Practica de Seaborn terminada

Jacobo Hirsch Rdoriguez A00829679

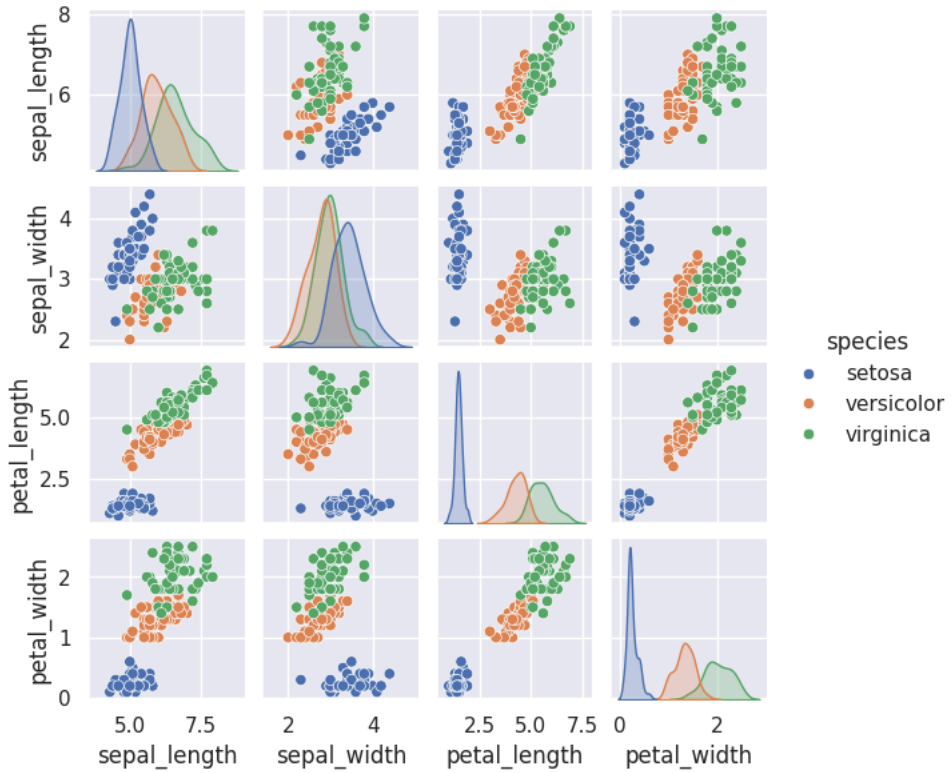
Lun 25 de agosto de 2024

```
import seaborn as sns
import matplotlib.pyplot as plt

iris = sns.load_dataset('iris')
iris.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
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

```

#Sirve para configurar el entorno de modo que los gráficos generados por matplotlib (una librería de gráficos
#dentro del notebook, en lugar de en una ventana separada.
%matplotlib inline
import seaborn as sns
sns.set()

sns.pairplot(iris, hue='species', size=1.5);
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/axisgrid.py:2100: UserWarning: The `size` parameter has been renamed to `hei
  warnings.warn(msg, UserWarning)
```



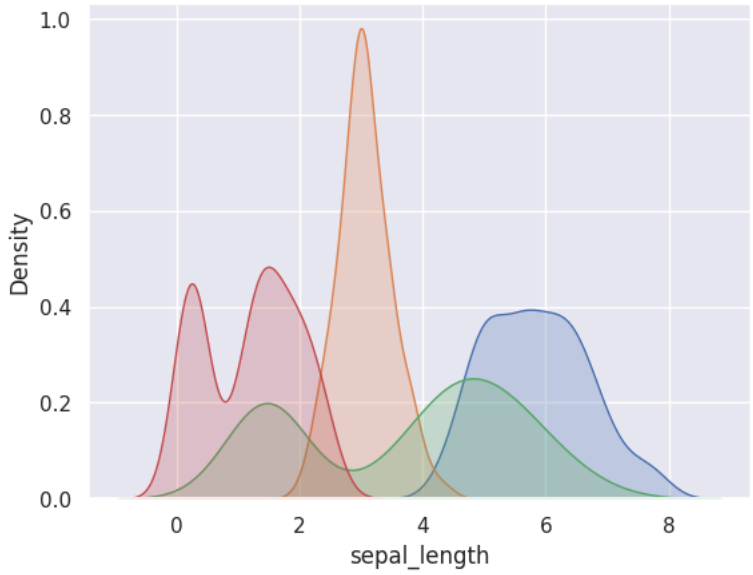
Esta línea de código crea un "pair plot" o gráfico de pares del DataFrame iris, que es un conjunto de datos famoso en el campo de la estadística y el aprendizaje automático, compuesto por medidas de las partes de la flor de iris para tres especies diferentes. El argumento hue='species' indica que el color de los puntos en el gráfico debe variar según la columna species en el DataFrame, lo cual ayuda a visualizar cómo las

diferentes especies se diferencian entre sí según las características medidas. El argumento size=1.5 especifica el tamaño de cada gráfico en el pair plot. Sin embargo, en versiones recientes de seaborn, este parámetro ha sido reemplazado por height, que especifica la altura de cada faceta del gráfico.

Rather than a histogram, we can get a smooth estimate of the distribution using a kernel density estimation, which Seaborn does with sns.kdeplot

```
1
2 sns.kdeplot(iris['sepal_length'], shade=True)
3 sns.kdeplot(iris['sepal_width'], shade=True)
4 sns.kdeplot(iris['petal_length'], shade=True)
5 sns.kdeplot(iris['petal_width'], shade=True)
```

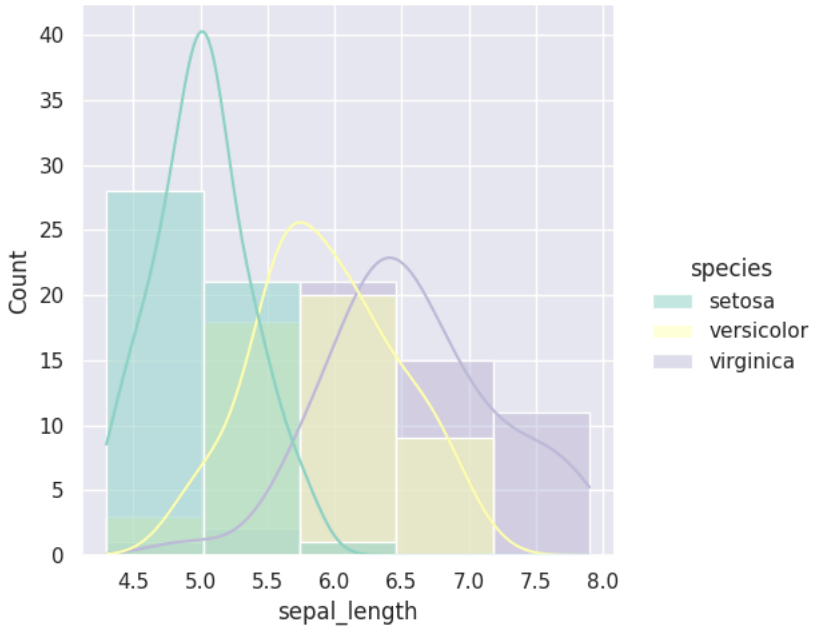
```
<ipython-input-19-2807d5d07346>:1: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(iris['sepal_length'], shade=True)
<ipython-input-19-2807d5d07346>:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(iris['sepal_width'], shade=True)
<ipython-input-19-2807d5d07346>:3: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(iris['petal_length'], shade=True)
<ipython-input-19-2807d5d07346>:4: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(iris['petal_width'], shade=True)
<Axes: xlabel='sepal_length', ylabel='Density'>
```



Histograms and KDE can be combined using distplot

```
1 sns.displot(x ='sepal_length',kde=True,bins = 5 ,
2             hue = iris['species'] , palette = 'Set3', data=iris)
```

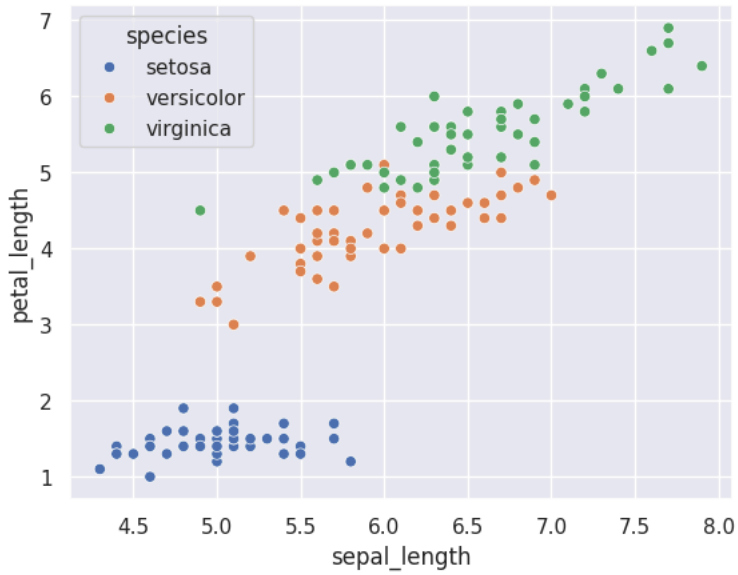
```
<seaborn.axisgrid.FacetGrid at 0x7fb3b3eb48b0>
```



You can also use scatter plots and color them by class easily

```
sns.scatterplot(x='sepal_length', y ='petal_length' ,data = iris , hue = 'species')
```

```
<Axes: xlabel='sepal_length', ylabel='petal_length'>
```



Practice Seaborn

Load the **diamonds** dataset from the Seaborn library (load_dataset)

Print the first 5 rows

Use a boxplot to see the spread of carats

Use a scatter plot to see carats compared to price and hue will be the color

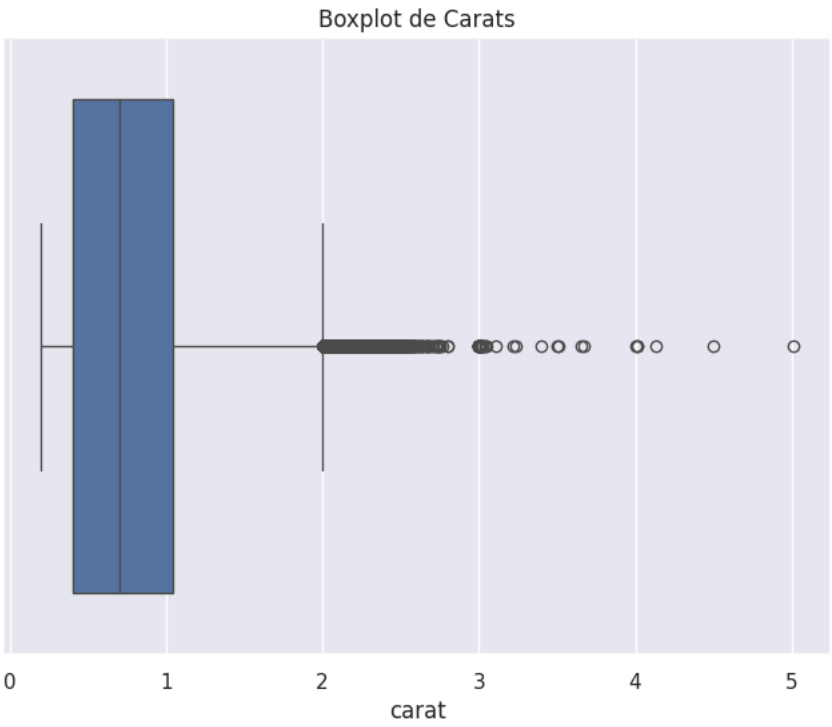
Use a categorical scatter plot to see differences in colors(x) and carats(y)

```
# Load the diamonds dataset

import matplotlib.pyplot as plt

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

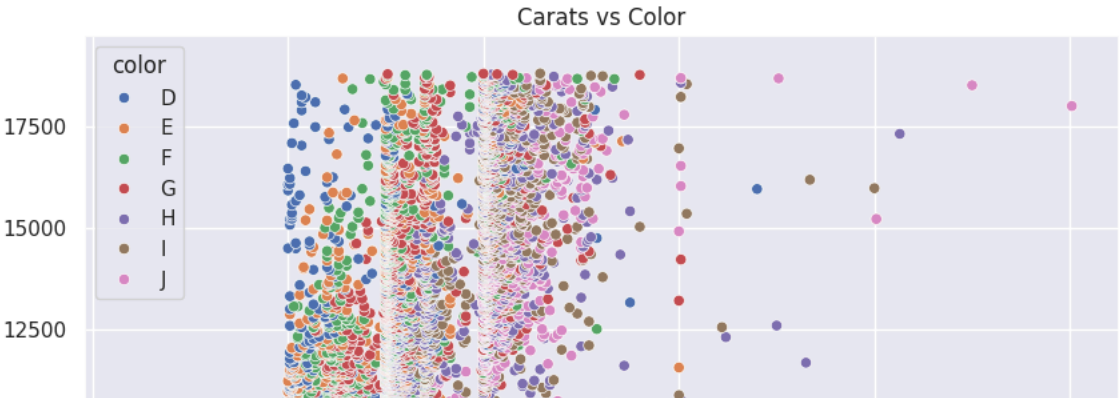
```

# Print the first 5 rows of the dataset
print(diamonds.head())
```

```
   carat      cut color clarity  depth  table  price     x     y     z
0   0.23    Ideal     E     SI2   61.5   55.0    326  3.95  3.98  2.43
1   0.21  Premium     E     SI1   59.8   61.0    326  3.89  3.84  2.31
2   0.23     Good     E     VS1   56.9   65.0    327  4.05  4.07  2.31
3   0.29  Premium     I     VS2   62.4   58.0    334  4.20  4.23  2.63
4   0.31     Good     J     SI2   63.3   58.0    335  4.34  4.35  2.75
```

```
# Use a boxplot to see the spread of caratsplt.figure(figsize=(8, 6))
plt.figure(figsize=(8, 6))
sns.boxplot(x='carat', data=diamonds)
plt.title('Boxplot de Carats')
plt.show()
```




```
# Use a scatter plot to see carats compared to price and hue will be the color
plt.figure(figsize=(10, 8))
sns.scatterplot(x='carat', y='price', hue='color', data=diamonds)
plt.title('Carats vs Color')
plt.show()
```




```
# Use a categorical scatter plot to see differences in colors(x) and carats(y)
plt.figure(figsize=(10, 8))
sns.stripplot(x='color', y='carat', data=diamonds, jitter=True)
plt.title('Categorical Scatter Plot of Diamond Color vs. Carats')
plt.show()
```