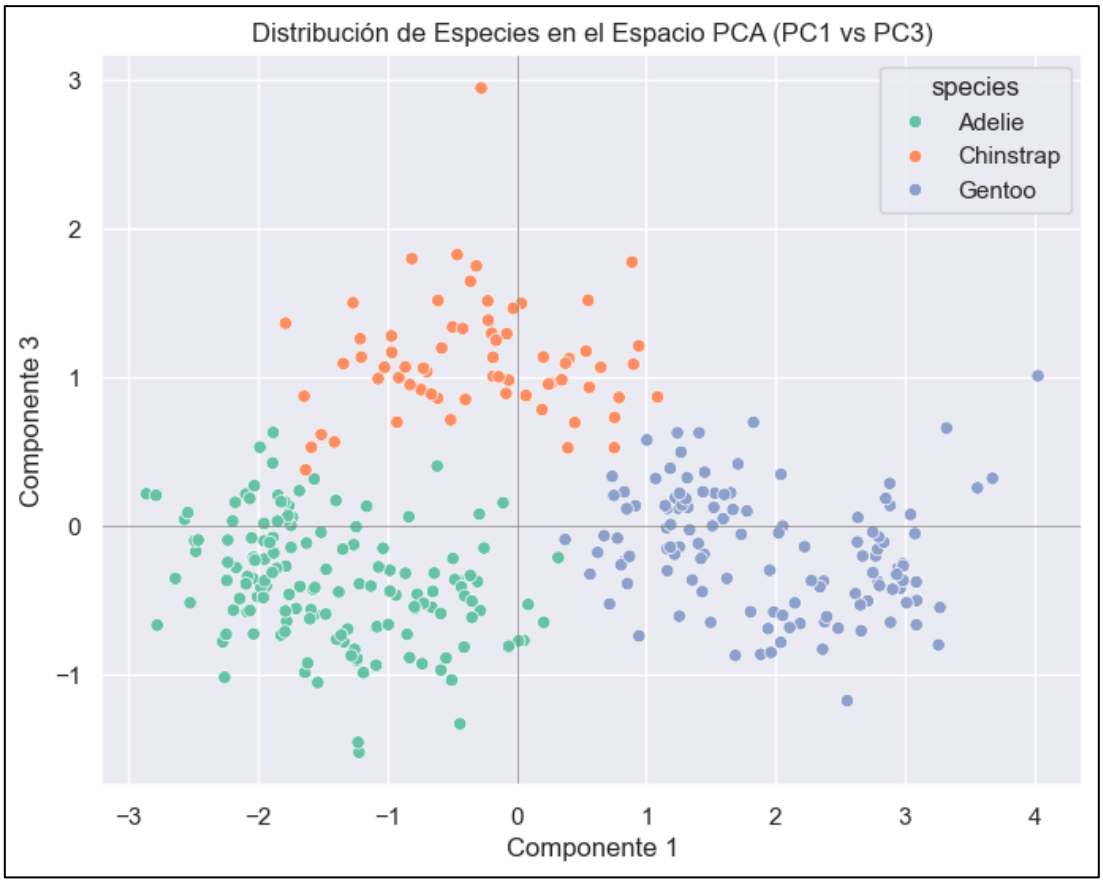

Ilustración 7 *Distribución de especies en el espacio PCA (PC1 vs PC3).*

Acá continuamos observando una clara separación de la especie **Gentoo** en el PCA1. Interesante que el **PCA 3** que se enfoca mas en la longitud del pico ahora si logra separar más la especie **Chinstrap** de las otras dos especies, por lo que podríamos decir que los pingüinos **Chinstrap** se caracterizan por tener picos más largos que las otras especies.

Finalmente, con respecto a la pregunta del índice para valorar de forma conjunta las características físicas de los pingüinos, la sugerencia será utilizar el **PCA 1** que es una combinación lineal de las variables físicas estandarizadas. El valor del índice podríamos obtenerlo como la media del valor del PCA 1 para cada una de las especies, por ejemplo:

```
datos_z_cp['indice_fisico'] = datos_z_cp['Componente 1']
indice_promedio =
datos_z_cp.groupby('species')['indice_fisico'].mean()
print(indice_promedio)

species
Adelie      -1.427559
Chinstrap   -0.322691
Gentoo       1.930931
Name: indice_fisico, dtype: float64
```

Código 9 *índice para valorar las características físicas de las especies.*

En esa salida podemos observar que el valor para la especie **Adelie** sería -1.42 y para la especie **Chinstrap** -0.32, los que nos dice que estas especies son mucho más pequeñas físicamente que la especie **Gentoo**.

3. Clustering

Para iniciar con la parte de *clustering* vamos a hacer un mapa de calor con agrupamiento inicial:

```
# Configuración de estilo
sns.set(font_scale=0.9)  # Ajusta el tamaño del texto
sns.set_style("white")
g = sns.clustermap(
    datos_z_cp[variables_num_z2],
    cmap='coolwarm',
    annot=False,            # Evita sobrecargar con números
    figsize=(10, 7),
    row_cluster=True,
    col_cluster=True,
    yticklabels=False       # Quita etiquetas de las filas si hay
muchas
)
g.fig.suptitle("Mapa de calor de características físicas
estandarizadas", fontsize=14, y=1.02)
g.ax_heatmap.set_xlabel("Variables físicas estandarizadas",
fontsize=12)
g.ax_heatmap.set_ylabel("Individuos (Pingüinos)", fontsize=12)
plt.show()
```

Código 10 *Código para generar mapa de calor inicial de clustering.*

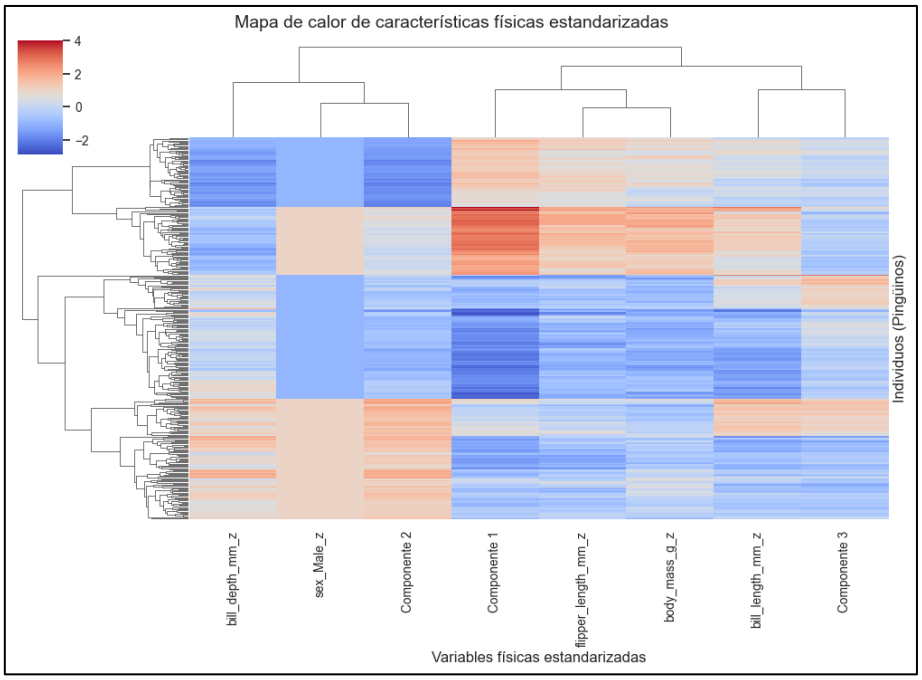

Ilustración 8 *mapa de calor inicial de clustering.*

Esta información inicial parece indicar que un buen numero de grupos sería tres, lo cual coincide con las especies que tenemos en el conjunto de datos. Ahora vamos a hacer el agrupamiento jerárquico y pintar el dendrograma para verificar la selección de grupos:

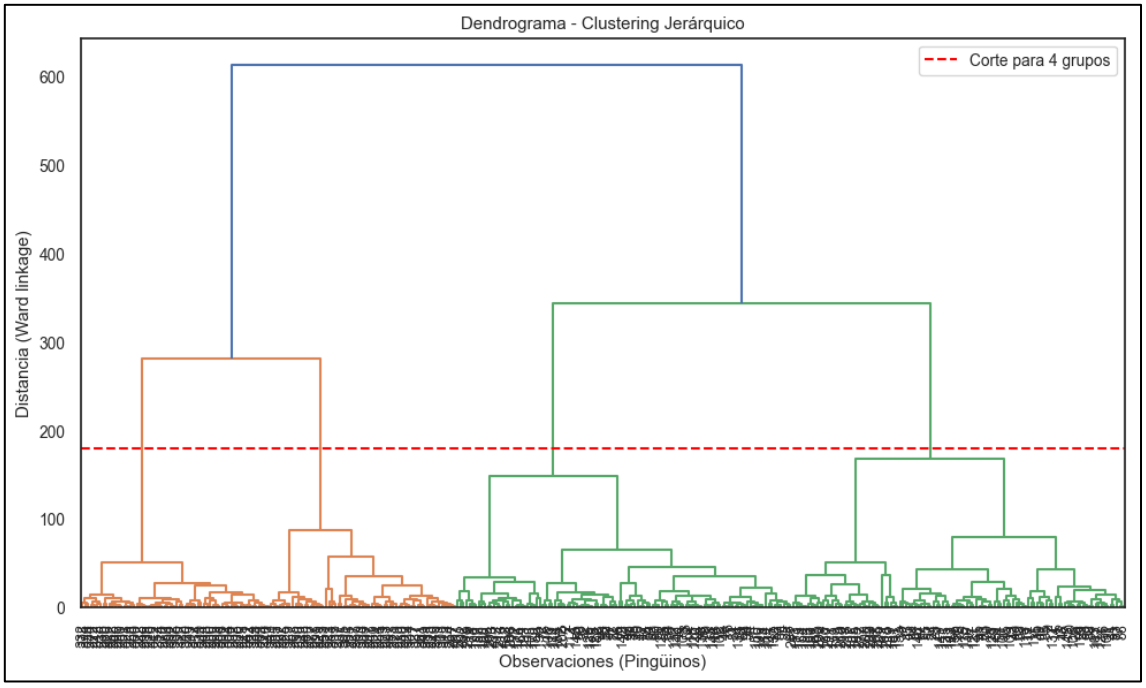

Ilustración 9 *Dendrograma jerárquico - marca para cuatro grupos.*

Con el grafico anterior vemos que es mejor hacer cuatro grupos para minimizar la distancia entre ellos, ya que irnos más arriba a tres grupos haría que la distancia entre grupos sea muy alta, y por debajo de la marca de 180 pasaríamos a tener ya 6 grupos lo cual es demasiados grupos para nuestro conjunto de datos, por lo que un numero apropiado de grupos sería cuatro (4), sin embargo, esto lo verificaremos a continuación con el algoritmo de K-means y el "WCSS".

En el siguiente bloque de código vamos a generar los diferentes grupos con el algoritmo de K-Means desde 1 a 11 grupos y luego evaluar con la métrica de WCSS y el método del codo cual sería el número de grupo más apropiado:

```
wcss = []

for k in range(1, 11):  # You can choose a different range of K values
    kmeans = KMeans(n_clusters=k, random_state=0)
    kmeans.fit(datos_z_cp[variables_num_z2])
    wcss.append(kmeans.inertia_)  # Inertia is the WCSS value
plt.figure(figsize=(8, 6))
plt.plot(range(1, 11), wcss, marker='o', linestyle='-', color='b')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('WCSS')
plt.grid(True)
plt.show()
```

Código 11 *Generación de K-means y WCSS*

El grafico generado:

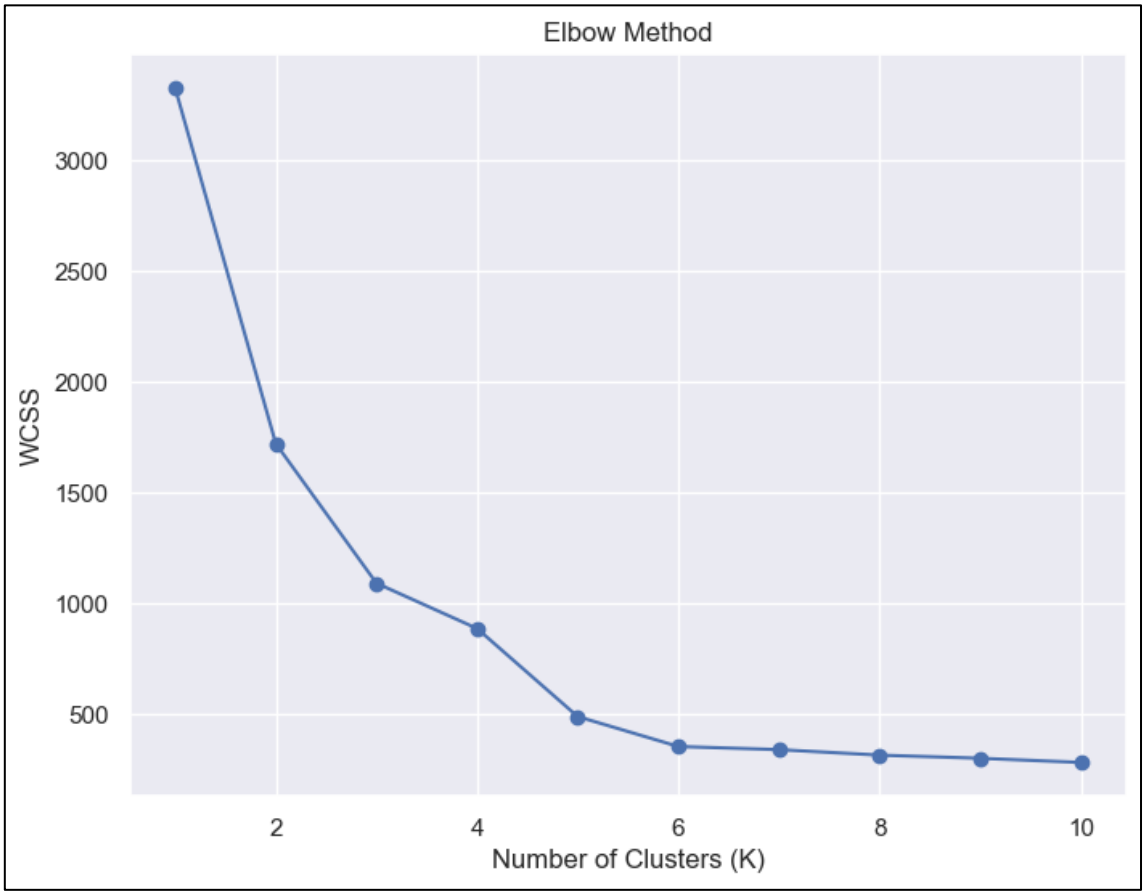

Ilustración 10 *Grafico del codo K-Means*

El grafico de codo anterior indica que un buen número de grupos sería 5 en lugar de 4 como lo teníamos anteriormente en el clúster jerárquico y dendrograma. Colocando la línea de corte 160 puntos de distancia se obtendría 5 grupos en el método jerárquico:

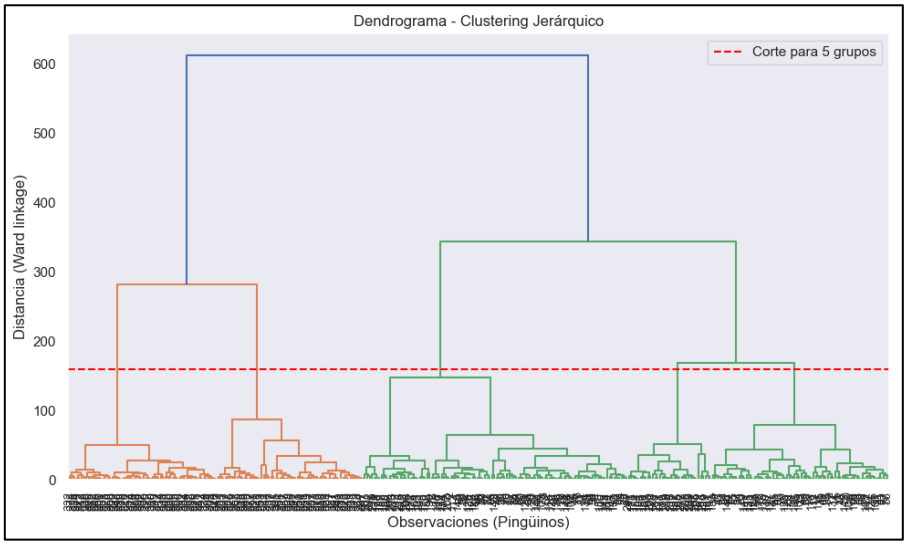

Ilustración 11 *Dendrograma para 5 grupos.*

Con base en estos resultados vamos a calcular el puntaje de silueta tanto para cuatro y cinco grupos del método K-Means y decidimos cual es mejor luego de esta prueba:

```
silhouette_scores = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=0)
    kmeans.fit(datos_z_cp[variables_num_z2])
    labels = kmeans.labels_
    silhouette_avg = silhouette_score(datos_z_cp[variables_num_z2],
labels)
    silhouette_scores.append(silhouette_avg)

plt.figure(figsize=(8, 6))
plt.plot(range(2, 11), silhouette_scores, marker='o', linestyle='-',
color='b')
plt.title('Silhouette Method')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Silhouette Score')
plt.grid(True)
plt.show()
```

Código 12 *Calculo valores método de silueta.*

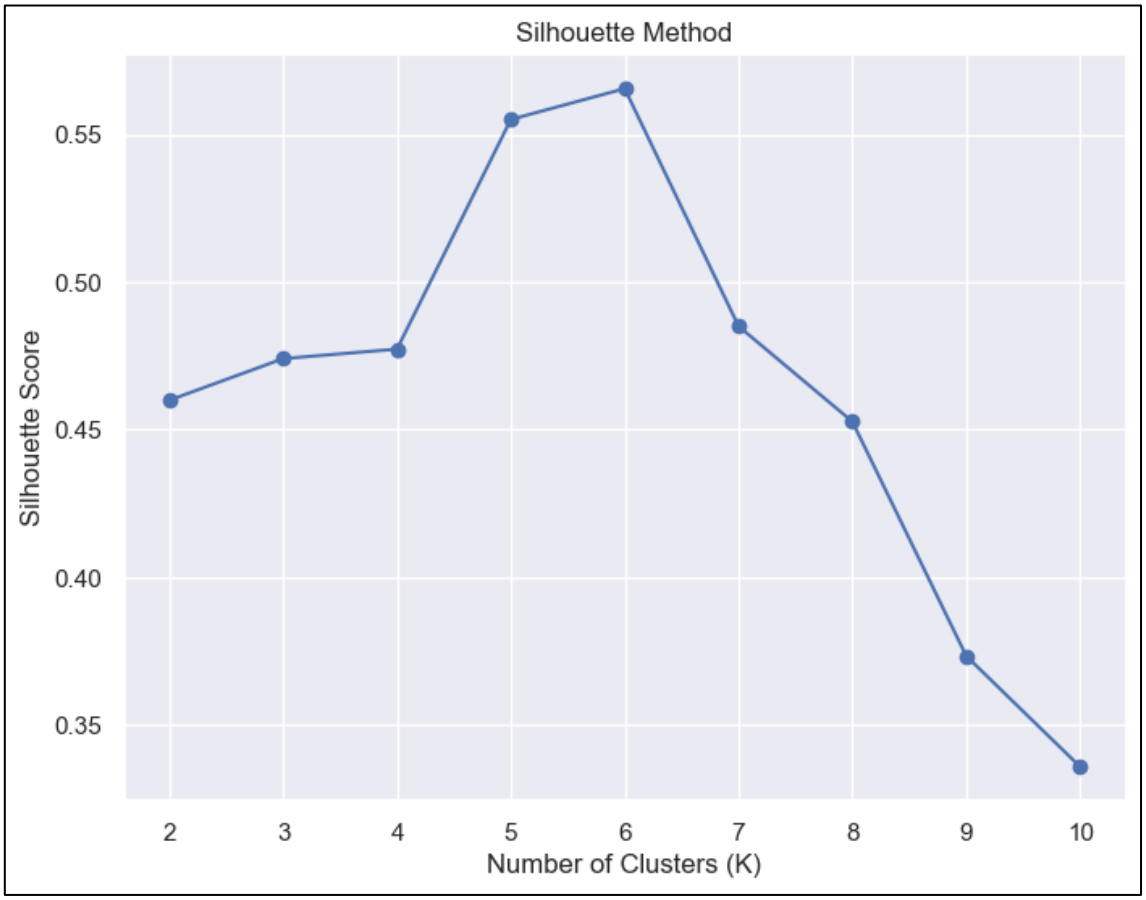

Ilustración 12 *Grafico score de silueta para K-Means.*

El resultado obtenido nos indica claramente que se logra mejor agrupamiento con **cinco (5) grupos.** El cambio entre 5 y 6 grupos es muy bajo en el score, por lo que optamos por el menor número de grupos.

Una vez ya hemos decidido utilizar cinco grupos, hacemos la asignación de estos grupos tanto por el método jerárquico como K-Means y los representamos en un gráfico de dispersión para los componentes PCA1 y PCA2, y poder comentar las diferencias de asignación:

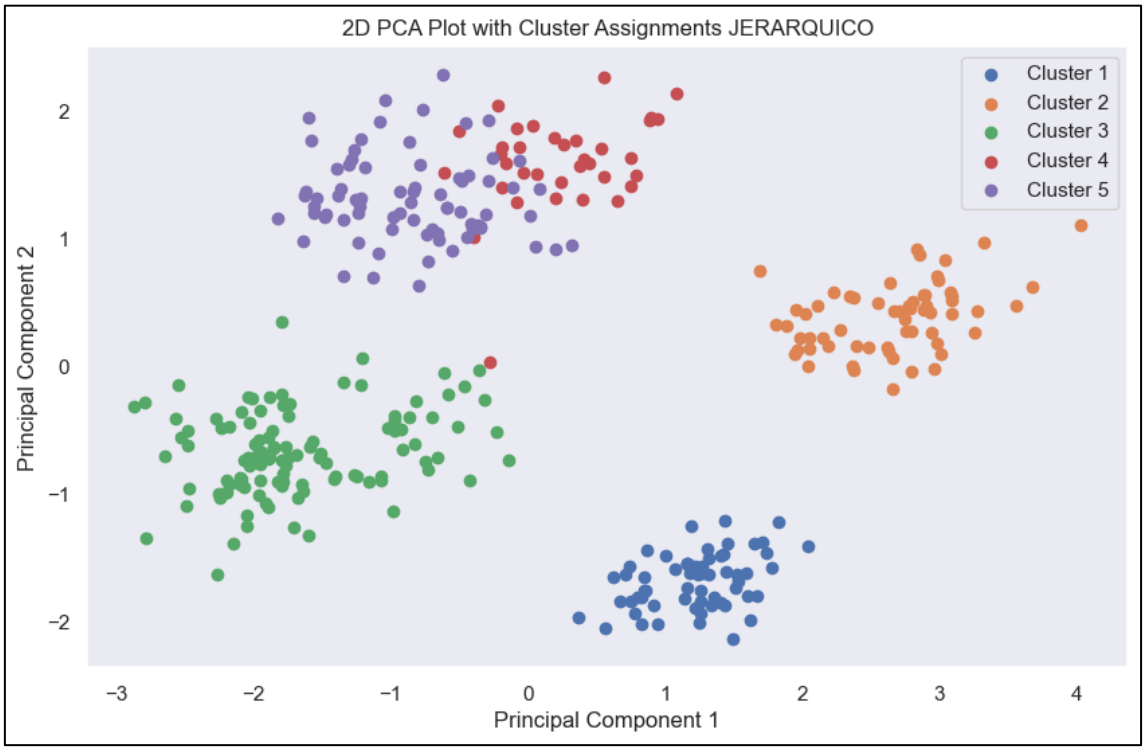

Ilustración 13 *Diagrama asignación clúster jerárquico.*

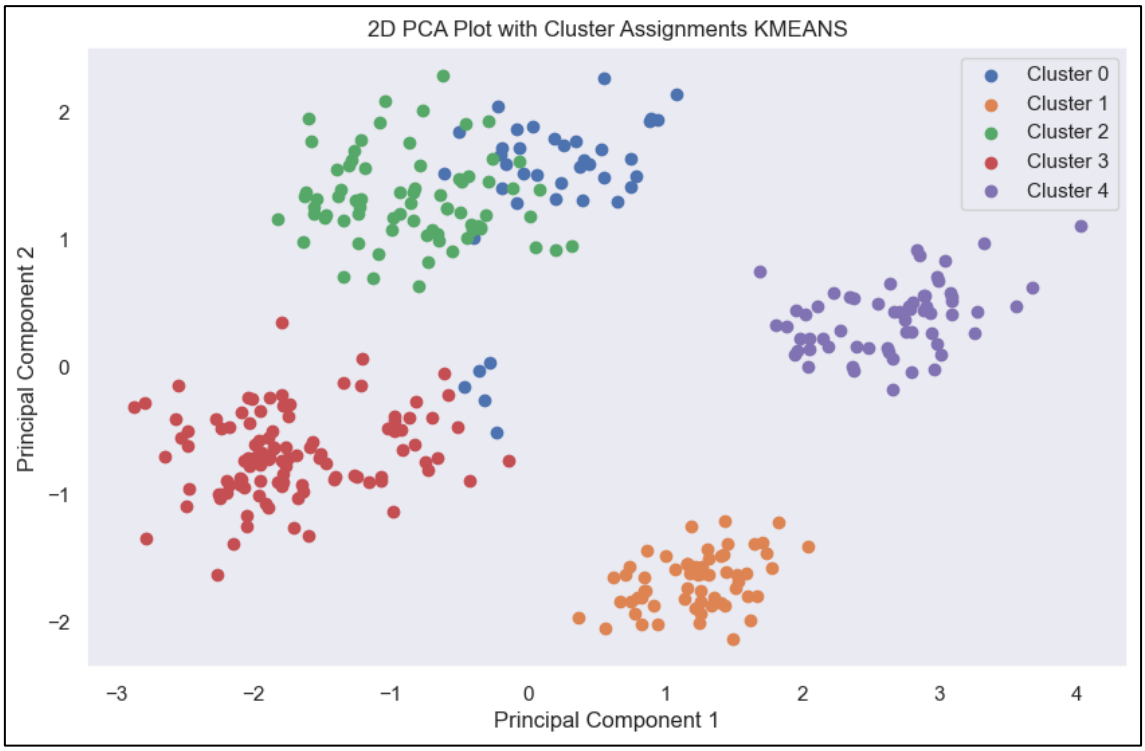

Ilustración 14 *Diagrama asignación clúster Kmeans.*

Los gráficos muestran que ambos métodos dan como resultados agrupaciones similares, hay unas pequeñas diferencias en los valores centrales de al menos 4 pingüinos. Para interpretar mejor los resultados por especie y sexo tenemos esta tabla resumen de los grupos:

Cluster jerárquico	Especie dominante	Sexo dominante	Interpretación
1	Gentoo (100%)	Female (100%)	Pingüinos **Gentoo hembras**
2	Gentoo (100%)	Male (100%)	Pingüinos **Gentoo machos**
3	Adelie (70%), Chinstrap (30%)	Female (100%)	Mayormente **Adelie hembras**, con algunas Chinstrap hembras
4	Chinstrap (100%)	Male (97%)	**Chinstrap machos.** Con 3% de hembras Chinstrap
5	Adelie (100%)	Male (100%)	**Adelie machos**
Cluster K-Means	**Especie dominante**	**Sexo dominante**	**Interpretación**
0	Chinstrap (100%)	Male (87%)	Principalmente **Chinstrap machos**
1	Gentoo (100%)	Female (100%)	**Gentoo hembras**
2	Adelie (100%)	Male (100%)	**Adelie machos**
3	Adelie (72%), Chinstrap (27%)	Female (100%)	**Adelie hembras**, con algunas Chinstrap hembras
4	Gentoo (100%)	Male (100%)	**Gentoo machos**

Tabla 2 *Resumen de clústeres jerárquico y Kmeans, grupos de sexo y especie.*

Cluster Jerárquico	Equivalente en K-Means
1 (Gentoo hembra)	1
2 (Gentoo macho)	4
3 (Adelie + Chinstrap hembras)	3
4 (Chinstrap macho)	0
5 (Adelie macho)	2

Tabla 3 *Equivalencia etiqueta de Clúster jerárquico y Kmeans.*

Los colores resaltados en la tabla 2 muestran las diferencias entre los grupos asignados, las cuales son relativamente pequeñas. A su vez, observando las asignaciones de especie y sexo, se puede ver que las estrategias de clústeres fueron muy efectivas para separar la especie Gentoo (macho y hembra), y los machos de la especie Adelie y Chinstrap, sin embargo, para las hembras de estas especies lo que hizo fue agruparlas en un grupo, y algunos machos cayeron en estos grupos, puede ser que físicamente sean pequeños y por ello fueron considerados hembras.

En resumen, las técnicas de clústeres lograron separar los grupos de manera efectiva por especie y sexo, las diferencias entre el clúster jerárquico y Kmeans no fueron tan grandes, por lo que podríamos considerarlos similares en su comportamiento en este caso. Ahora bien, el clúster jerárquico y el dendrograma tienen mucha importancia, ya que permiten tener una idea de la cantidad de grupos que deben ser asignados para el conjunto de datos, para luego confirmarlo con las estimaciones de Kmeans. La puntuación de silueta resultó ser fundamental para dirimir si fueran cuatro o cinco grupos para el agrupamiento, ya que la técnica del codo no era clara si incluso podían considerarse hasta 6 grupos, lo cual se vio descartado en la técnica de silueta al ver el poco incremento de la puntuación con seis grupos.