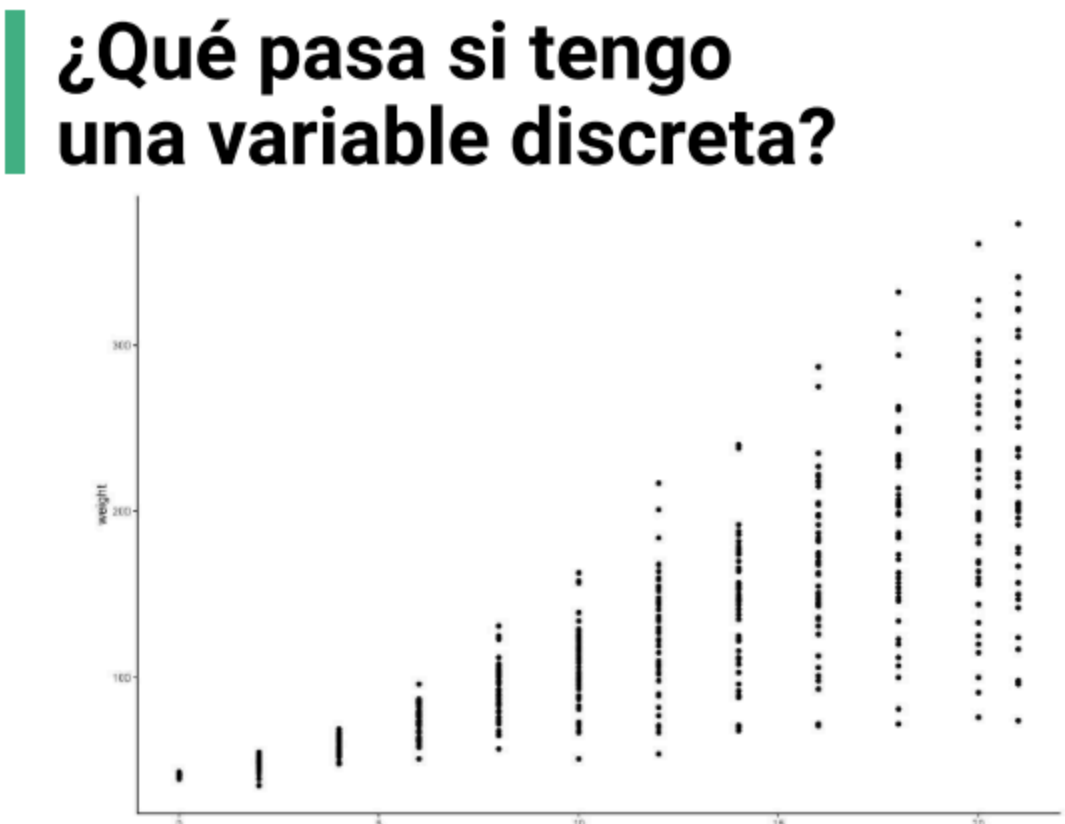
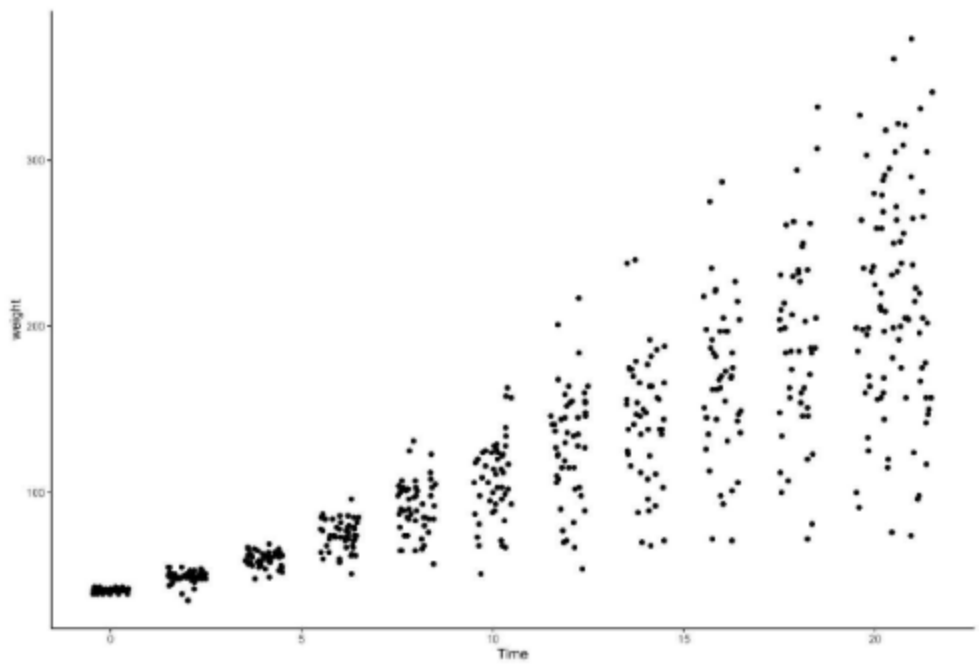
Estableciendo relaciones: gráficos de violín y boxplot

Estableciendo relaciones

Los datos reales siempre tiene un comportamiento en el que existirán huecos, o lineas rectas. Esto pasa por tener una variable discreta.

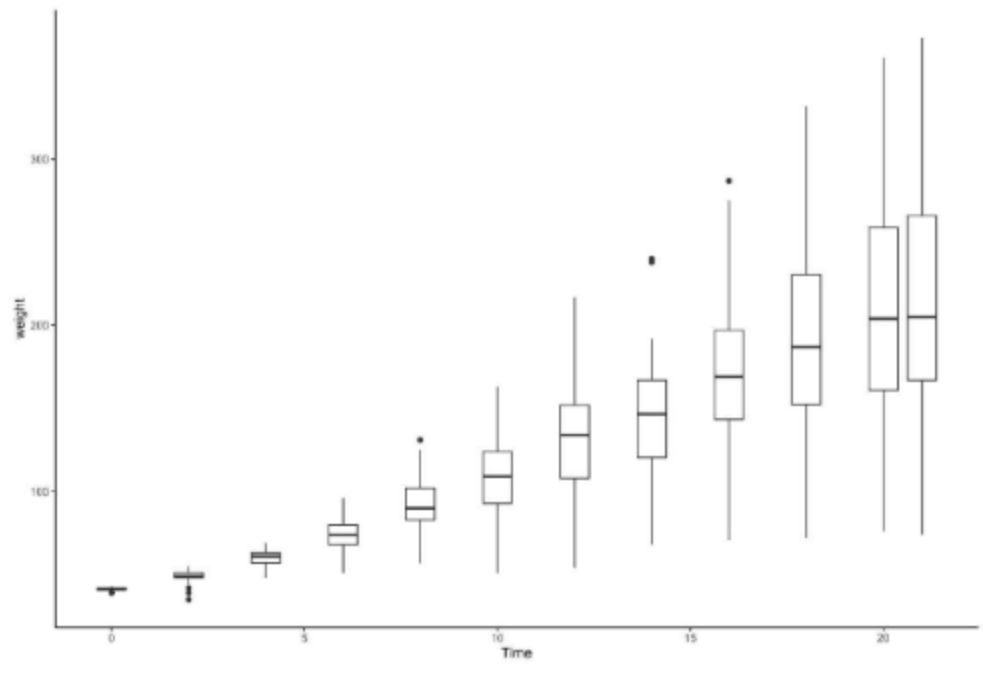


Otras formas en las que podemos progresar con este tipo de gráficas y darles una interpretación útil, porque en este caso los puntos se acumularán en una sola linea; es agregar ruido aleatorio que nos ayude a separar los puntos y visualizar de una mejor forma.



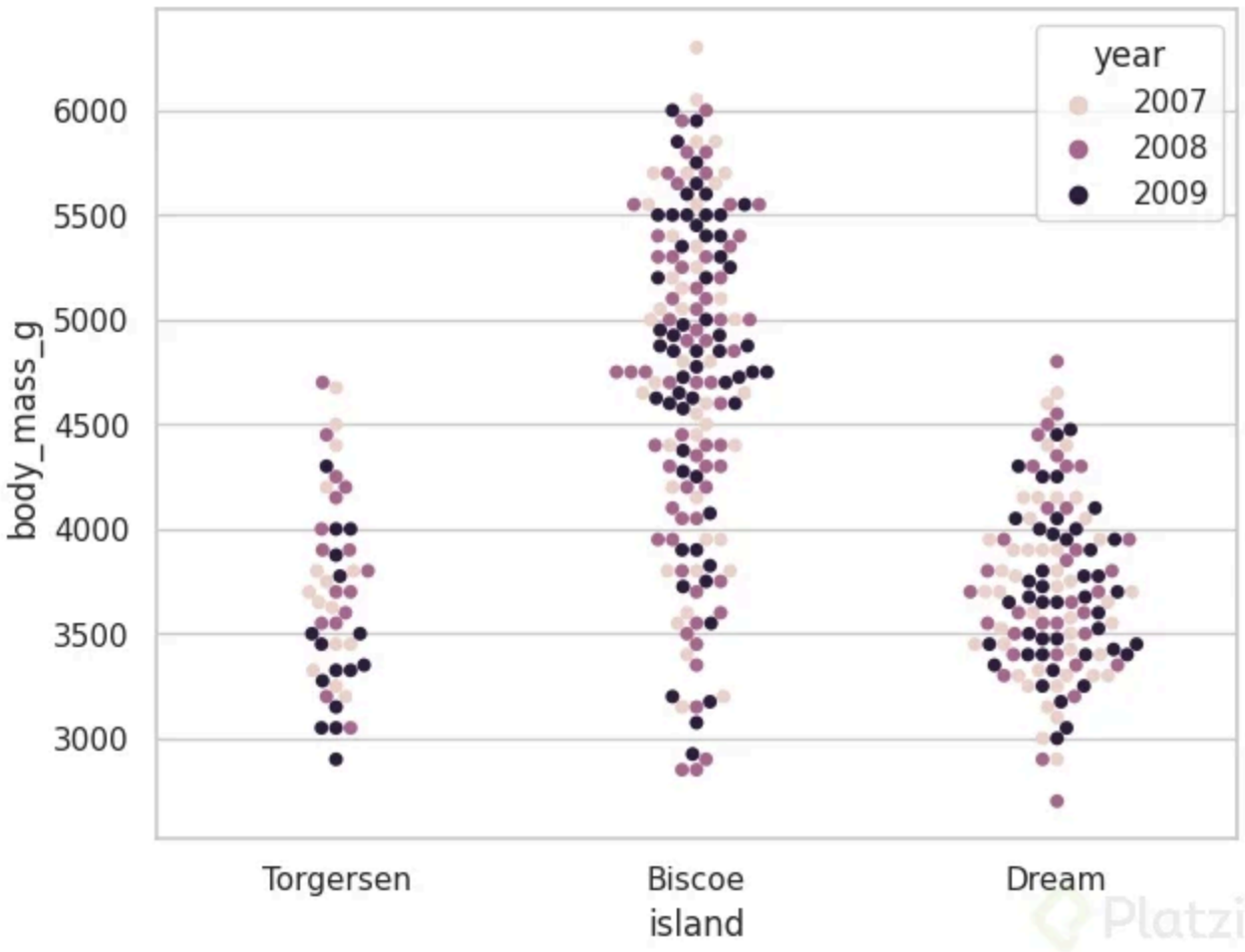
Ya hemos agregado ruido en la gráfica anterior, así tenemos una mejor idea de como están distribuidos los datos. En realidad los datos pertenecen a una sola observación, pero puedo ver en que partes se acumulan los datos.

Continuando con la idea, si nosotros tenemos una variable **numérica** y una variable **discreta**, es que podríamos visualizarlo con un box plot.



En este caso graficariamos la variable discreta en el eje X, y la variable numérica (distribución) en el eje Y.

Entonces ya estaríamos combinando 2 variables, una discreta y otra continua.



In [ ]:
```
# Importando librerías
import empiricaldist
import janitor
import matplotlib.pyplot as plt
import numpy as np
import palmerpenguins
import pandas as pd
import scipy.stats
import seaborn as sns
import sklearn.metrics
import statsmodels.api as sm
import statsmodels.formula.api as smf
import statsmodels.stats as ss
import session_info
```

Establecer apariencia general de las gráficas

In [ ]:
```
%matplotlib inline
sns.set_style(style='whitegrid')
sns.set_context(context='notebook')
plt.rcParams['figure.figsize'] = (11, 9.4)

penguin_color = {
    'Adelie': '#ff6602ff',
    'Gentoo': '#0f7175ff',
    'Chinstrap': '#c65dc9ff'
}
```

Cargar los datos

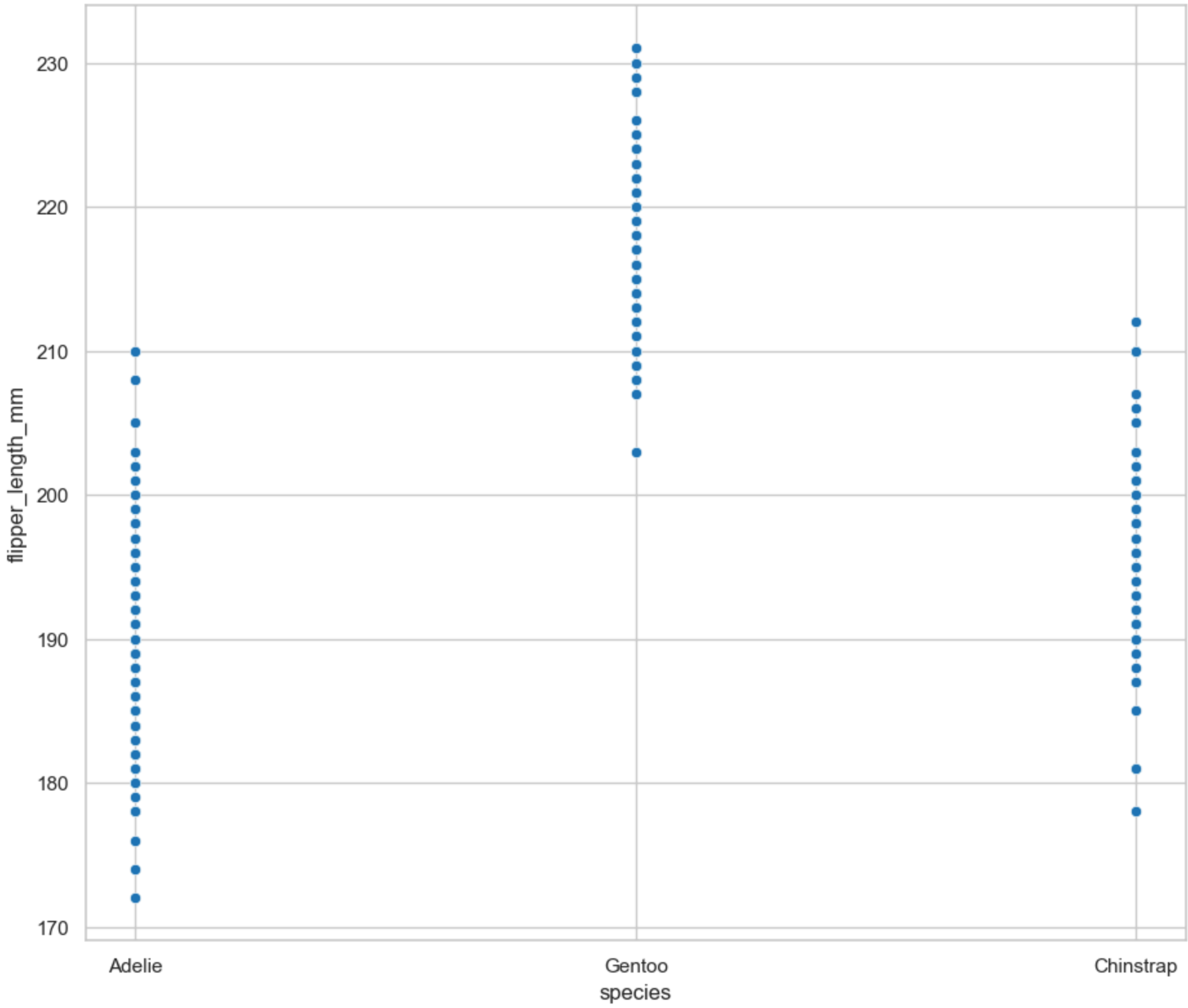
Datos Preprocesados

In [ ]:
```
preprocessed_penguins_df = pd.read_csv('dataset/penguins.csv')
```

Explorando relación entre variable categorica y numérica

In [ ]:
```
sns.scatterplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
)
```

Out[ ]:
```
<AxesSubplot: xlabel='species', ylabel='flipper_length_mm'>
```



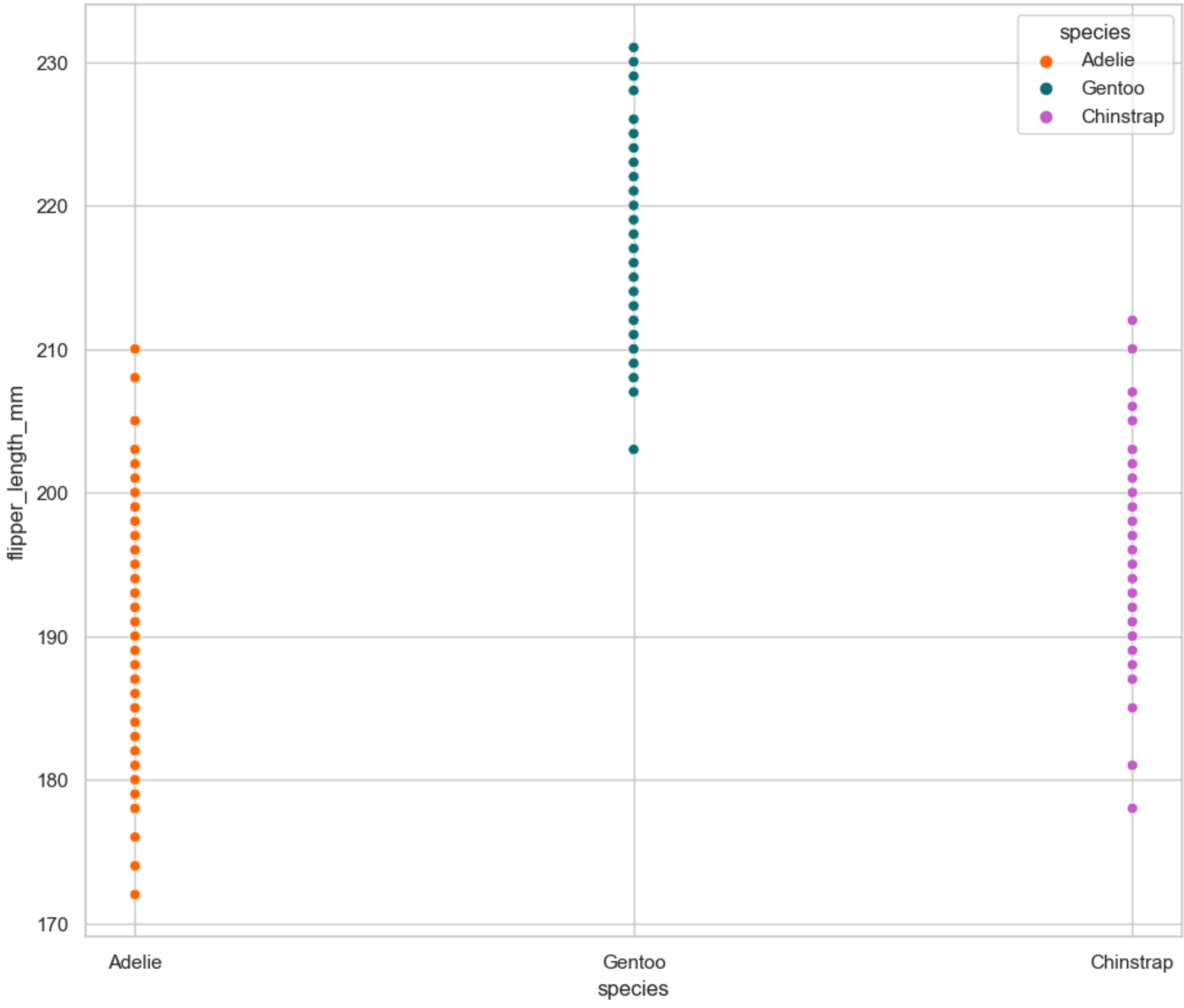
Tengo 3 lineas:

- Adelie
- Gentoo
- Chinstrap

Entonces claramente visualizar puntos por especie, yo puedo decir que los **Gentoo** tienen mayor longitud de aletas y además por observación visual la especie que sigue es la **Chinstrap**, o que se distribución esta más comprimida o junta.

In [ ]:
```
sns.scatterplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    hue='species',
    palette=penguin_color,
)
```

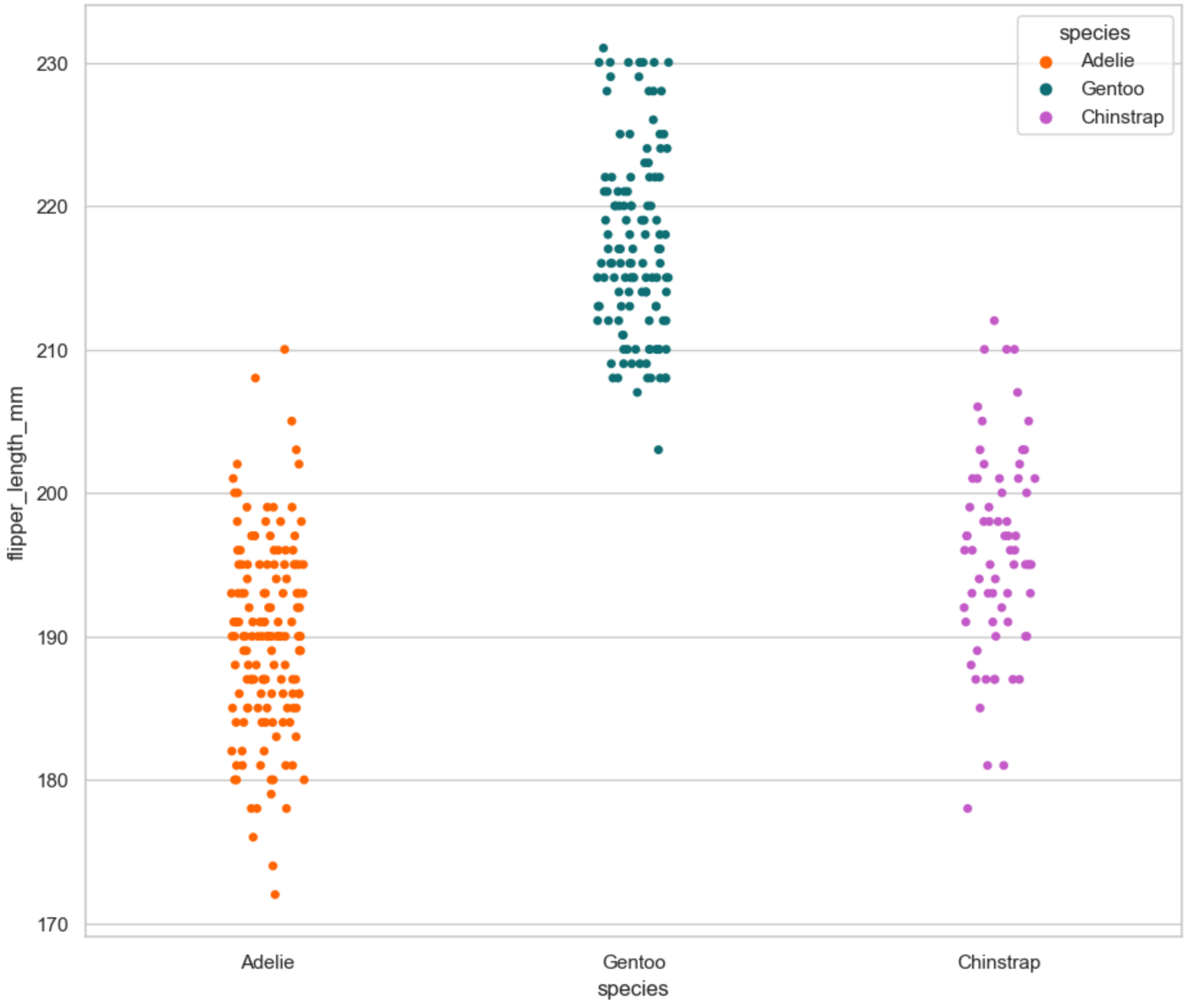
Out[ ]:
```
<AxesSubplot: xlabel='species', ylabel='flipper_length_mm'>
```



Otra forma de visualizar es añadirle ruido, para saber si se están encimando. Para hacerlo tenemos un plot diferente `stripplot`

In [ ]:
```
sns.stripplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    hue='species',
    palette=penguin_color,
)
```

Out[ ]:
```
<AxesSubplot: xlabel='species', ylabel='flipper_length_mm'>
```

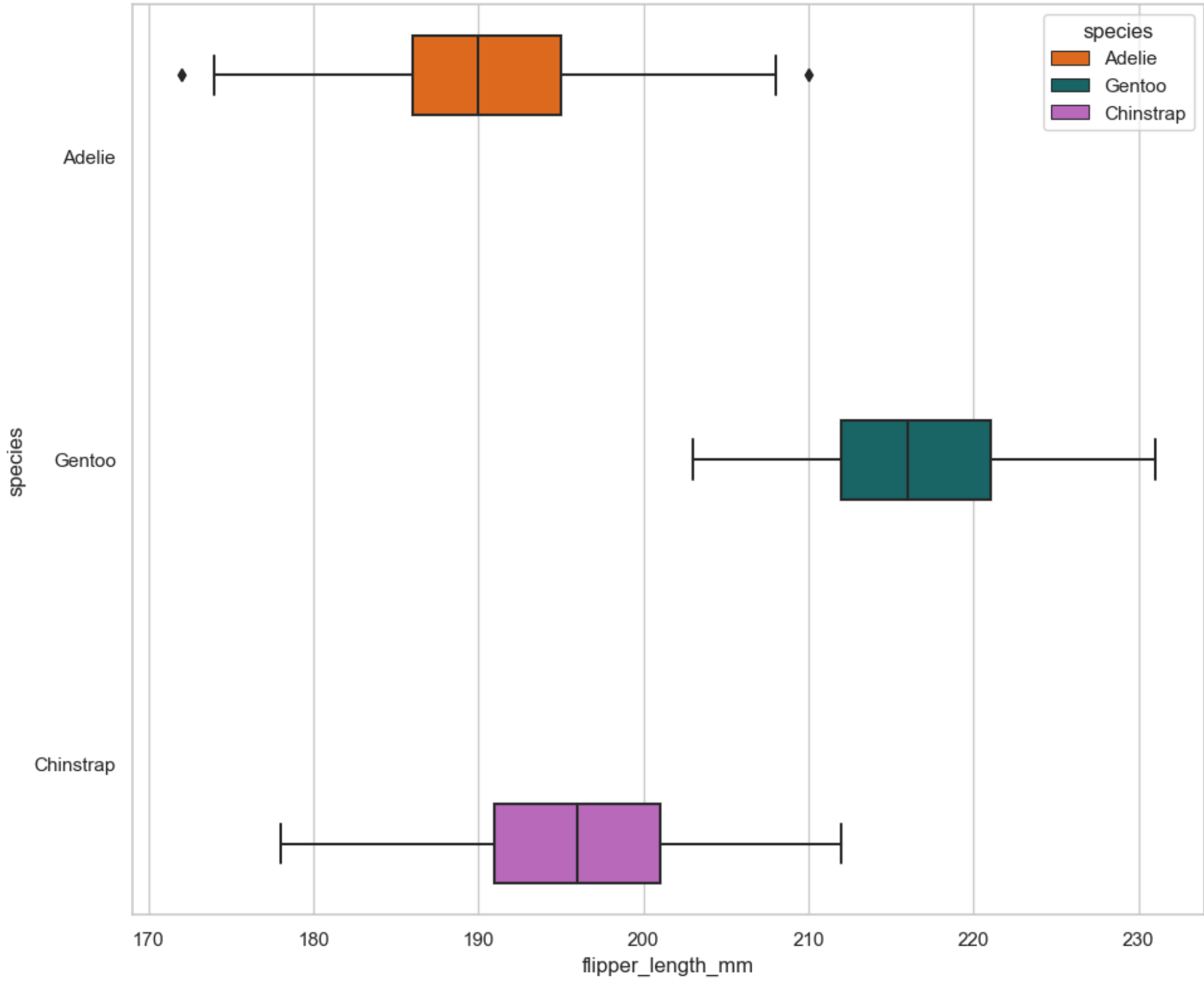


En este momento vemos la 3 categorías y podemos analizar por ejemplo en la especie **Adelie**, en el valor 190 se están juntando los valores.

Esto nos sirve para ver como se comportan 2 variables, otra cosa que podríamos hacer es un **boxplot**, para analizar las propiedades estadísticas que tienen las variables.

In [ ]:
```
sns.boxplot(
    data=preprocessed_penguins_df,
    y='species',
    x='flipper_length_mm',
    hue='species',
    palette=penguin_color,
)
```

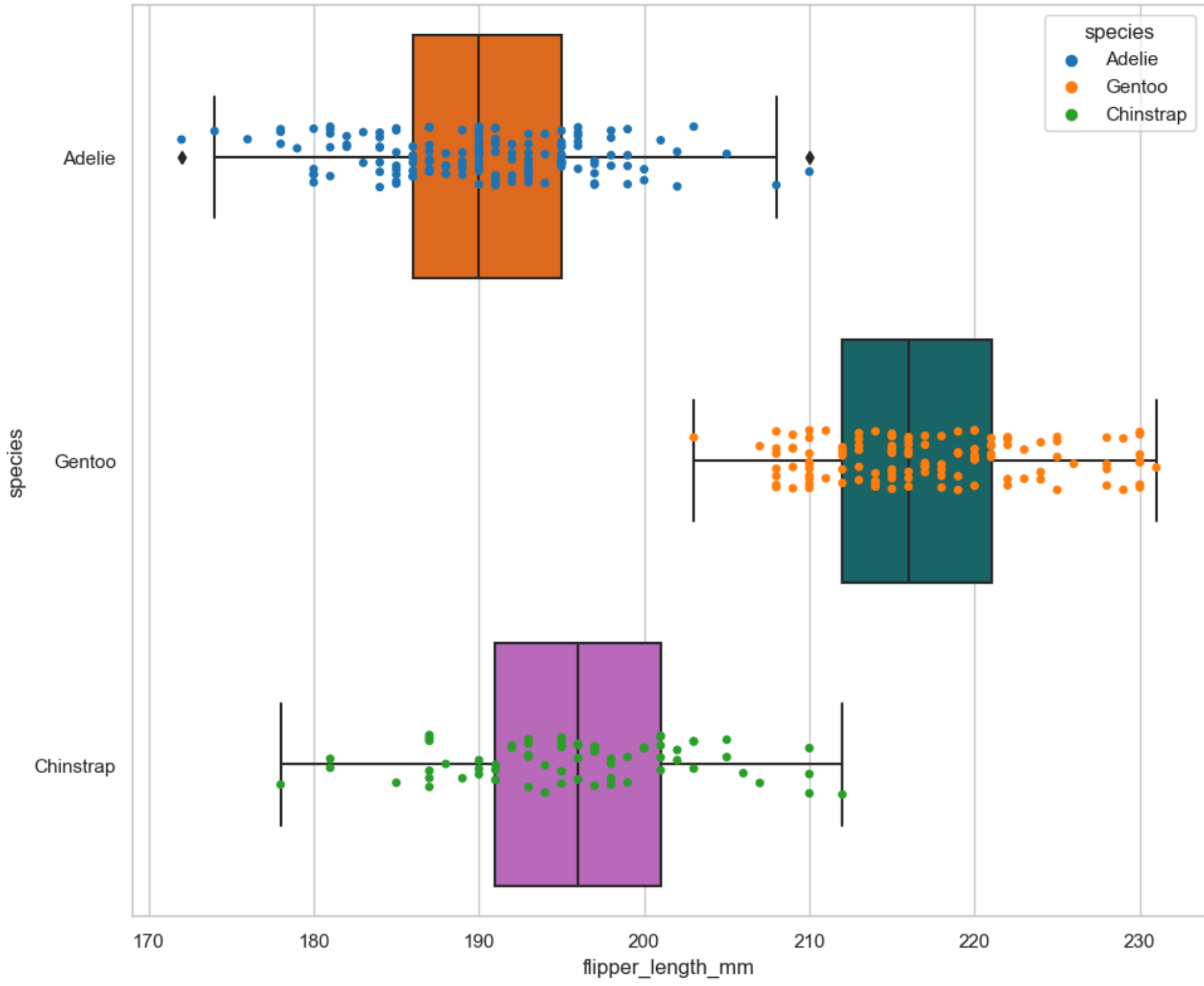
Out[ ]:
```
<AxesSubplot: xlabel='flipper_length_mm', ylabel='species'>
```



Como se puede observar, para mis pingüinos **Adelie**, tengo valores atípicos, se puede estimar el rango intercuartil, media, mínimo y máximo.

Sin embargo los boxplot, no nos dejan ver como se encuentran distribuidos nuestros datos. Entonces para apoyar a nuestro boxplot podemos añadir los puntos del scatter plot.

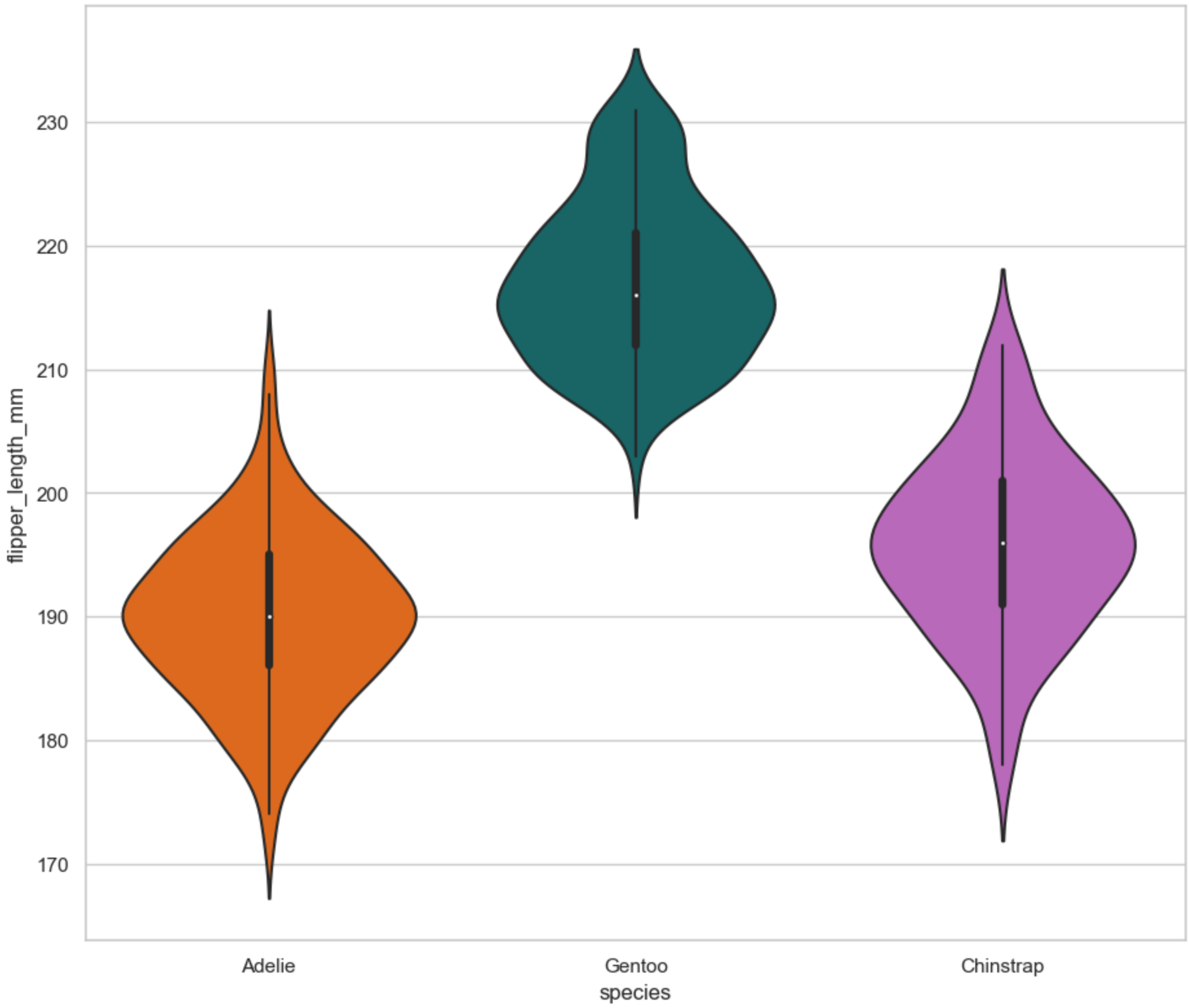
In [ ]:
```
ax = sns.boxplot(
    data=preprocessed_penguins_df,
    y='species',
    x='flipper_length_mm',
    palette=penguin_color,
)

ax = sns.stripplot(
        data=preprocessed_penguins_df,
    y='species',
    x='flipper_length_mm',
    hue='species',
    #palette=penguin_color,
)
```



Entonces ya podemos ver como se están distribuyendo mis puntos.

Gráfica de violin

In [ ]:
```
ax= sns.violinplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    #hue='species',
    palette=penguin_color,
)
```



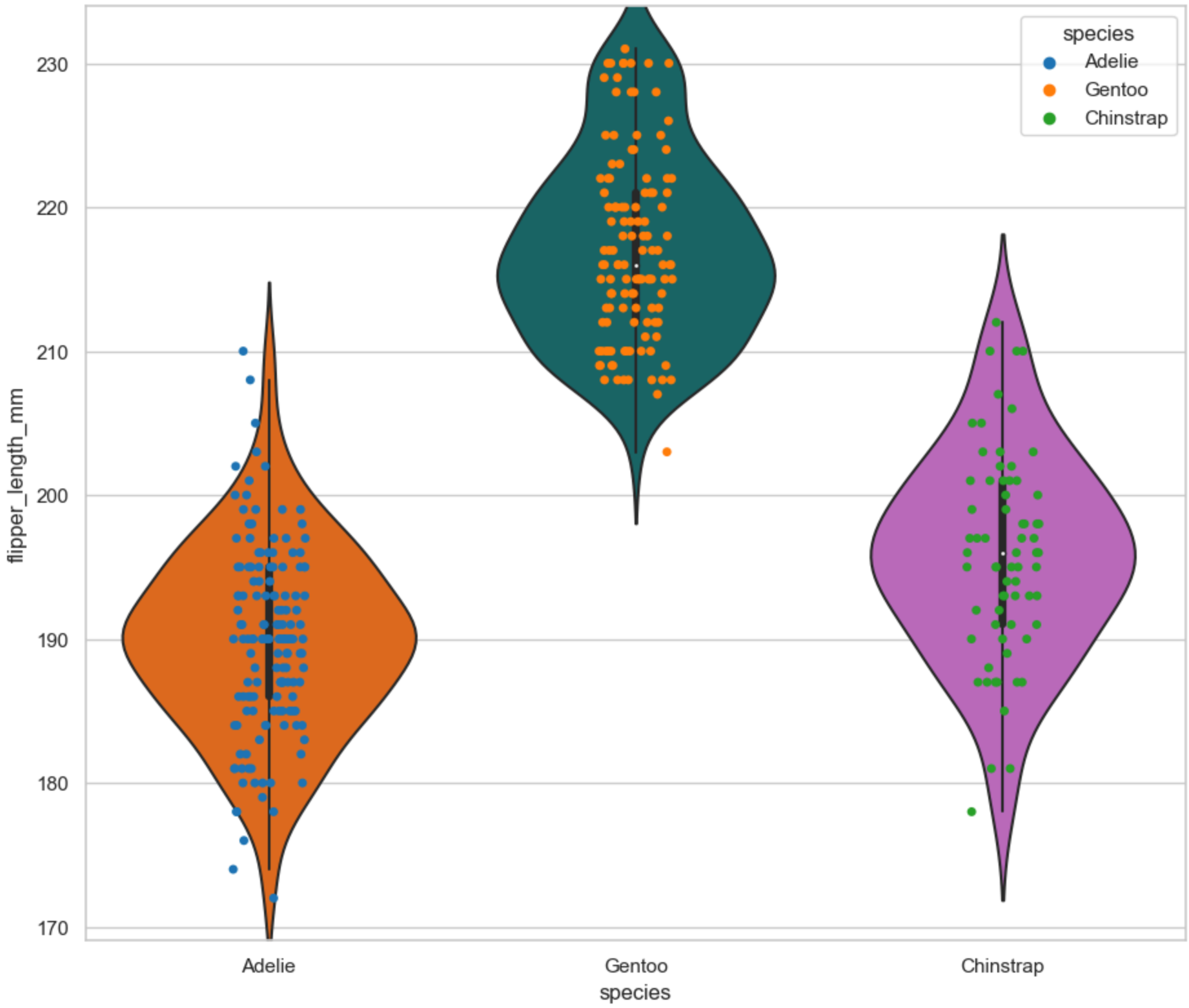
La diferencia entre esta gráfica y el KDE, es que me permite ver de manera simétrica como se comportan mis datos, y a la vez me da otros parámetros estadísticos dentro de el como:

- promedio o media.

También le podemos agregar puntos a la gráfica.

In [ ]:
```
ax= sns.violinplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    #hue='species',
    palette=penguin_color,
)
```

```
ax = sns.stripplot(
        data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    hue='species',
    #palette=penguin_color,
)
```



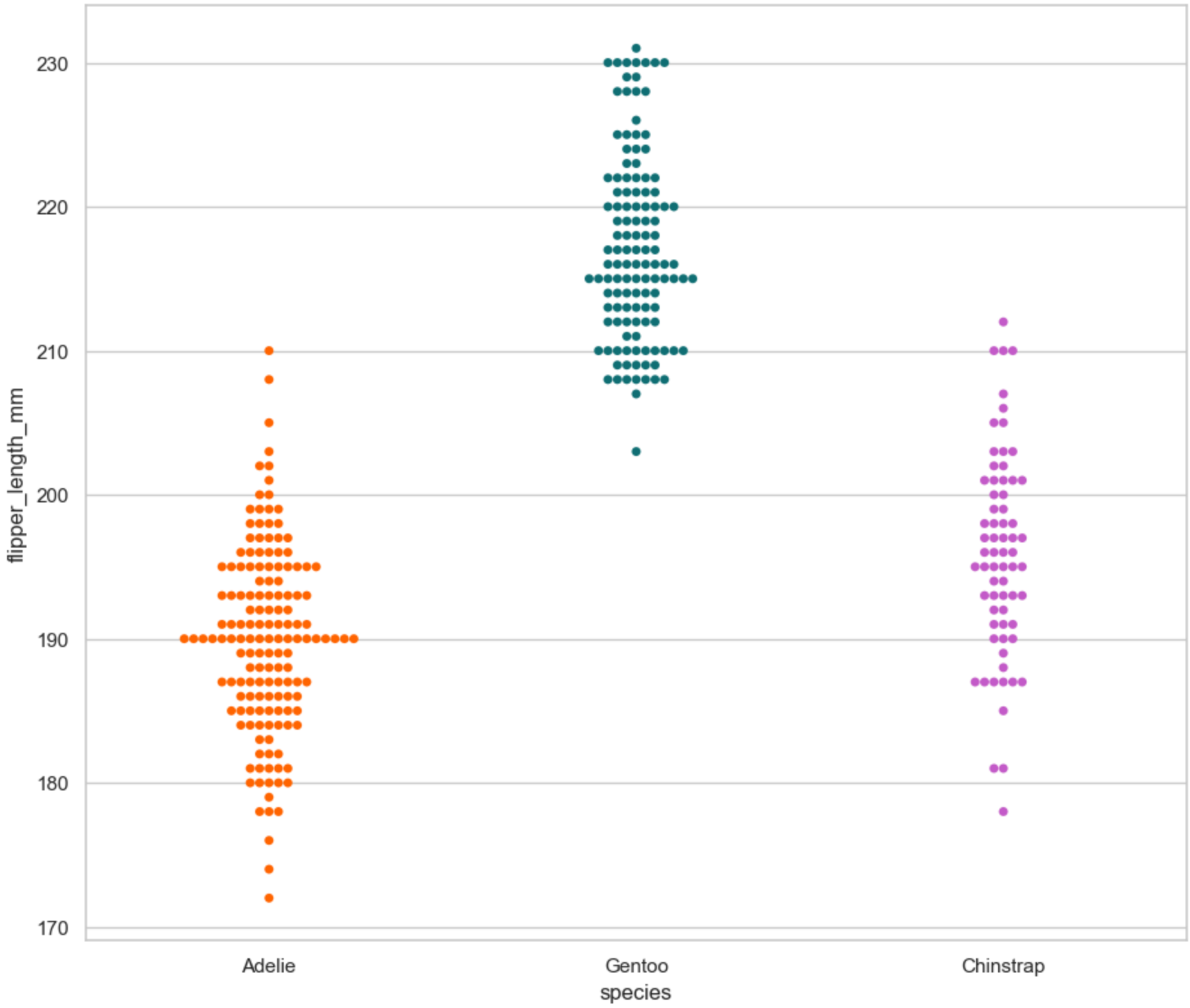
Combinar este tipo de gráficas me ayuda a tener mayor información de los datos.

Ahora ya tengo la idea de como están distribuidos mis puntos, pero también tengo la idea de la densidad.

Aquí puedo empezar a formular hipótesis; como por ejemplo si tomo los datos de mi especie **Adelie** se comporta como una distribución normal, pero si la mezclo con otra ya no se cumple. Y esto es simplemente de manera visual, con lo que después tendría que verificarlo con parámetros estadísticos.

In [ ]:
```
ax= sns.swarmplot(
    data=preprocessed_penguins_df,
    x='species',
    y='flipper_length_mm',
    #hue='species',
    palette=penguin_color,
)
```

```
/tmp/ipykernel_166433/2793029549.py:1: FutureWarning: Passing `palette` without assigning `hue` is deprecated.
  ax= sns.swarmplot(
```



Es similar a `stripplot` pero me permite ver la cantidad de puntos acumulados de una forma más ordenada y clara.

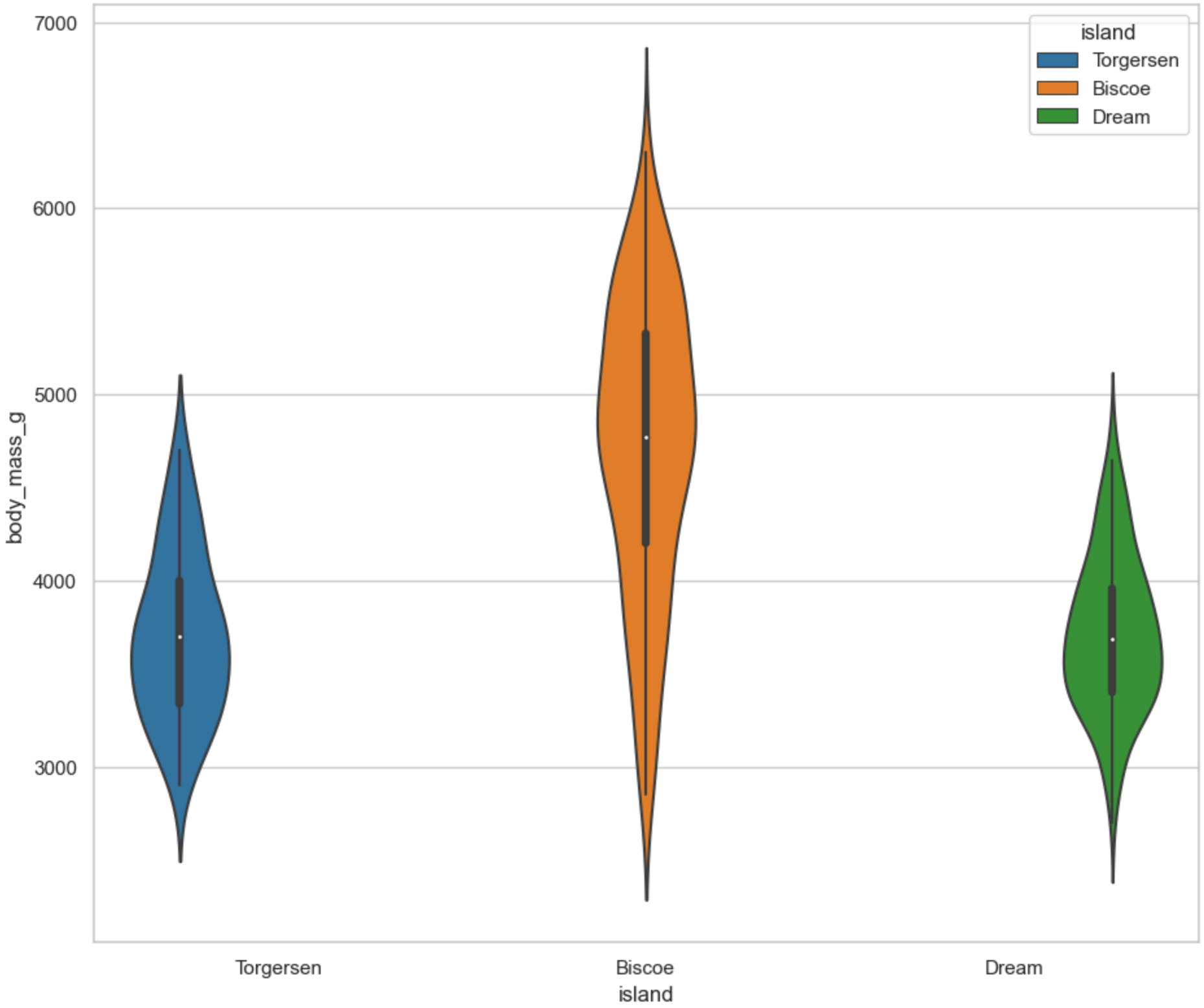
Así de manera visual puedo ver cuantos pingüinos tengo en cada medida del parámetro Y.

Así son diferentes visualizaciones, en algunos momentos voy a necesitar diferentes tipos.

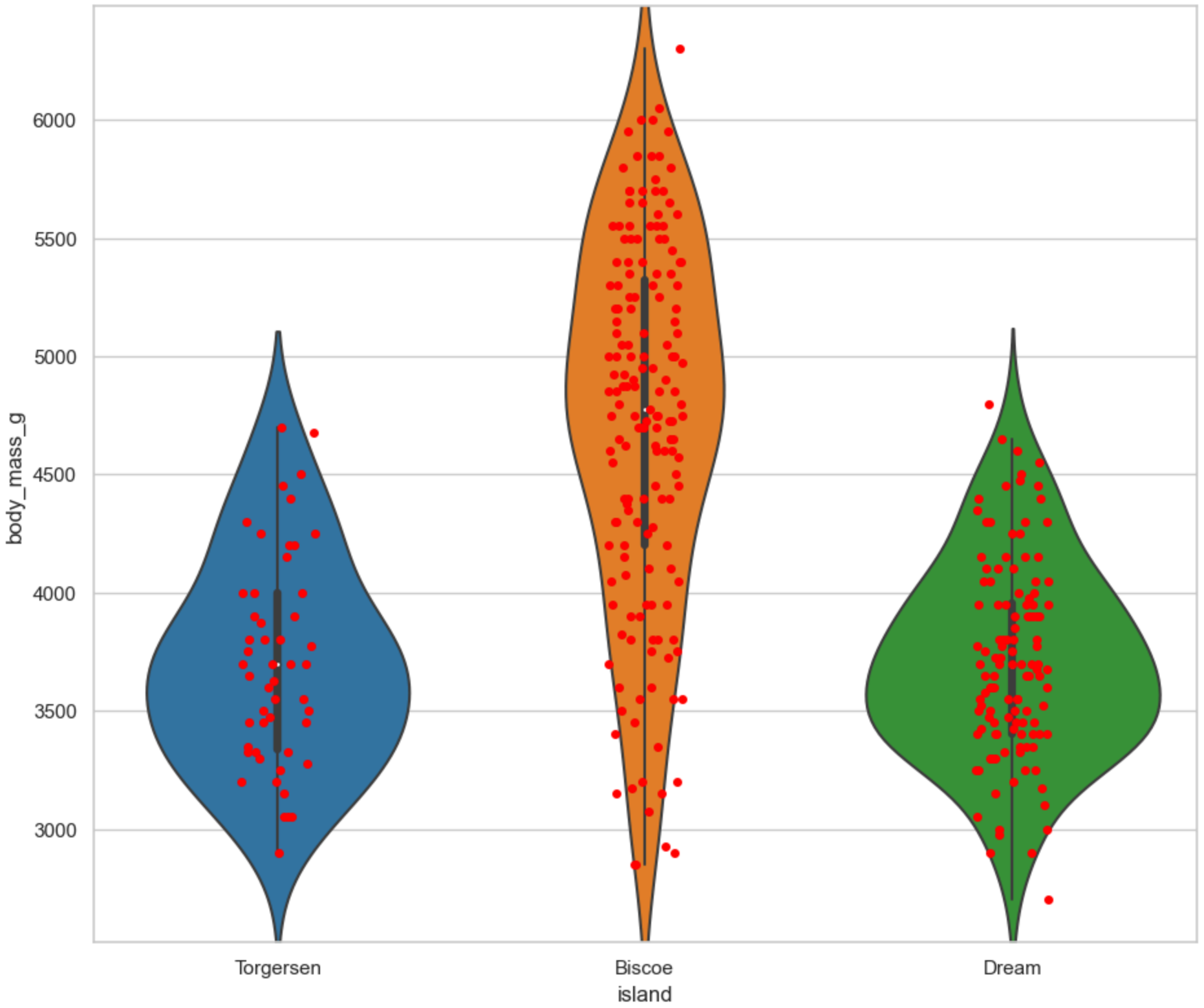
In [ ]:
```python
for columna in preprocessed_penguins_df.columns:
    print(columna)
```

```
species
island
bill_length_mm
bill_depth_mm
flipper_length_mm
body_mass_g
sex
year
```

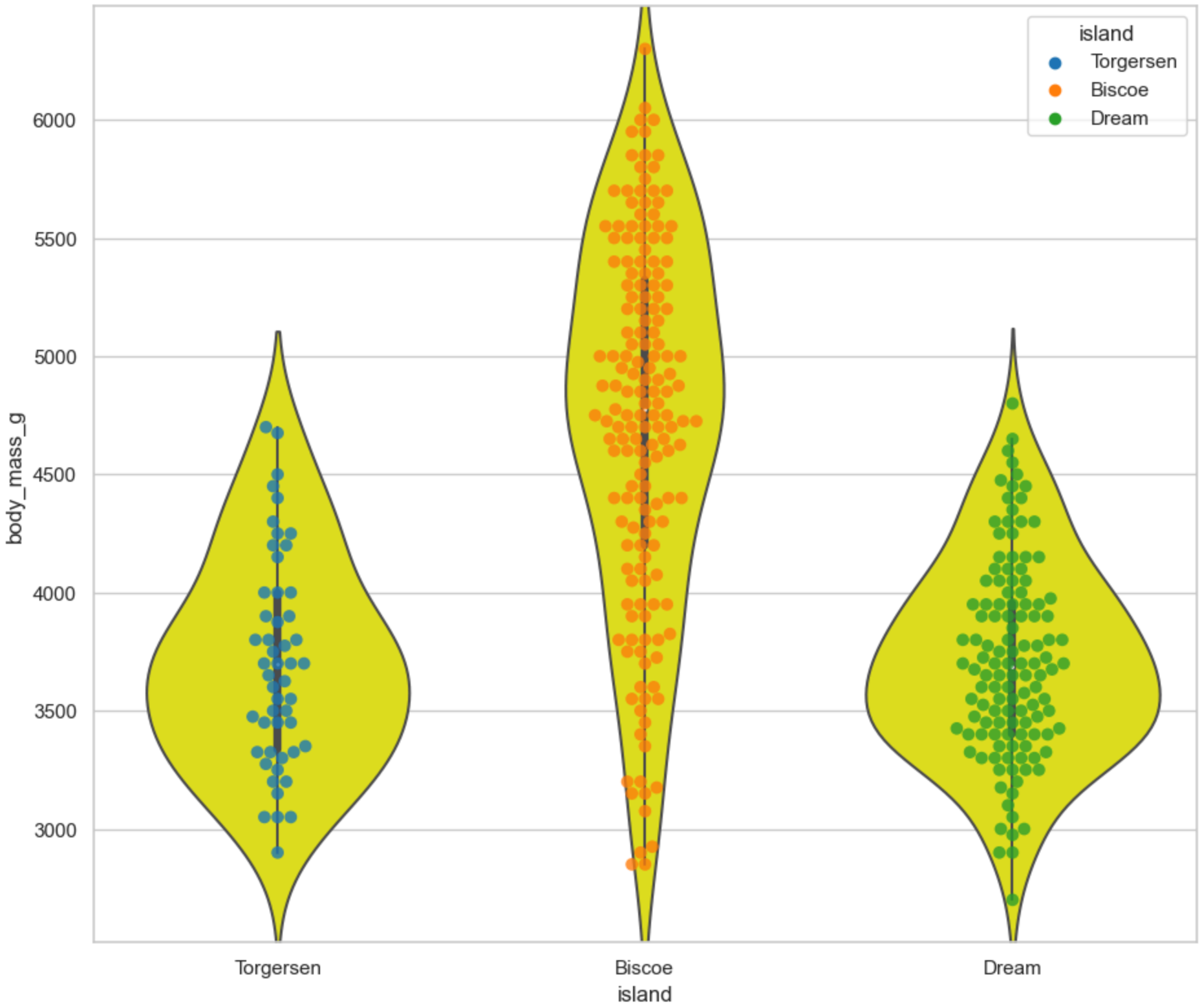
In [ ]:
```python
# ax= sns.boxplot(
#     data=preprocessed_penguins_df,
#     x='island',
#     y='body_mass_g',
# )

ax= sns.violinplot(
    data=preprocessed_penguins_df,
    x='island',
    y='body_mass_g',
    hue='island',
)
```



In [ ]:
```
ax= sns.violinplot(
    data=preprocessed_penguins_df,
    x='island',
    y='body_mass_g',
    #hue='species',
)

ax = sns.stripplot(
    data=preprocessed_penguins_df,
    x='island',
    y='body_mass_g',
    color='red'
    #hue='species',
    #palette=penguin_color,
)
```



In [ ]:
```
ax= sns.violinplot(
    data=preprocessed_penguins_df,
    x='island',
    y='body_mass_g',
    color='yellow'
)

ax = sns.swarmplot(
    data=preprocessed_penguins_df,
    x='island',
    y='body_mass_g',
    #color='red',
    s=7,
    alpha=0.8,
    hue='island',
    #palette=penguin_color,
)
```


island
Torgersen
Biscoe
Dream
body_mass_g
6000
5500
5000
4500
4000
3500
3000
Torgersen
Biscoe
Dream
island