

ACADEMICA

Agenda

Repaso: Matplotlib, bitácora y challenge.

Explicación: implementación de Seaborn

Break.

Hands-on training

Bonus Track: otras formas de visualizar.

Cierre.



Visualizaciones

Hoy es el turno de Seaborn, una librería de alto nivel para hacer gráficos en Python.

REPASO

A



Repaso del encuentro pasado

A



matplotlib

Nuestra primer Librería para la visualización de datos

¿Recuerdan cuáles eran sus beneficios y desventajas?

¿Qué tipos de gráficos se acuerdan?

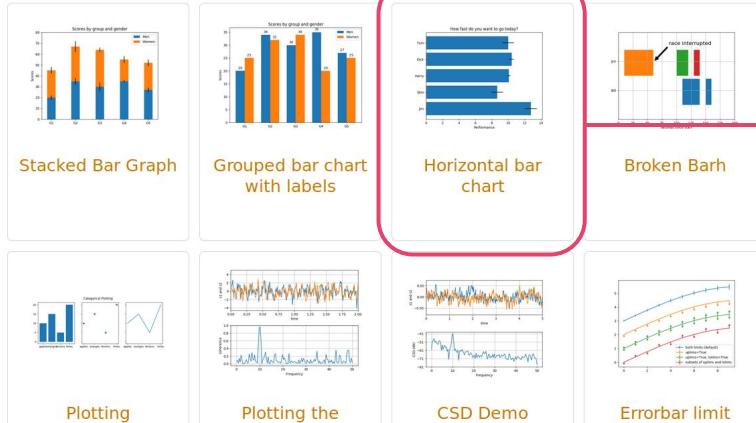
Documentación

Gallery

This gallery contains examples of the many things you can do with Matplotlib. Click on any image to see the full code.

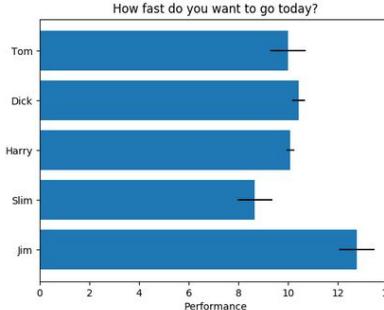
For longer tutorials, see our [tutorials page](#). You can also find [external resources](#) and a [FAQ](#) in our [user guide](#).

Lines, bars and markers



Horizontal bar chart

This example showcases a simple horizontal bar chart.



```
import matplotlib.pyplot as plt
import numpy as np

# Fixing random state for reproducibility
np.random.seed(19680801)

plt.rcParams()

fig, ax = plt.subplots()

# Example data
people = ('Tom', 'Dick', 'Harry', 'Slim', 'Jim')
y_pos = np.arange(len(people))
performance = 3 + 10 * np.random.rand(len(people))
error = np.random.rand(len(people))

ax.bart(y_pos, performance, xerr=error, align='center')
ax.set_yticks(y_pos)
ax.set_yticklabels(people)
ax.invert_yaxis() # labels read top-to-bottom
ax.set_xlabel('Performance')
ax.set_title('How fast do you want to go today?')

plt.show()
```



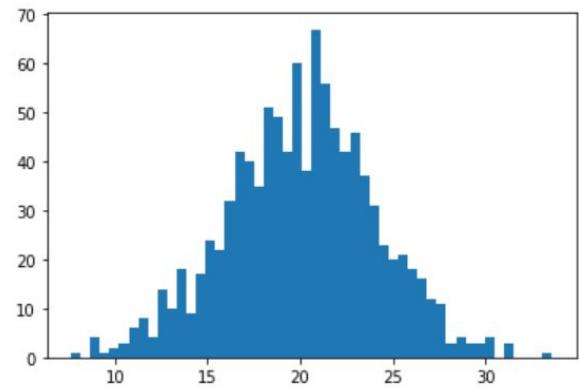
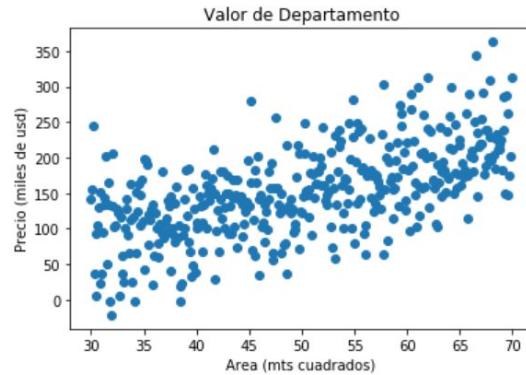
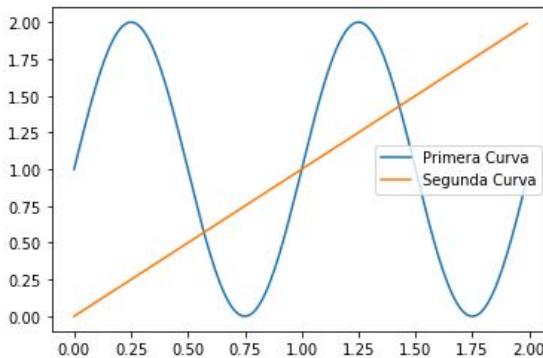
Graficar en un notebook

La parte de la librería que usaremos para graficar es **matplotlib.pyplot** y se suele importar con el nombre **plt**

```
[ ]: import matplotlib.pyplot as plt
```

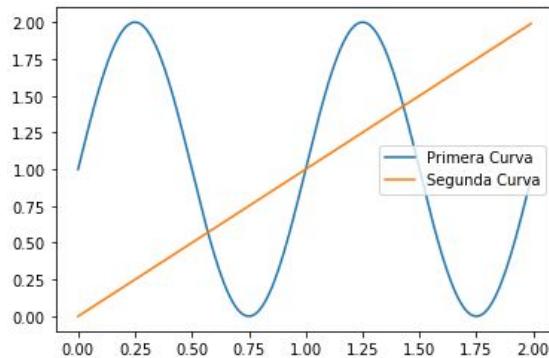


Tipos de Gráficos

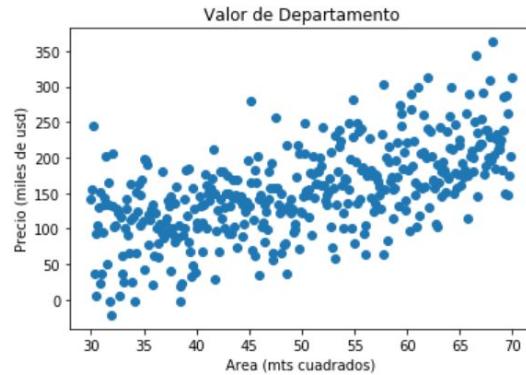


¿Cómo llamamos a cada gráfico?
¿Cuándo usarían uno u otro?

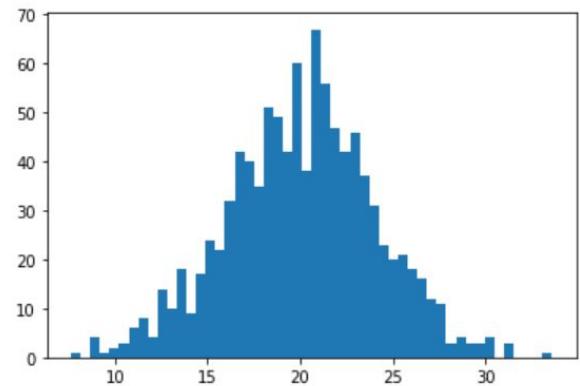
Tipos de Gráficos



Line Plot



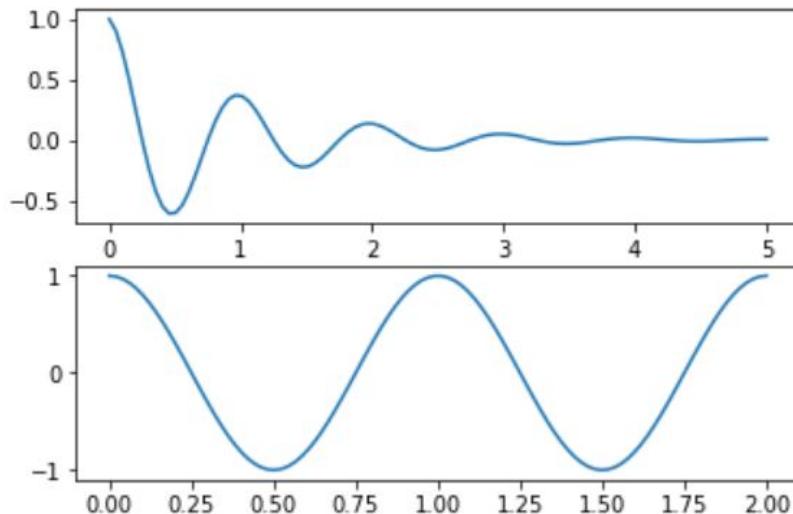
Scatter
Plot



Histograma

Tipos de Gráficos

Graficos Alineados en forma vertical



¿Y este tipo de gráfico?
¿Cuándo será útil?

EJERCICIO DEL ENCUENTRO PASADO

¡Muéstranos qué hiciste!

¿Qué cosas te costaron más del ejercicio? ¿Cómo las resolviste?

¿Cuál es el principal aprendizaje que te llevas?

Si tuvieras que hacerle alguna recomendación a alguien que va a hacer el ejercicio por primera vez, ¿qué le dirías?



EJERCICIO DEL ENCUENTRO PASADO

¿Alguien hizo algo diferente que quiera mostrar?



Repaso de la bitácora

A



REPASO

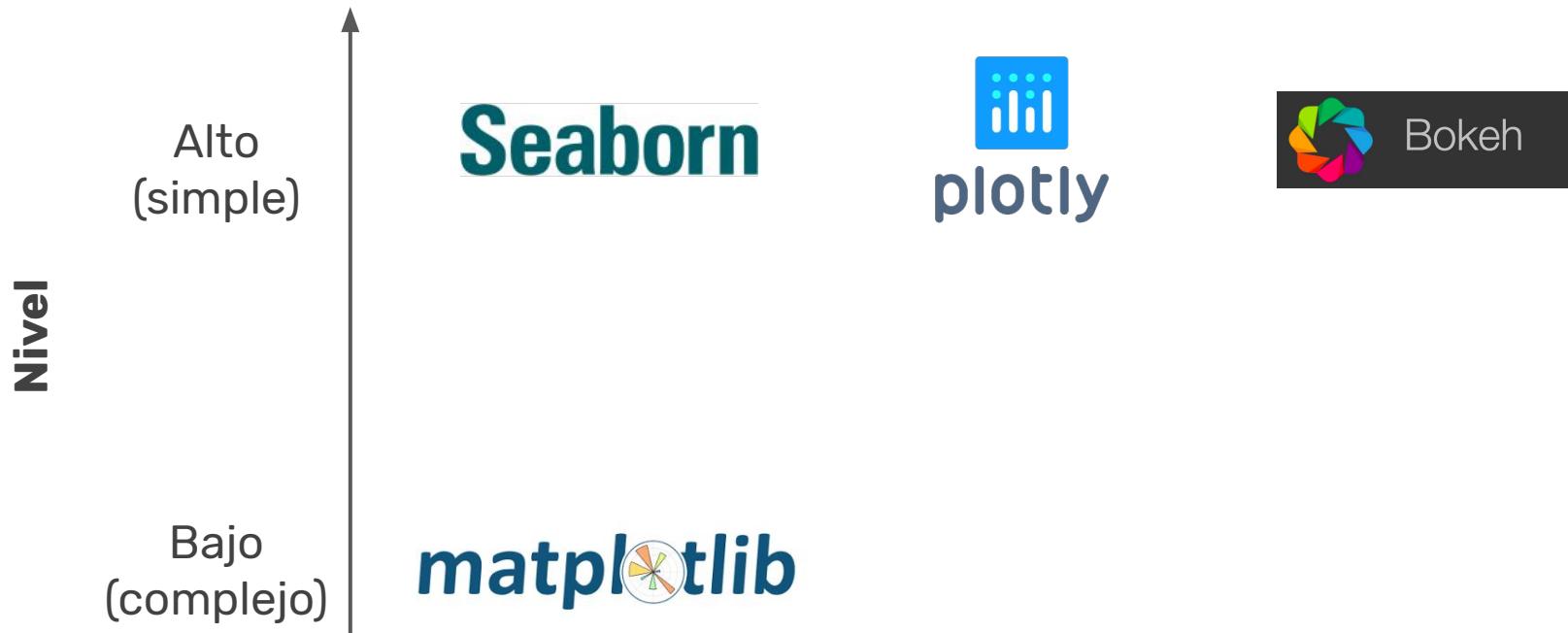
TEMAS BITÁCORA



Seaborn



Herramientas de Visualización



Herramientas de Visualización



¿ Seaborn o matplotlib ?



¿ Seaborn o matplotlib ?



- **Seaborn corre sobre Matplotlib.** Es por esto que trabaja con objetos definidos en esa librería, como por ejemplo figuras y ejes.
- La **manera de utilizarlo** es muy parecida (sabiendo usar Matplotlib resulta fácil usar Seaborn).
- La **principal diferencia** radica en la habilidad de Seaborn de poder importar los datos directamente desde un DataFrame.

Seaborn vs. matplotlib

VENTAJAS

- Facilita el trabajo con DataFrames
- Mejora automáticamente la estética de los gráficos
- Sintaxis sencilla para algunos gráficos complejos.

DESVENTAJAS

- Precisa la instalación de una librería adicional (esto puede ser perjudicial en algunos contextos)
- Menos flexible (configurable) que Matplotlib



CHALLENGE BITÁCORA

1. ¿Escribiste las listas?

- a. Gráficos apropiados para obtener la distribución de una variable numérica.
- b. Gráficos apropiados para obtener la distribución de una variable categórica.
- c. Gráficos apropiados para obtener la relación entre dos variables numéricas x e y .



2. ¿Resolviste el challenge del notebook?



CHALLENGE BITÁCORA



¿Alguien hizo algo diferente que quiera mostrar?

Implementación de Seaborn



Seaborn y DataFrames

Supongan que estamos trabajando con el dataset Iris:

```
[30]: iris_data = pd.read_csv('DS_Clase_04_iris.csv')
iris_data.head()
```

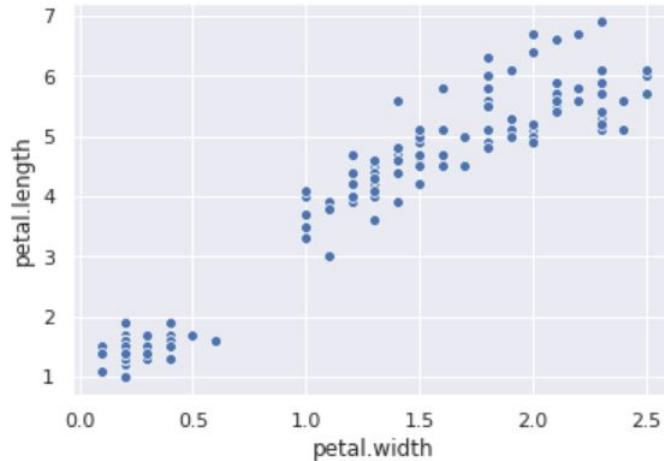
```
[30]:   sepal.length  sepal.width  petal.length  petal.width  variety
      0           5.1          3.5          1.4          0.2    Setosa
      1           4.9          3.0          1.4          0.2    Setosa
      2           4.7          3.2          1.3          0.2    Setosa
      3           4.6          3.1          1.5          0.2    Setosa
      4           5.0          3.6          1.4          0.2    Setosa
```

Seaborn y DataFrames

Con Seaborn podemos graficar tomando directamente los datos desde el DataFrame que los contiene:

```
[31]: sns.scatterplot(x="petal.width", y="petal.length", data=iris_data)
```

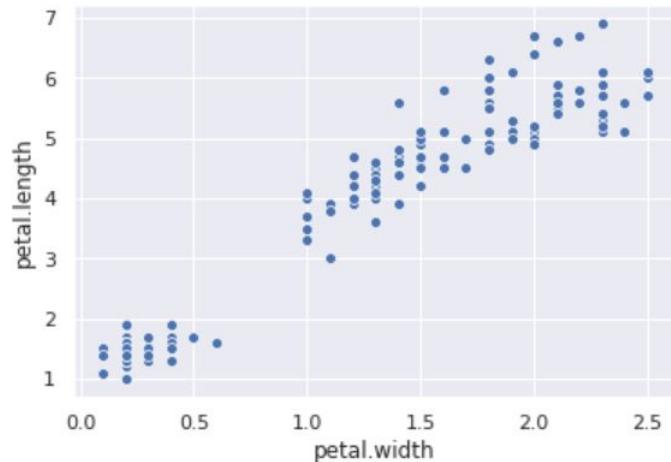
```
[31]: <matplotlib.axes._subplots.AxesSubplot at 0x7f48a8d4f208>
```



Seaborn y DataFrames

En '**x**' e '**y**' le decimos que columnas del DataFrame queremos que tome para graficar.

```
[31]: sns.scatterplot(x="petal.width", y="petal.length", data=iris_data)  
[31]: <matplotlib.axes._subplots.AxesSubplot at 0x7f48a8d4f208>
```



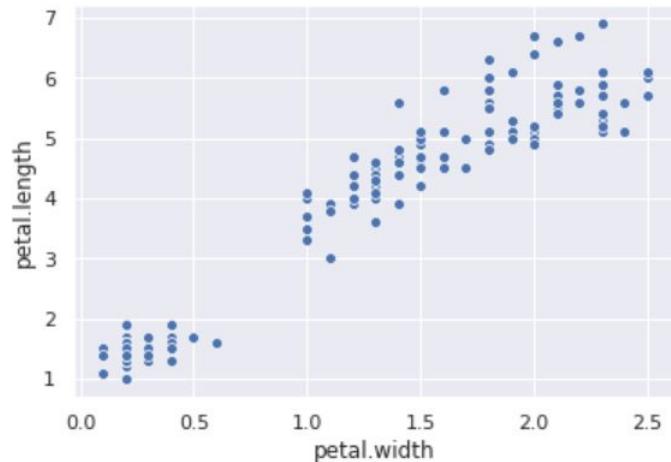
En 'data' le indicamos el nombre del **DataFrame**

Seaborn y DataFrames

Acá pasamos un **string**, con el nombre de las columnas del **DataFrame** que queremos que tome para graficar en el eje 'x' y el eje 'y'.

```
[31]: sns.scatterplot(x="petal.width", y="petal.length", data=iris_data)  
[31]: <matplotlib.axes._subplots.AxesSubplot at 0x7f48a8d4f208>
```

En 'data' le indicamos el nombre del **DataFrame**



Seaborn • Argumentos

Una de las características de **Seaborn** es que a través de los **argumentos** de sus funciones, nos permite diferenciar distintos **subsets** de los datos fácilmente.

seaborn

0.9.0

Gallery

Tutorial

API

Site ▾

Page ▾

seaborn.scatterplot

```
seaborn.scatterplot (x=None, y=None, hue=None, style=None, size=None, data=None, palette=None, hue_order=None,  
hue_norm=None, sizes=None, size_order=None, size_norm=None, markers=True, style_order=None, x_bins=None, y_bins=None, units=None,  
estimator=None, ci=95, n_boot=1000, alpha='auto', x_jitter=None, y_jitter=None, legend='brief', ax=None, **kwargs)
```

Seaborn • Argumentos

Estos argumentos son: **hue**, **style** y **size**.

seaborn 0.9.0 Gallery Tutorial API Site ▾ Page ▾

seaborn.scatterplot

```
seaborn.scatterplot (x=None, y=None, hue=None, style=None, size=None, data=None, palette=None, hue_order=None, hue_norm=None, sizes=None, size_order=None, size_norm=None, markers=True, style_order=None, x_bins=None, y_bins=None, units=None, estimator=None, ci=95, n_boot=1000, alpha='auto', x_jitter=None, y_jitter=None, legend='brief', ax=None, **kwargs)
```

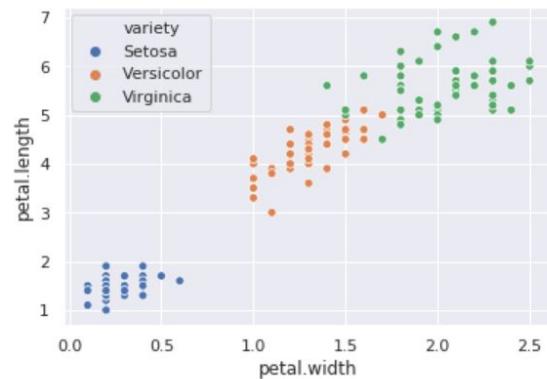
Seaborn • Argumento HUE

HUE

El argumento **hue** nos permite diferenciar nuestros datos según alguna **variables categórica** de nuestro dataset:

```
sns.scatterplot(x="petal.width", y="petal.length", hue="variety", data=iris_data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f48a8c8b208>
```



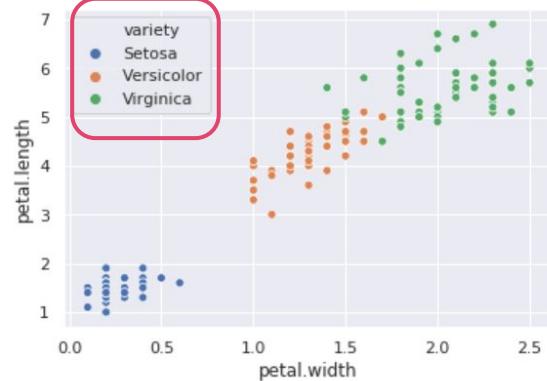
Seaborn • Argumento HUE

HUE

El argumento **hue** nos permite diferenciar nuestros datos según alguna **variables categórica** de nuestro dataset:

```
sns.scatterplot(x="petal.width", y="petal.length", hue="variety", data=iris_data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f48a8c8b208>
```



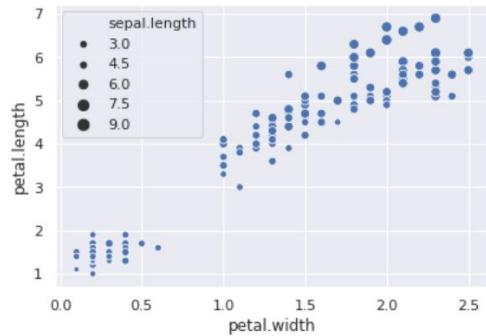
	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa

Seaborn • Otros argumentos

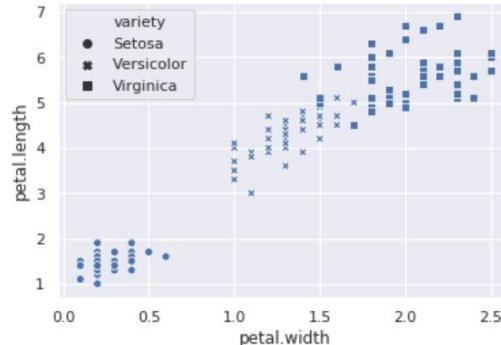
Otros argumentos

Explorá en la ejercitación los demás argumentos (size, style, etc).

```
sns.scatterplot(x="petal.width", y="petal.length", size="sepal.length", data=iris_data)  
<matplotlib.axes._subplots.AxesSubplot at 0x7f48a8f80780>
```



```
sns.scatterplot(x="petal.width", y="petal.length", style="variety", data=iris_data)  
<matplotlib.axes._subplots.AxesSubplot at 0x7f48a8cc39e8>
```

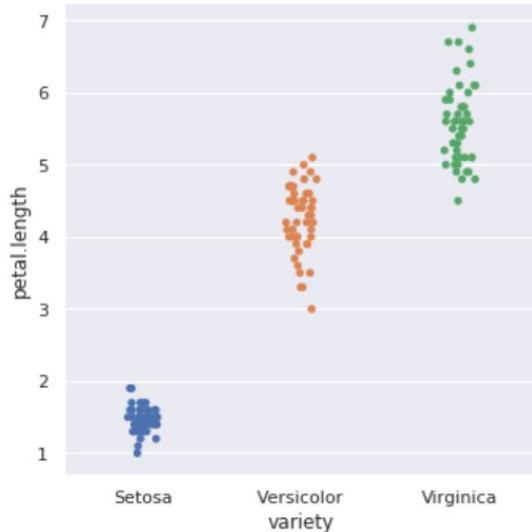


Categorical plots

Como su nombre indica, esta función de Seaborn nos permite graficar utilizando una **variable categórica** como eje.

Categorical plots

```
: sns.catplot(x="variety", y="petal.length", data=iris_data)
: <seaborn.axisgrid.FacetGrid at 0x7f48a8c941d0>
```

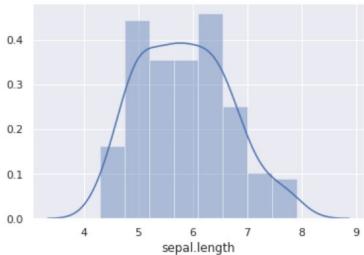


Seaborn realmente se **destaca** en este tipo de gráficos respecto a **Matplotlib**.

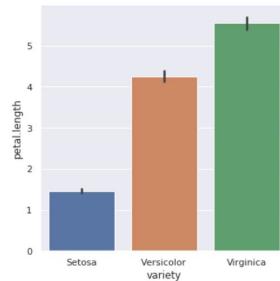
Noten que no “amontona” los puntos, permitiendo distinguir la cantidad.

Más tipos de gráficos

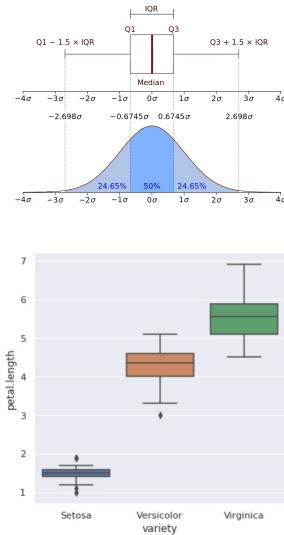
Histogram



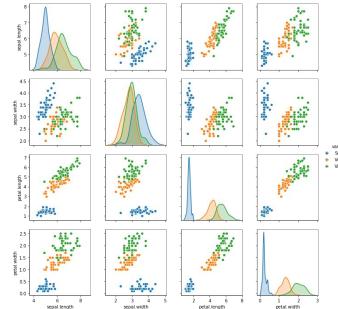
Barplot



Boxplot



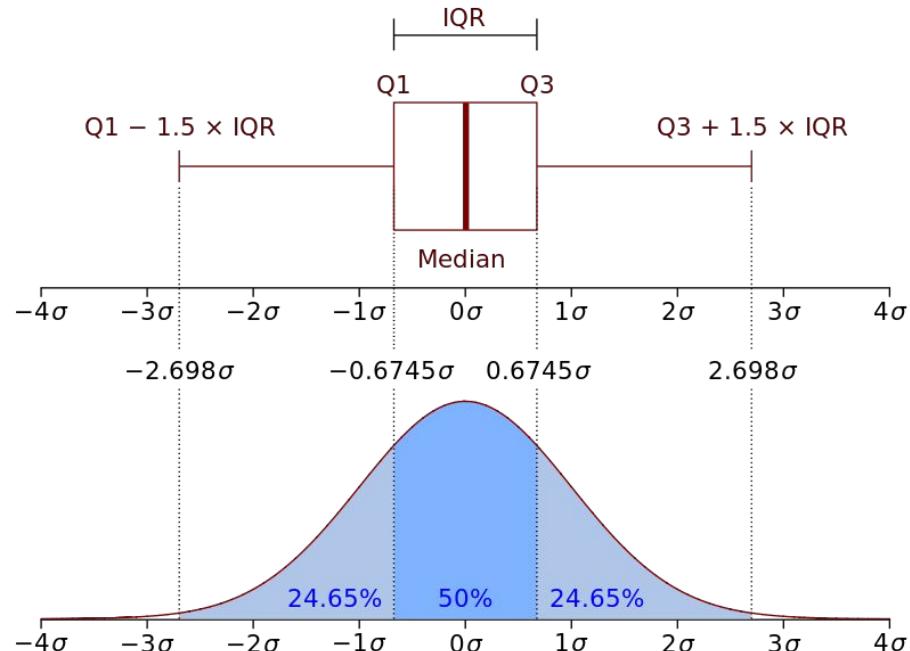
Pair plots



Boxplots

El **diagrama de cajas** es una forma de visualizar un conjunto de valores.

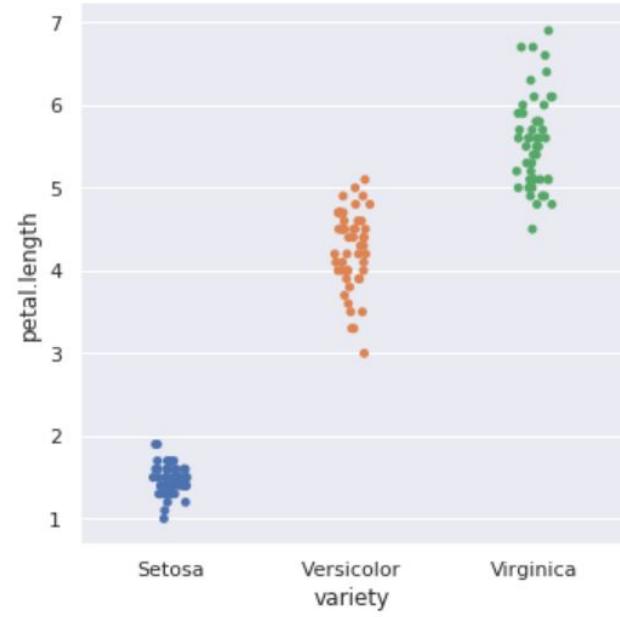
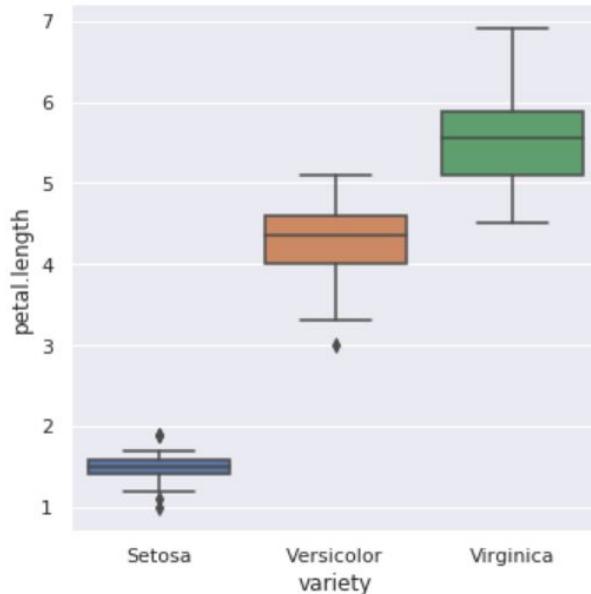
Muchas veces resulta más **informativa** que simplemente dibujar un punto por cada valor, ya que nos permite tener una idea de como es la distribución subyacente.



Misma información: dos visualizaciones

```
sns.catplot(x="variety", y="petal.length", kind='box', data=iris_data)
```

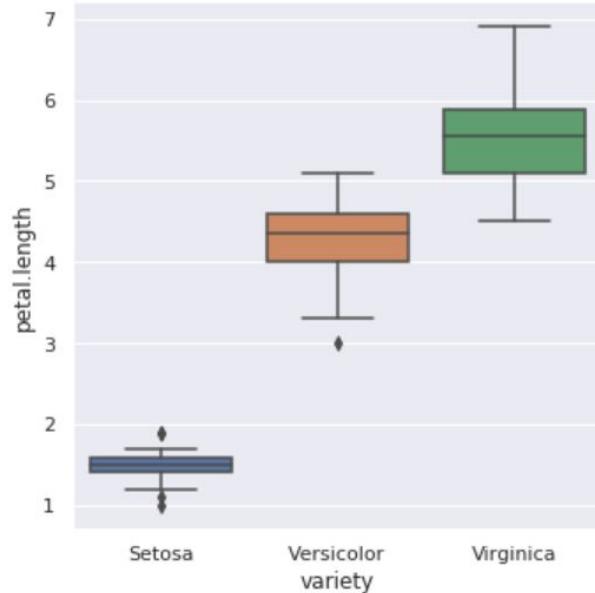
```
<seaborn.axisgrid.FacetGrid at 0x7f48a8b58e48>
```



Boxplots

```
sns.catplot(x="variety", y="petal.length", kind='box', data=iris_data)
```

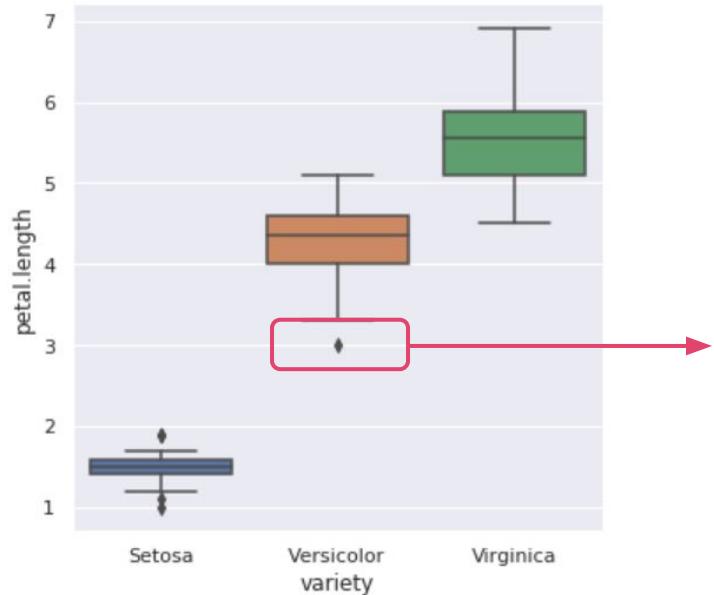
```
<seaborn.axisgrid.FacetGrid at 0x7f48a8b58e48>
```



Boxplots

```
sns.catplot(x="variety", y="petal.length", kind='box', data=iris_data)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f48a8b58e48>
```

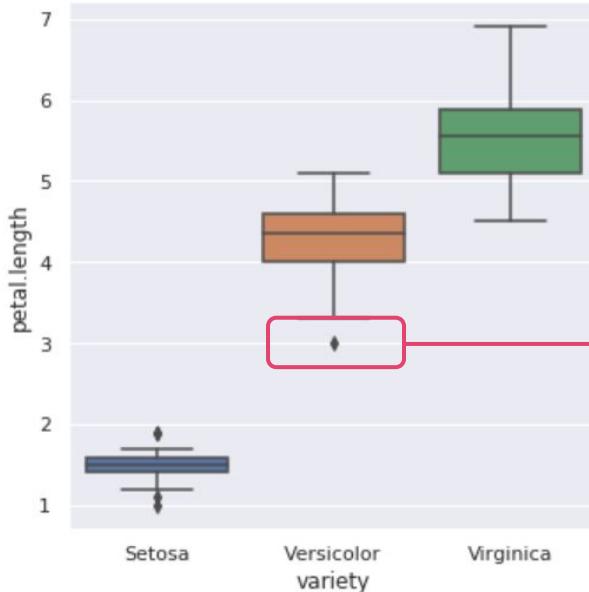


Los puntos fuera de los “**whiskers**”
pueden considerarse **Outliers**.

Boxplots

```
sns.catplot(x="variety", y="petal.length", kind='box', data=iris_data)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f48a8b58e48>
```



También se puede
hacer con el comando
Boxplot

Los puntos fuera de los “**whiskers**”
pueden considerarse **Outliers**.

Boxplots

Outlier:

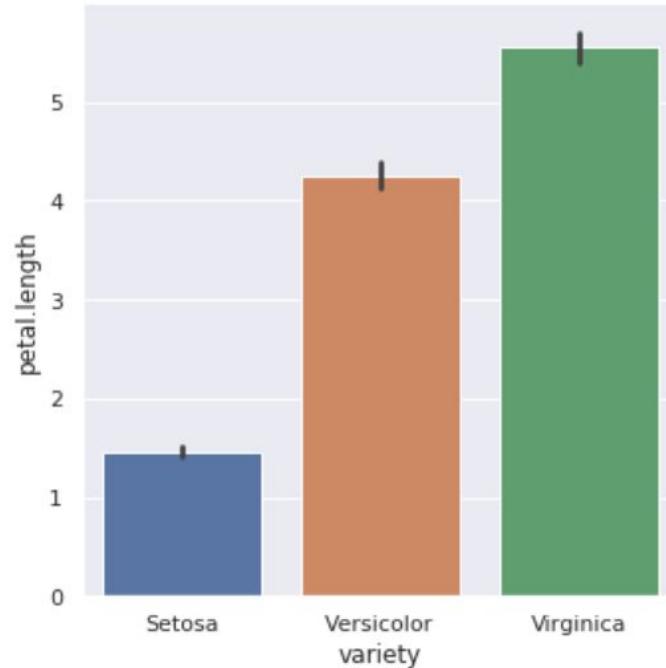
Es un concepto importante que desarrollaremos más adelante.



Categorical plots: Barplot

```
sns.catplot(x="variety", y="petal.length", kind='bar', data=iris_data)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f48a8e85c88>
```



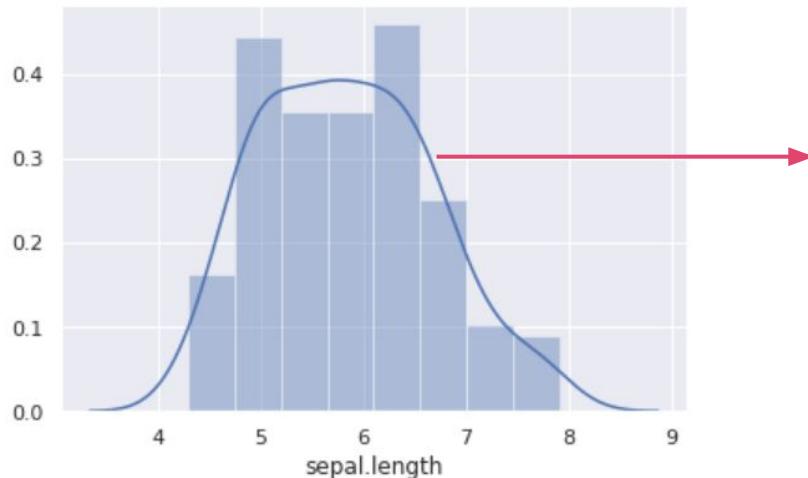
El **gráfico de barras** puede ser útil para ciertas circunstancias, pero nos da menos información que el Boxplot.

Histogramas

Seaborn también tiene la opción de graficar automáticamente **histogramas**, para hacerlo utiliza la función **hist** de **Matplotlib**.

```
sns.distplot(iris_data['sepal.length'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f48a89d5e80>
```



Por **default**, la función nos incluye una curva que intenta aproximar la distribución de la que vienen los datos. Su nombre es **KDE** ([kernel density estimation](#)).

Histogramas

seaborn.distplot

```
seaborn.distplot (a, bins=None, hist=True, kde=True, rug=False, fit=None, hist_kws=None, kde_kws=None, rug_kws=None,  
fit_kws=None, color=None, vertical=False, norm_hist=False, xlabel=None, label=None, ax=None)
```

Flexibly plot a univariate distribution of observations.

This function combines the matplotlib `hist` function (with automatic calculation of a good default bin size) with the seaborn `kdeplot()` and `rugplot()` functions. It can also fit `scipy.stats` distributions and plot the estimated PDF over the data.

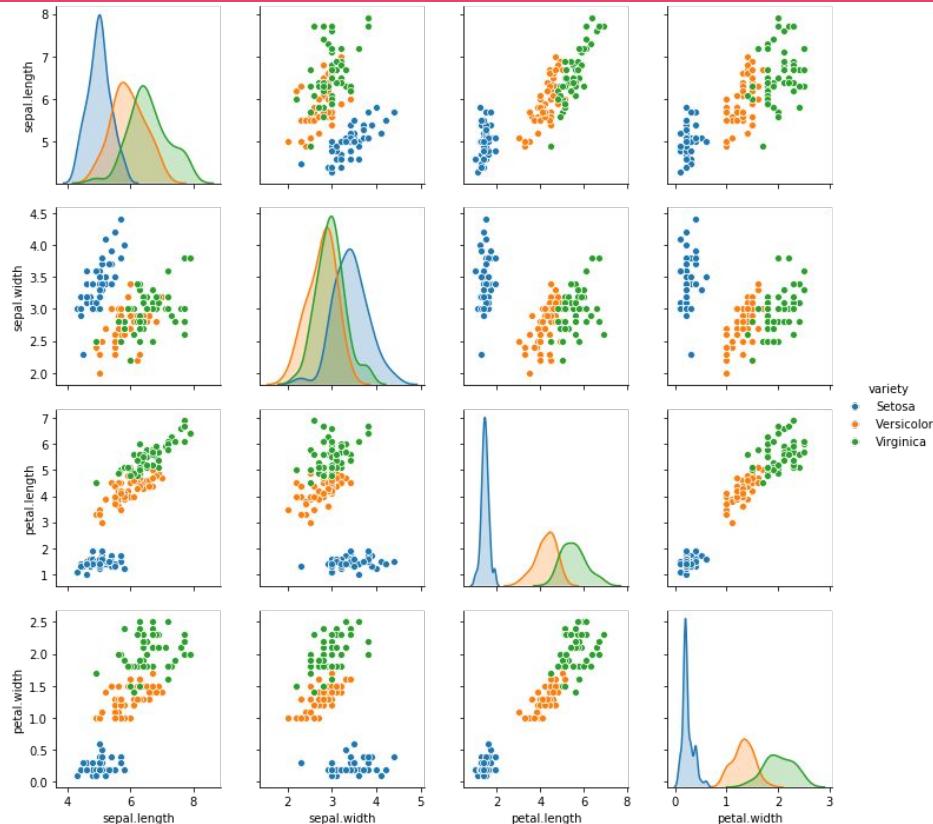
Pair Plots

Pairplot es una función de Seaborn que nos va a resultar **muy útil** a la hora de explorar los features de un dataset.

Compara todas las variables numéricas del dataset entre sí (gráficos scatter 2D). Al mismo tiempo nos permite colorear (**hue**) según una variable categórica.

Pair Plots

```
sns.pairplot(data=iris_data, hue="variety");
```





iBREAK!



Hands-on training



Hands-on
training

DS_Bitácora_07_Seaborn.ipynb



Otras formas de visualizar



Ejercitación

1. Elige uno de los siguientes gráficos que te mostramos a continuación y busca información que respalte estos resultados.
2. Piensa en qué otra información se puede visualizar de esta manera.

Think of the Children

% of children living in poverty



GRÁFICO #1

World Religions

By followers

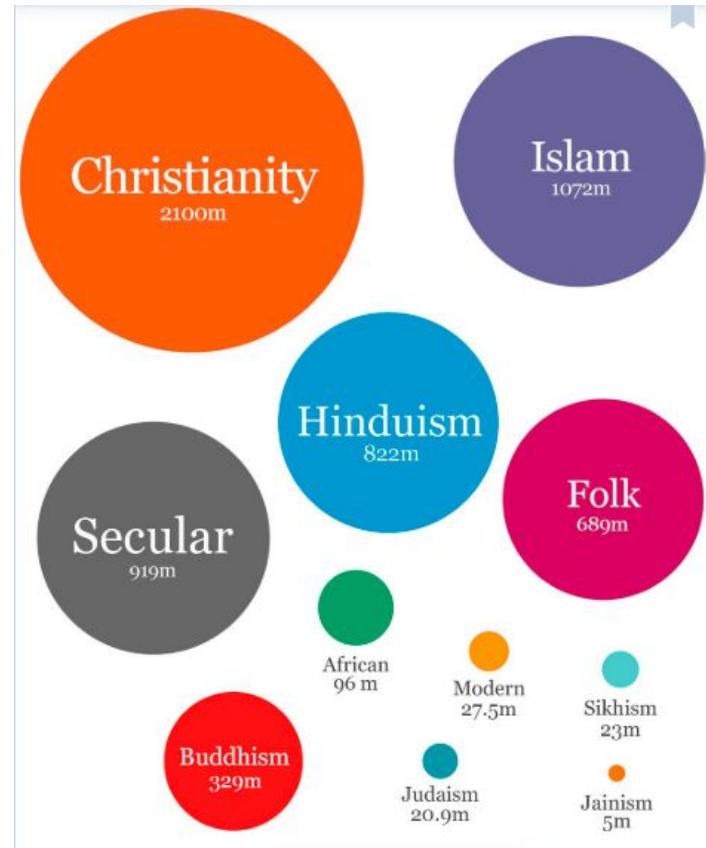


GRÁFICO #2

Recursos



Seaborn

- [**An Introduction to Seaborn**](#) - La documentación oficial de Seaborn ofrece una muy buena introducción a la librería, así como también un tutorial muy completo y accesible
- **Capítulo 4, "Visualization With Matplotlib", de [**Python Data Science Handbook**](#)**
 - Ese capítulo también tiene una entrada sobre Seaborn.
- [**10 tips to improve your plotting**](#) - una nota interesante con consejos generales sobre visualización en el ámbito de Data Science.

Para la próxima

- Termina el notebook de hoy.
- Lee la bitácora 08 y carga las dudas que tengas al Trello.

El próximo encuentro cerramos el primer tema de esta carrera:
Análisis Exploratorio de Datos.

Haremos un repaso y empezaremos a trabajar en el primer proyecto.

ACADEMICA