

```
# EMILIO BERBER MALDONADO - A01640603
# ACT 3: Cartwheel and Iris
# SEMANA TEC
```

Visualizing Data in Python

When working with a new dataset, one of the most useful things to do is to begin to visualize the data. By using **tables**, **histograms**, **boxplots**, **scatter plots** and other visual tools, we can get a better idea of what the data may be trying to tell us, and we can gain insights into the data that we may have not discovered otherwise.

In this notebook will use the [Seaborn](#) data processing library, which is a higher-level interface to **Matplotlib** that can be used to simplify many visualization tasks

The **Seaborn** provides visualisations tools that will allow to explore data from a graphical perspective.

Acknowledgments

- Data from <https://www.coursera.org/> from the course "Understanding and Visualizing Data with Python" by University of Michigan

▼ Importing libraries

```
# Import the packages that we will be using
import pandas as pd
import seaborn as sns ## graficas avanzadas
import matplotlib.pyplot as plt
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

▼ Importing data

```
# Define where you are running the code: colab or local
RunInColab = True # (False: no | True: yes)
```

```
# If running in colab:
```

```
if RunInColab:
```

```
    # Mount your google drive in google colab
```

```
    from google.colab import drive
```

```
    drive.mount('/content/drive')
```

```
    # Find location
```

```
    #!pwd
```

```
    #!ls
```

```
    #!ls "/content/drive/My Drive/Colab Notebooks/MachineLearningWithPython/"
```

```
    # Define path del proyecto
```

```
    Ruta = "/content/drive/My Drive/Colab Notebooks/MachineLearningWithPython/"
```

```
else:
```

```
    # Define path del proyecto
```

```
    Ruta = ""
```

```
    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
# url string that hosts our .csv file
```

```
url = "/content/drive/MyDrive/cartwheel.csv"
```

```
# Read the .csv file and store it as a pandas Data Frame
```

```
data = pd.read_csv(url)
data
```

	ID	Age	Gender	GenderGroup	Glasses	GlassesGroup	Height	Wingspan	CWDistance	Complete	CompleteGroup	Score
0	1	56.0	F	1	Y	1	62.00	61.0	79	Y	1.0	7
1	2	26.0	F	1	Y	1	62.00	60.0	70	Y	1.0	8
2	3	33.0	F	1	Y	1	66.00	64.0	85	Y	1.0	7
3	4	39.0	F	1	N	0	64.00	63.0	87	Y	1.0	10
4	5	27.0	M	2	N	0	73.00	75.0	72	N	0.0	4
5	6	24.0	M	2	N	0	75.00	71.0	81	N	0.0	3
6	7	28.0	M	2	N	0	75.00	76.0	107	Y	1.0	10
7	8	22.0	F	1	N	0	65.00	62.0	98	Y	1.0	9

Exploring the content of the data set

Get a general 'feel' of the data

```
data.head()
```

	ID	Age	Gender	GenderGroup	Glasses	GlassesGroup	Height	Wingspan	CWDistance	Complete	CompleteGroup	Score
0	1	56.0	F	1	Y	1	62.0	61.0	79	Y	1.0	7
1	2	26.0	F	1	Y	1	62.0	60.0	70	Y	1.0	8
2	3	33.0	F	1	Y	1	66.0	64.0	85	Y	1.0	7
3	4	39.0	F	1	N	0	64.0	63.0	87	Y	1.0	10
4	5	27.0	M	2	N	0	73.0	75.0	72	N	0.0	4


data.shape

```
(52, 12)
```

data.describe()

	ID	Age	GenderGroup	GlassesGroup	Height	Wingspan	CWDistance	CompleteGroup	Score
count	52.000000	51.000000	52.000000	52.000000	51.000000	51.000000	52.000000	51.000000	52.000000
mean	26.500000	28.411765	1.500000	0.500000	68.971569	67.313725	85.576923	0.843137	7.173077
std	15.154757	5.755611	0.504878	0.504878	5.303812	5.624021	14.353173	0.367290	2.211566
min	1.000000	22.000000	1.000000	0.000000	61.500000	57.500000	63.000000	0.000000	2.000000
25%	13.750000	25.000000	1.000000	0.000000	64.500000	63.000000	72.000000	1.000000	6.000000
50%	26.500000	27.000000	1.500000	0.500000	69.000000	66.000000	85.000000	1.000000	8.000000
75%	39.250000	30.000000	2.000000	1.000000	73.000000	72.000000	96.500000	1.000000	9.000000
max	52.000000	56.000000	2.000000	1.000000	79.500000	76.000000	115.000000	1.000000	10.000000

```
vars = ["Age", "Gender", "Glasses", "Height", "Wingspan", "CWDistance", "Complete", "Score"]
data2 = data[vars].dropna()
data2
```

	Age	Gender	Glasses	Height	Wingspan	CWDistance	Complete	Score	
0	56.0	F	Y	62.00	61.0	79	Y	7	
1	26.0	F	Y	62.00	60.0	70	Y	8	
2	33.0	F	Y	66.00	64.0	85	Y	7	
3	39.0	F	N	64.00	63.0	87	Y	10	
4	27.0	M	N	73.00	75.0	72	N	4	
5	24.0	M	N	75.00	71.0	81	N	3	
6	28.0	M	N	75.00	76.0	107	Y	10	
7	22.0	F	N	65.00	62.0	98	Y	9	
8	29.0	M	Y	74.00	73.0	106	N	5	
9	33.0	F	Y	63.00	60.0	65	Y	8	
10	30.0	M	Y	69.50	66.0	96	Y	6	
11	28.0	F	Y	62.75	58.0	79	Y	10	
12	25.0	F	Y	65.00	64.5	92	Y	6	
13	23.0	F	N	61.50	57.5	66	Y	4	
14	31.0	M	Y	73.00	74.0	72	Y	9	
15	26.0	M	Y	71.00	72.0	115	Y	6	
16	26.0	F	N	61.50	59.5	90	N	10	
17	27.0	M	N	66.00	66.0	74	Y	5	
18	23.0	M	Y	70.00	69.0	64	Y	3	
19	24.0	F	Y	68.00	66.0	85	Y	8	
20	23.0	M	Y	69.00	67.0	66	N	2	
21	29.0	M	N	71.00	70.0	101	Y	8	
22	25.0	M	N	70.00	68.0	82	Y	4	
23	26.0	M	N	69.00	71.0	63	Y	5	
24	23.0	F	Y	65.00	63.0	67	N	3	
25	28.0	M	N	75.00	76.0	111	Y	10	
26	24.0	M	N	78.40	71.0	92	Y	7	
27	25.0	M	Y	76.00	73.0	107	Y	8	
28	32.0	F	Y	63.00	60.0	75	Y	8	
29	38.0	F	Y	61.50	61.0	78	Y	7	
30	27.0	F	Y	62.00	60.0	72	Y	8	
31	33.0	F	Y	65.30	64.0	91	Y	7	
32	38.0	F	N	64.00	63.0	86	Y	10	
33	27.0	M	N	77.00	75.0	100	Y	8	
34	24.0	F	N	67.80	62.0	98	Y	9	
35	27.0	M	N	68.00	66.0	74	Y	5	
36	25.0	F	Y	65.00	64.5	92	Y	6	
37	26.0	F	N	61.50	59.5	90	Y	9	
38	31.0	M	Y	73.00	74.0	72	Y	9	
39	30.0	M	Y	69.50	66.0	96	Y	6	
40	23.0	F	N	70.40	71.0	66	Y	4	
41	26.0	M	Y	73.50	72.0	115	Y	6	

▼ Frequency tables

The `value_counts()` method can be used to determine the number of times that each distinct value of a variable occurs in a data set. In statistical terms, this is the "frequency distribution" of the variable. The `value_counts()` method produces a table with two columns. The first column contains all distinct observed values for the variable. The second column contains the number of times each of these values occurs. Note that the table returned by `value_counts()` is actually a **Pandas** data frame, so can be further processed using any Pandas methods for working with data frames.

```
# Number of times that each distinct value of a variable occurs in a data set
data2.Age.value_counts()
```

```
26.0    7
27.0    6
24.0    5
28.0    5
23.0    5
25.0    4
33.0    3
30.0    3
39.0    2
29.0    2
31.0    2
38.0    2
56.0    1
22.0    1
32.0    1
Name: Age, dtype: int64
```

```
# Proportion of each distinct value of a variable occurs in a data set PORCENTAJE%
x = data2.Age.value_counts()
x / x.sum()
```

```
26.0    0.142857
27.0    0.122449
24.0    0.102041
28.0    0.102041
23.0    0.102041
25.0    0.081633
33.0    0.061224
30.0    0.061224
39.0    0.040816
29.0    0.040816
31.0    0.040816
38.0    0.040816
56.0    0.020408
22.0    0.020408
32.0    0.020408
Name: Age, dtype: float64
```

Note that the `value_counts()` method excludes missing values. We confirm this below by adding up observations to your data frame with some missing values and then computing `value_counts()` and comparing this to the total number of rows in the data set, which is 28. This tells us that there are $28 - (21+6) = 1$ missing values for this variable (other variables may have different numbers of missing values).

```
# Total number of observations
Numberrows=data.shape[1]
Numberrows
```

```
# Total number of null observations
data.count()
```

```
# Total number of counts (excluding missing values)
dataEMV = data.dropna()
dataEMV.count()
```

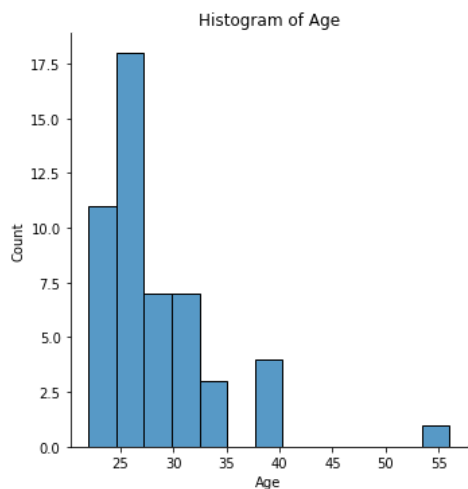
```
ID          49
Age          49
Gender       49
GenderGroup  49
Glasses      49
GlassesGroup 49
Height       49
Wingspan     49
CWDistance   49
```

```
Complete      49
CompleteGroup 49
Score         49
dtype: int64
```

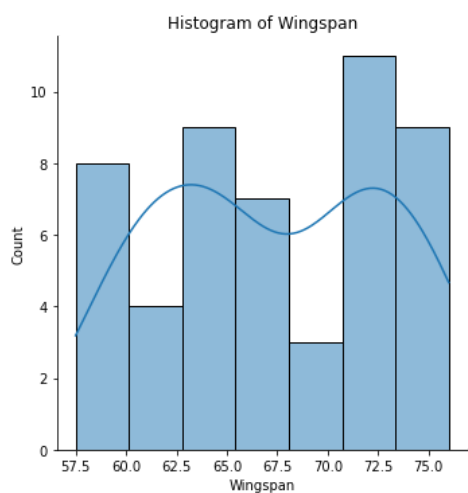
▼ Histogram

It is often good to get a feel for the shape of the distribution of the data.

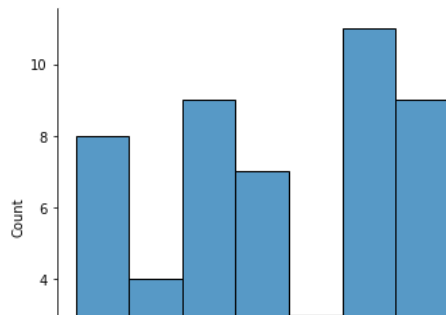
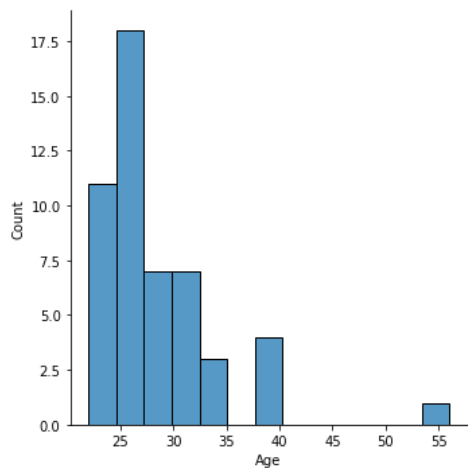
```
# Plot histogram of the total bill only
sns.displot(data["Age"], kde = False)
plt.title("Histogram of Age")
plt.show()
```



```
# Plot distribution of the tips only LA LINEA - MAYOR PROPABILIDAD
sns.displot(data["Wingspan"], kde = True)
plt.title("Histogram of Wingspan")
plt.show()
```



```
# Plot histogram of both the Age and the Wingspan
sns.displot(data["Age"], kde = False)
sns.displot(data["Wingspan"], kde = False)
plt.show()
```

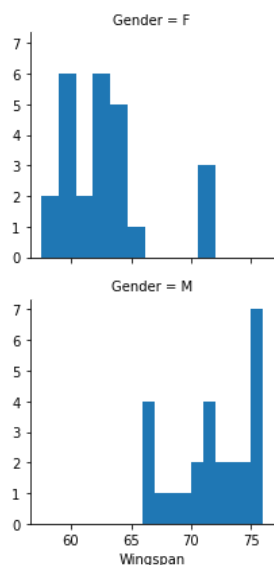


▼ Histograms plotted by groups

While looking at a single variable is interesting, it is often useful to see how a variable changes in response to another. Thus, we can create a histograms of one quantitative variable grouped by another categorical variables.

```
# Create histograms of the "Wingspan" grouped by "Gender" VARIABLE CATEGORICS GENERO
g = sns.FacetGrid(data, row = "Gender")
g = g.map(plt.hist, "Wingspan")

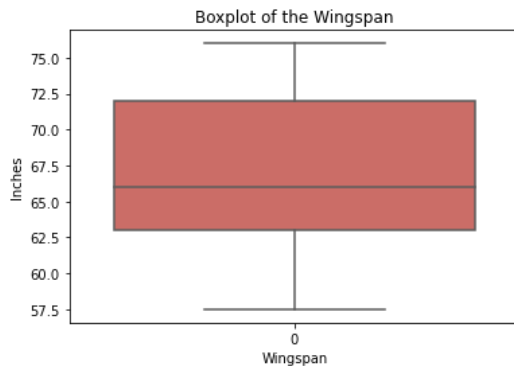
plt.show()
```



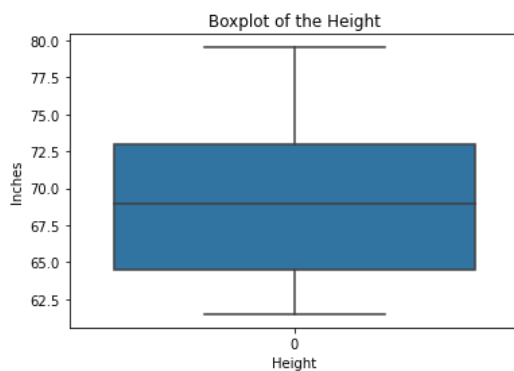
▼ Boxplots

Boxplots do not show the shape of the distribution, but they can give us a better idea about the center and spread of the distribution as well as any potential outliers that may exist. Boxplots and Histograms often complement each other and help an analyst get more information about the data

```
# Create the boxplot of the "total bill" amounts
sns.boxplot(data["Wingspan"], orient="v", palette="hls").set_title("Boxplot of the Wingspan")
plt.xlabel("Wingspan")
plt.ylabel("Inches")
plt.show()
```

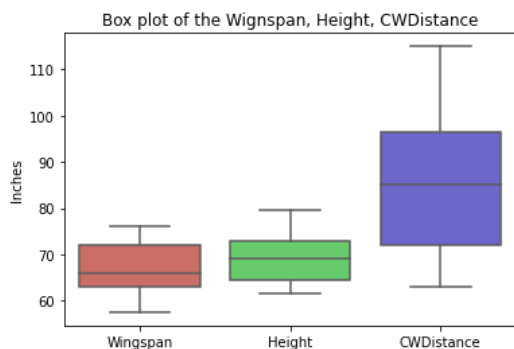


```
# Create the boxplot of "Height"
sns.boxplot(data["Height"], orient="v").set_title("Boxplot of the Height")
plt.xlabel("Height")
plt.ylabel("Inches")
plt.show()
```



```
# Create the boxplots of the "Wingspan" and of the "Height" amounts
x = data.loc[:, ["Wingspan", "Height", "CWDistance"]]
```

```
x2bp = sns.boxplot(data=x, orient="v", palette="hls")
x2bp.set_title("Box plot of the Wingspan, Height, CWDistance")
plt.ylabel("Inches")
plt.show()
```

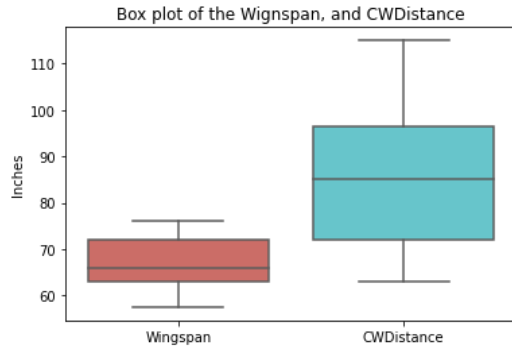


```
# Create the boxplots of the "Wingspan" and of "CWDistance"
y = data.loc[:, ["Wingspan", "CWDistance"]]

y2bp = sns.boxplot(data = y, orient = "v", palette = "hls")
```



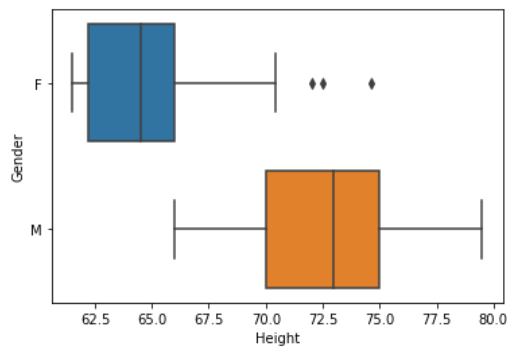
```
y2bp.set_title("Box plot of the Wignspan, and CWDistance")
plt.ylabel("Inches")
plt.show()
```



▼ Boxplots plotted by groups

While looking at a single variable is interesting, it is often useful to see how a variable changes in response to another. Thus, we can create a side-by-side boxplots of one quantitative variable grouped by another categorical variables.

```
# Create side-by-side boxplots of the "Height" grouped by "Gender"
sns.boxplot(x = data["Height"], y = data["Gender"])
plt.show()
```

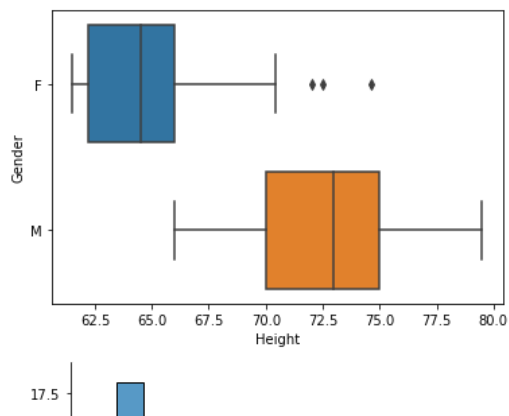


▼ Histograms and boxplots plotted by groups

We cal also create both boxplots and histograms of one quantitative variable grouped by another categorical variables

```
# Create a boxplot and histogram of "Height" grouped by "Gender"

sns.boxplot(x = data["Height"], y = data["Gender"])
sns.displot(data["Age"], kde = False)
plt.show()
```



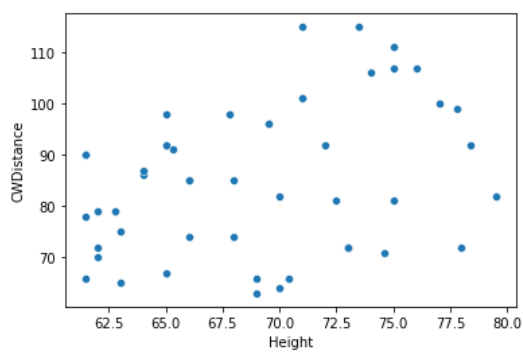
▼ Scatter plot

Plot values of one variable versus another variable to see how they are correlated



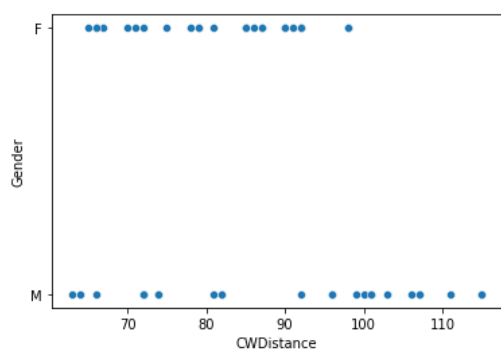
scatter plot between two variables

```
sns.scatterplot(data = data, y = "CWDistance",x= "Height",)
plt.show()
```



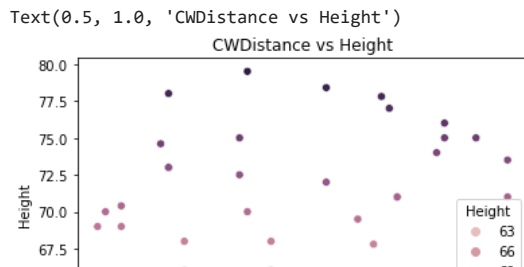
scatter plot between two variables (one categorical)

```
sns.scatterplot(data = data, x = "CWDDistance",y= "Gender",)
plt.show()
```



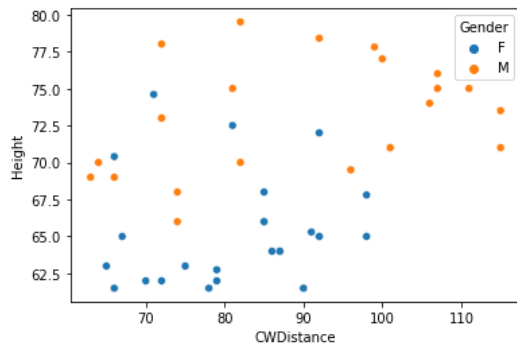
scatter plot between two variables (one categorical)

```
sns.scatterplot(data=data,x = "CWDistance",y = "Height", hue = data.Height)
plt.title("CWDistance vs Height")
```



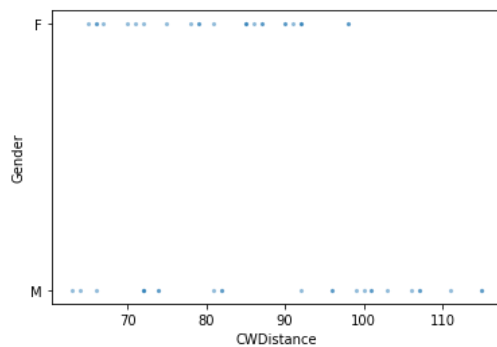
scatter plot between two variables grouped according to a categorical variable

```
sns.scatterplot(data=data, x = "CWDistance", y="Height", hue="Gender")
plt.show()
```



scatter plot between two variables grouped according to a categorical variable and with size of markers

```
plt.subplot(1,1,1)
sns.scatterplot(data=data, x="CWDistance", y="Gender", s=10, alpha = .5)
plt.show()
```



Final remarks

- Visualizing your data using **tables, histograms, boxplots, scatter plots** and other tools is essential to carry put analysis and extract conclusions
- There are several ways to do the same thing
- The **Seaborn** package provides visualisations tools that allow to explore data from a graphical perspective

▼ Activity: work with the iris dataset

Repeat this tutorial with the iris data set and respond to the following inquiries

1. Plot the histograms for each of the four quantitative variables
2. Plot the histograms for each of the quantitative variables
3. Plot the boxplots for each of the quantitative variables

4. Plot the boxplots of the petal width grouped by type of flower
5. Plot the boxplots of the setal length grouped by type of flower
6. Provide a description (explanation from your observations) of each of the quantitative variables

```
# url string that hosts our .csv file
url = "/content/drive/MyDrive/iris.csv"
# Read the .csv file and store it as a pandas Data Frame
dfI = pd.read_csv(url,header = None)
dfI
```

	0	1	2	3	4
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

```
dfI.shape
(150, 5)
```

```
dfI.describe()
```

	0	1	2	3
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
df2 = dfI.dropna()
df2
```

```
dfI = dfI.rename(columns={0: "Largo_Sepalo"})
dfI = dfI.rename(columns={1: "Ancho_Sepalo"})
dfI = dfI.rename(columns={2: "Largo_Petalo"})
dfI = dfI.rename(columns={3: "Ancho_Petalo"})
dfI = dfI.rename(columns={4: "Especie"})
```

```
dfI.head()
```

	Largo_Sepalo	Ancho_Sepalo	Largo_Petalo	Ancho_Petalo	Especie
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa

Number of times that each distinct value of a variable occurs in a data set
dfI.value_counts()

Largo_Sepalo	Ancho_Sepalo	Largo_Petalo	Ancho_Petalo	Especie	
5.8	2.7	5.1	1.9	Iris-virginica	2
6.2	2.2	4.5	1.5	Iris-versicolor	1
	2.9	4.3	1.3	Iris-versicolor	1
	3.4	5.4	2.3	Iris-virginica	1
6.3	2.3	4.4	1.3	Iris-versicolor	1
				..	
5.4	3.9	1.3	0.4	Iris-setosa	1
		1.7	0.4	Iris-setosa	1
5.5	2.3	4.0	1.3	Iris-versicolor	1
	2.4	3.7	1.0	Iris-versicolor	1
7.9	3.8	6.4	2.0	Iris-virginica	1

Length: 149, dtype: int64

Proportion of each distinct value of a variable occurs in a data set
dfI.value_counts(normalize=True).mul(100).round(1).astype(str) + '%'

Largo_Sepalo	Ancho_Sepalo	Largo_Petalo	Ancho_Petalo	Especie	
5.8	2.7	5.1	1.9	Iris-virginica	1.3%
6.2	2.2	4.5	1.5	Iris-versicolor	0.7%
	2.9	4.3	1.3	Iris-versicolor	0.7%
	3.4	5.4	2.3	Iris-virginica	0.7%
6.3	2.3	4.4	1.3	Iris-versicolor	0.7%
				...	
5.4	3.9	1.3	0.4	Iris-setosa	0.7%
		1.7	0.4	Iris-setosa	0.7%
5.5	2.3	4.0	1.3	Iris-versicolor	0.7%
	2.4	3.7	1.0	Iris-versicolor	0.7%
7.9	3.8	6.4	2.0	Iris-virginica	0.7%

Length: 149, dtype: object

Proportion of each distinct value of a variable occurs in a column
dfI.Especie.value_counts(normalize=True).mul(100).round(1).astype(str) + '%'

Iris-setosa	33.3%
Iris-versicolor	33.3%
Iris-virginica	33.3%

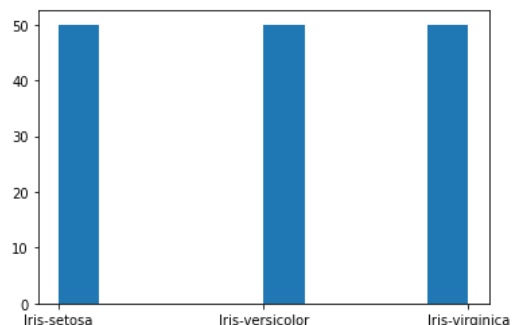
Name: Especie, dtype: object

HISTOGRAMS:

#Whole dataframe

plt.hist(dfI["Especie"])

plt.show()

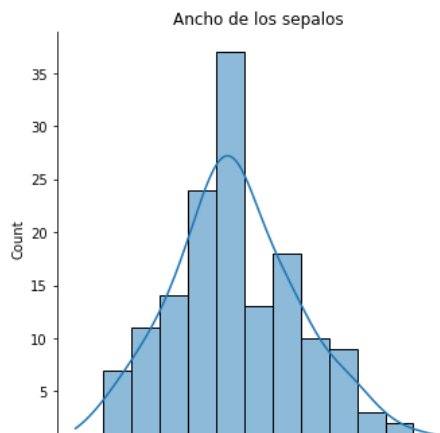


#Distribution

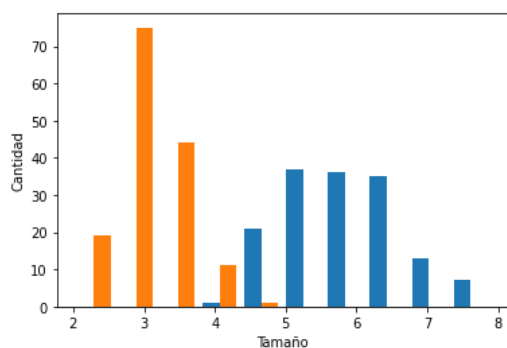
sns.displot(dfI["Ancho_Sepalo"], kde = True)

plt.title("Ancho de los sepalos")

plt.show()

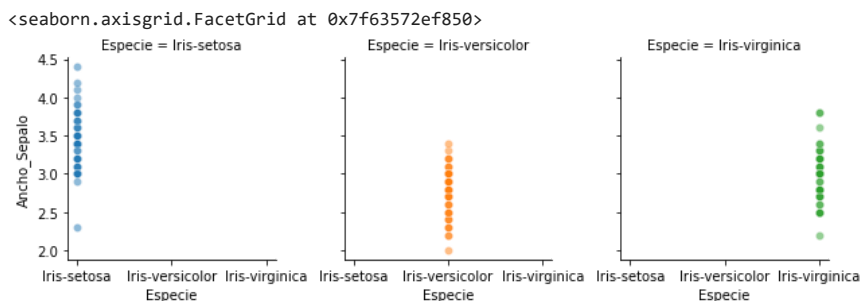


```
#Varias columnas
plt.hist([dfI["Largo_Sepalo"],dfI["Ancho_Sepalo"]])
plt.xlabel("Tamaño")
plt.ylabel("Cantidad")
plt.show()
```

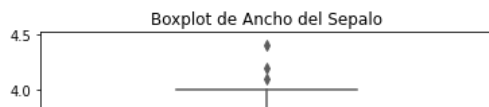


```
## Histograms plotted by groups:
sea = sns.FacetGrid(dfI, col = "Especie", hue = "Especie")

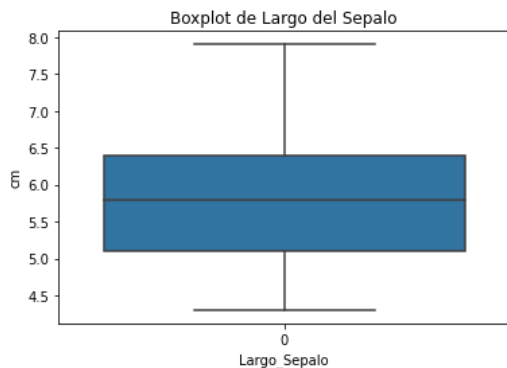
sea.map(sns.scatterplot, "Especie", "Ancho_Sepalo", alpha = .5)
```



```
## Boxplot:
# Create the boxplot of "Ancho_Sepalo"
sns.boxplot(dfI["Ancho_Sepalo"], orient="v", palette="hls").set_title("Boxplot de Ancho del Sepalo")
plt.xlabel("Ancho_Sepalo")
plt.ylabel("cm")
plt.show()
```

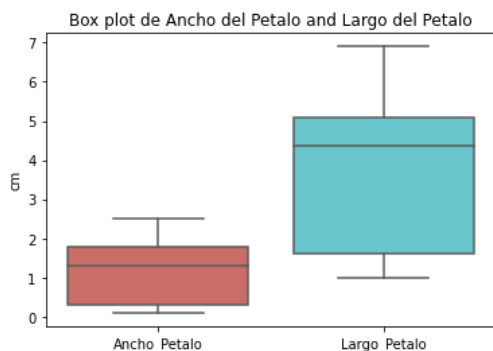


```
# Create the boxplot of "Largo_Sepalo"
sns.boxplot(dfI["Largo_Sepalo"], orient="v").set_title("Boxplot de Largo del Sepalo")
plt.xlabel("Largo_Sepalo")
plt.ylabel("cm")
plt.show()
```



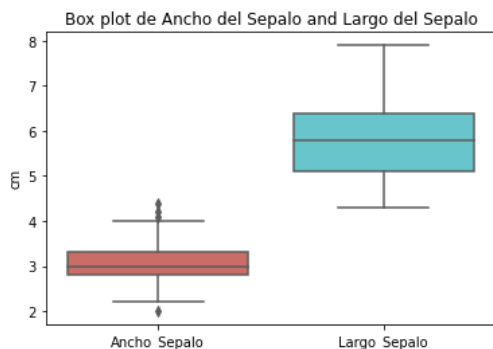
```
# Create the boxplots of the "Ancho_Petalo" and of "Largo_Petalo"
x = dfI.loc[:, ["Ancho_Petalo", "Largo_Petalo"]]

x2bp = sns.boxplot(data = x, orient = "v", palette = "hls")
x2bp.set_title("Box plot de Ancho del Petalo and Largo del Petalo")
plt.ylabel("cm")
plt.show()
```

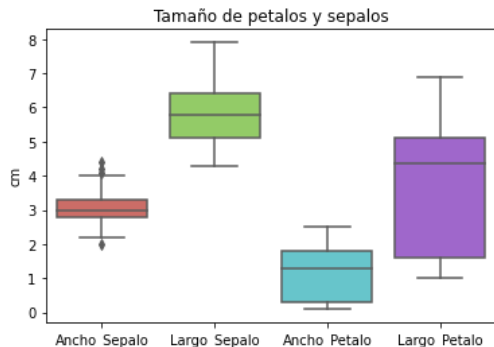


```
# Create the boxplots of the "Ancho_Sepalo" and of "Largo_Sepalo"
y = dfI.loc[:, ["Ancho_Sepalo", "Largo_Sepalo"]]

y2bp = sns.boxplot(data = y, orient = "v", palette = "hls")
y2bp.set_title("Box plot de Ancho del Sepalo and Largo del Sepalo")
plt.ylabel("cm")
plt.show()
```



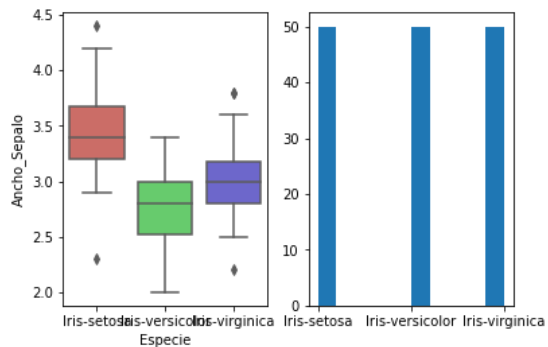
```
## Boxplot plotted by groups:
X = dfI.loc[:, ["Ancho_Sepalo", "Largo_Sepalo", "Ancho_Petalo", "Largo_Petalo"]]
x2bp = sns.boxplot(data=X, orient="v", palette="hls")
plt.ylabel("cm")
plt.title("Tamaño de petalos y sepalos")
plt.show()
```



```
## Histograms and boxplots plotted by groups:
fig, axs = plt.subplots(ncols = 2)
```

```
#X = dfI.loc[:, ["Especie", "Ancho_Sepalo"]]
sns.boxplot(x='Especie', y='Ancho_Sepalo', data=dfI, ax=axs[0], palette="hls")
plt.hist(dfI["Especie"])
```

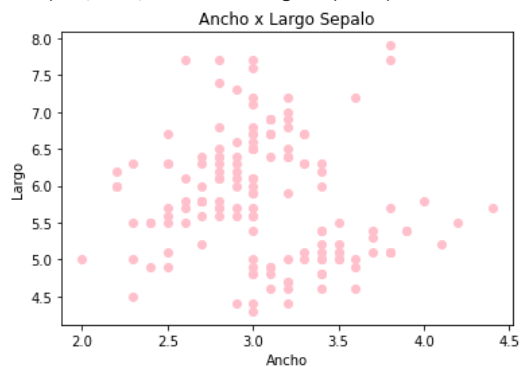
```
(array([50., 0., 0., 0., 0., 50., 0., 0., 0., 50.]),
 array([0., 0.2, 0.4, 0.6, 0.8, 1., 1.2, 1.4, 1.6, 1.8, 2. ]),
 <BarContainer object of 10 artists>)
```



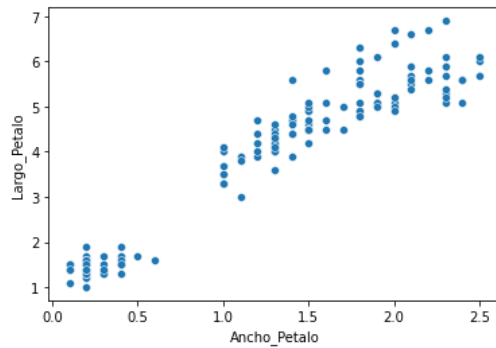
```
## Scatter plot:
# scatter plot between two variables
x = dfI.Ancho_Sepalo
y = dfI.Largo_Sepalo
```

```
plt.scatter(x, y, c="pink")
plt.xlabel("Ancho")
plt.ylabel("Largo")
plt.title("Ancho x Largo Sepalo")
```

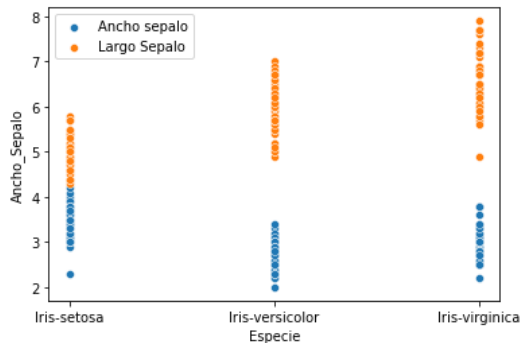
```
Text(0.5, 1.0, 'Ancho x Largo Sepalo')
```



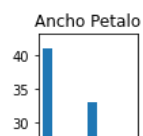
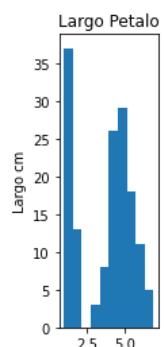
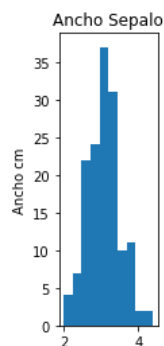
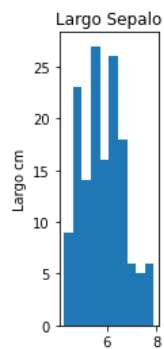

```
# scatter plot between two variables (one categorical)
sns.scatterplot(data = dfI, x = "Ancho_Petalo",y= "Largo_Petalo",)
plt.show()
```



```
#Agrupado categorico
plt.subplot(1,1,1)
sns.scatterplot(x="Especie",y="Ancho_Sepalo",data=dfI)
plt.subplot(1,1,1)
sns.scatterplot(x="Especie",y="Largo_Sepalo",data=dfI)
plt.legend(["Ancho sepalo", "Largo Sepalo"])
plt.show()
```

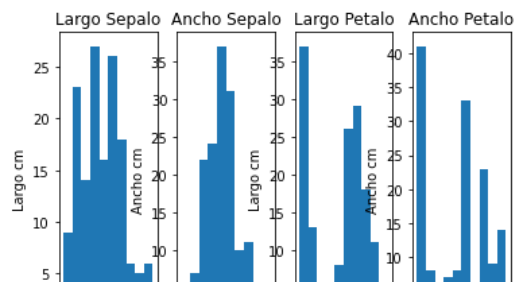


```
## ACTIVITY:
# 1: Plot the histograms for each of the four quantitative variables
plt.subplot(1,4,1)
plt.hist(dfI.Largo_Sepalo)
plt.title("Largo Sepalo")
plt.ylabel("Largo cm")
plt.show()
plt.subplot(1,4,2)
plt.hist(dfI.Ancho_Sepalo)
plt.title("Ancho Sepalo")
plt.ylabel("Ancho cm")
plt.show()
plt.subplot(1,4,3)
plt.hist(dfI.Largo_Petalo)
plt.title("Largo Petalo")
plt.ylabel("Largo cm")
plt.show()
plt.subplot(1,4,4)
plt.hist(dfI.Ancho_Petalo)
plt.title("Ancho Petalo")
plt.ylabel("Ancho cm")
plt.show()
```



```
# Largo Sepalo
plt.subplot(1,4,1)
plt.hist(dfI.Largo_Sepalo)
plt.title("Largo Sepalo")
plt.ylabel("Largo cm")
# Ancho Sepalo
plt.subplot(1,4,2)
plt.hist(dfI.Ancho_Sepalo)
plt.title("Ancho Sepalo")
plt.ylabel("Ancho cm")
# Largo Petalo
plt.subplot(1,4,3)
plt.hist(dfI.Largo_Petalo)
plt.title("Largo Petalo")
plt.ylabel("Largo cm")
# Ancho Petalo
plt.subplot(1,4,4)
plt.hist(dfI.Ancho_Petalo)
plt.title("Ancho Petalo")
plt.ylabel("Ancho cm")

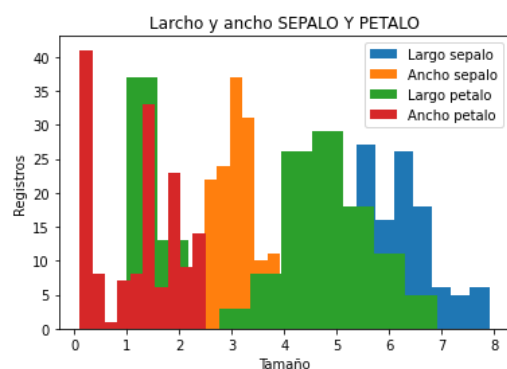
#Mostrar
plt.show()
```



Se puede observar que hay un registro mayor del largo del petalo (36cm) comparado con el mayor registro del largo del sepalo (27cm). En cuanto al ancho, se puede determinar con la ayuda de los histogramas que, el ancho del petalo tiene un registro mayor (41cm) a comparación del ancho del sepalo (36cm).

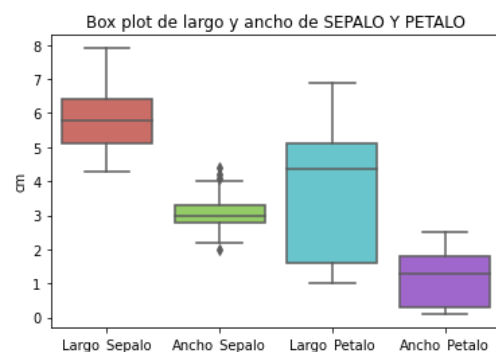
```
# 2: Plot the histograms for each of the quantitative variables
plt.hist(dfI.Largo_Sepalo)
plt.hist(dfI.Ancho_Sepalo)
plt.hist(dfI.Largo_Petalo)
plt.hist(dfI.Ancho_Petalo)

plt.ylabel("Registros")
plt.xlabel("Tamaño")
plt.title("Larcho y ancho SEPALO Y PETALO")
plt.legend(["Largo sepalo", "Ancho sepalo", "Largo petalo", "Ancho petalo"])
plt.show()
```



```
# 3: Plot the boxplots for each of the quantitative variables
x = dfI.loc[:, ["Largo_Sepalo", "Ancho_Sepalo", "Largo_Petalo", "Ancho_Petalo"]]

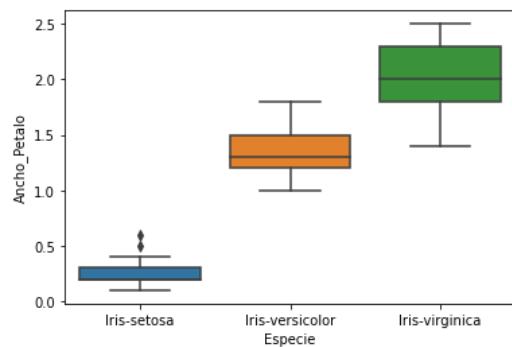
x2bp = sns.boxplot(data = x, orient = "v", palette="hls")
x2bp.set_title("Box plot de largo y ancho de SEPALO Y PETALO")
plt.ylabel("cm")
plt.show()
```



Podemos determinar que la variable con el registro menor es el ancho del petalo ya que va desde los 0.2 cm. Mientras que el que tiene el registro en cm de mayor valor es el largo del sepalo ya que llega hasta los 8 cm.

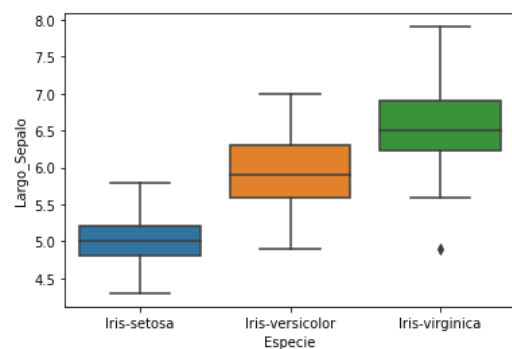
```
# 4: Plot the boxplots of the petal width grouped by type of flower
sns.boxplot(x='Especie',y='Ancho_Petalo', data=dfI)
```

```
plt.show()
```



La especie con el mayor ancho de petalo es iris virginica y el menor es iris setosa.

```
# 5: Plot the boxplots of the setal length grouped by type of flower
sns.boxplot(x='Especie',y='Largo_Sepalo', data=dfI)
plt.show()
```



La especie con el mayor largo de sepalo es iris virginica y el menor es iris setosa.

```
# 6: Provide a description (explanation from your observations) of each of the quantitative variables
# Largo Sepalo: Largo del sepalo de cada uno de los 3 tipos de flor (en cm)
# Ancho Sepalo: Ancho del sepalo de cada uno de los 3 tipos de flor (en cm)
# Largo Petalo: Largo del petalo de cada uno de los 3 tipos de flor (en cm)
# Ancho Petalo: Ancho del petalo de cada uno de los 3 tipos de flor (en cm)
```

✓ 0 s se ejecutó 18:53

● ✕