Data Analysis – Lab 2

M. Sébastien MASCHA

M.Pierre Sauvage

ISEP Paris - 17 Septembre 2019

M. Sublime

Lab 1 – Data Analysis

Import of libraries

This document has been done using python on Jupyter Notebook with the librairies:

- Numpy to manipulate arrays
- matplotlib to plot graphics
- pandas to import csv
- scipy for mathematicals usage
- maths for sqrt, pi, exp

In [38]:

```
# coding: utf-8

import data
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns; sns.set()
from math import sqrt,pi,exp
```

Exercice A - Multivariate data set : Fisher Iris

In this exercice, we study the Iris data set.

Question 1 - Open iris.csv as a matrice

We use the comma separator because we saw in the text editor that the data was separated by commas.

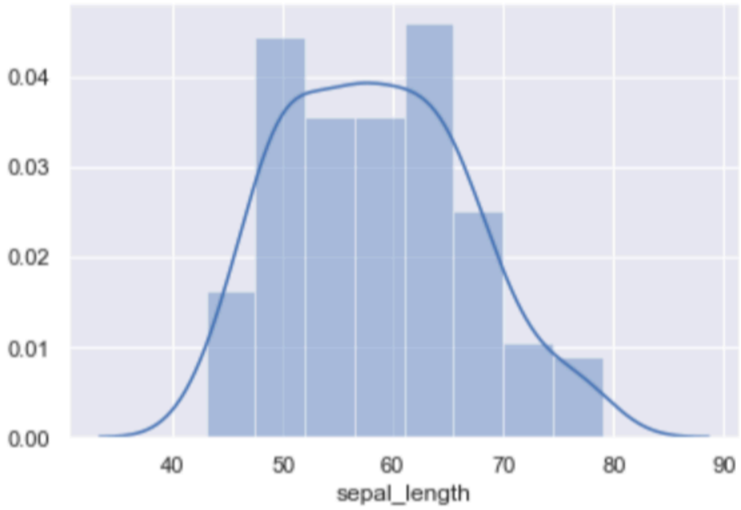
In [ ]:

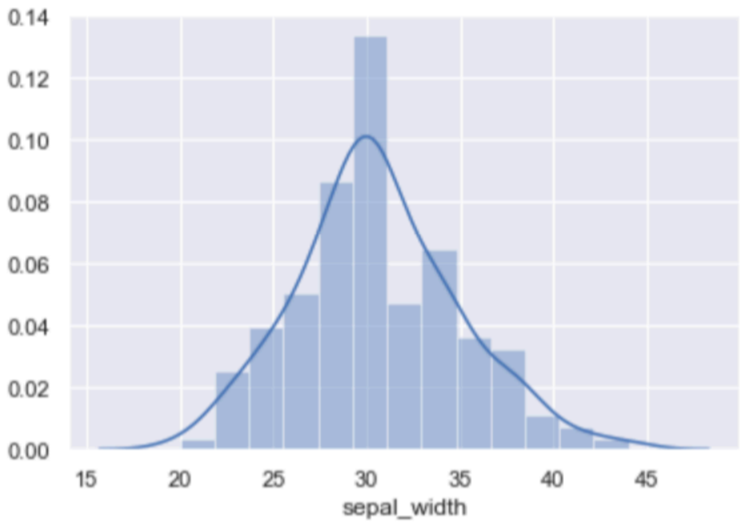
```
dataframe = pd.read_csv("data/iris.csv")
print(dataframe.shape)

dataframe.head()
dataframe.shape()
```

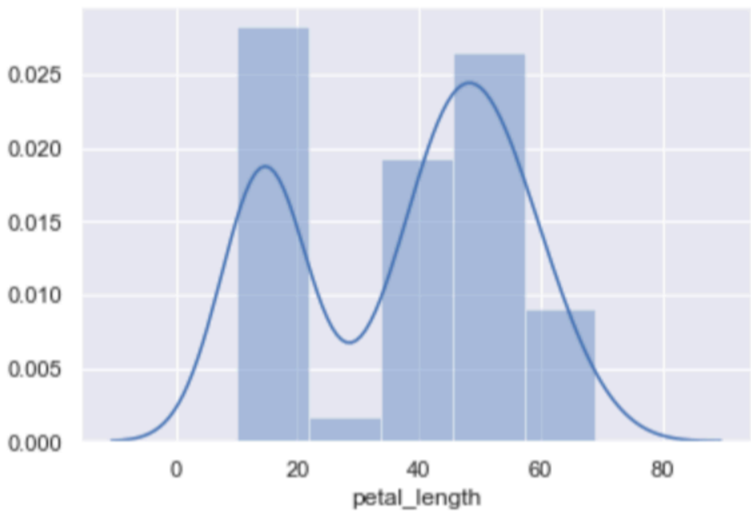
As we can see from the command dataframe.head() & dataframe.shape(), our dataset contains attributes on 150 flowers : their sepals length and width and the same measure for their petals

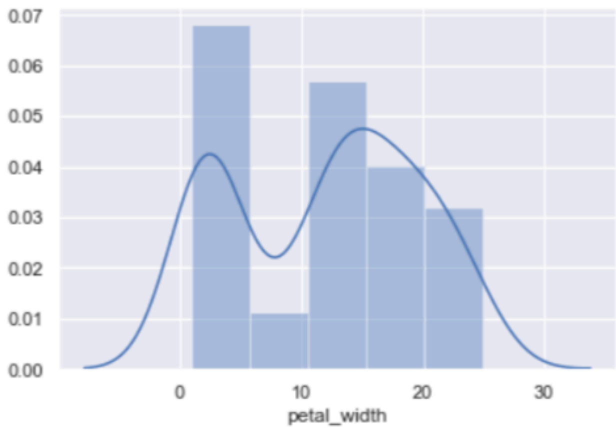
Question 2 - Display the histograms of the different attributes.

You may use the displot function from the seaborn library.


0.04
0.03
0.02
0.01
0.00
40
50
60
70
80
90
sepal_length


0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
15
20
25
30
35
40
45
sepal_width


0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
petal_length



What can you say about their distributions ?

- We cannot say anything clearly about the distribution of the first attribute.
- The second one follows approximately a normal distribution with a mode of 30.
- The third and the forth one are bimodal.

Question 3 - Compute the coefficient of correlation between all attributes

In [46]:

```
def cov(a, b):
    if len(a) != len(b):
        return "Error : the two vectors should have an equal size."
    return np.sum((a-np.mean(a))*(b-np.mean(b)))/(len(a)-1)
```

In [62]:

```
covMatrix = np.zeros((4, 4))
corrMatrix = np.zeros((4, 4))

for i in range(dataframe.shape[1]):
    for j in range(dataframe.shape[1]):
        covMatrix[i][j] = cov(dataframe.values[: , i],dataframe.values[: , j])
        corrMatrix[i][j]= covMatrix[i][j]/(np.std(dataframe.values[: , i]) * np.
std(dataframe.values[: , j]))
print(covMatrix)
print(corrMatrix)
```

```
# Checking values :
dataframe.cov()
```

Out[60]:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	68.569351	-4.243400	127.431544	51.627069
sepal_width	-4.243400	18.997942	-32.965638	-12.163937
petal_length	127.431544	-32.965638	311.627785	129.560940
petal_width	51.627069	-12.163937	129.560940	58.100626

In [61]:

```
dataframe.corr()
```

Out[61]:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.117570	0.871754	0.817941
sepal_width	-0.117570	1.000000	-0.428440	-0.366126
petal_length	0.871754	-0.428440	1.000000	0.962865
petal_width	0.817941	-0.366126	0.962865	1.000000

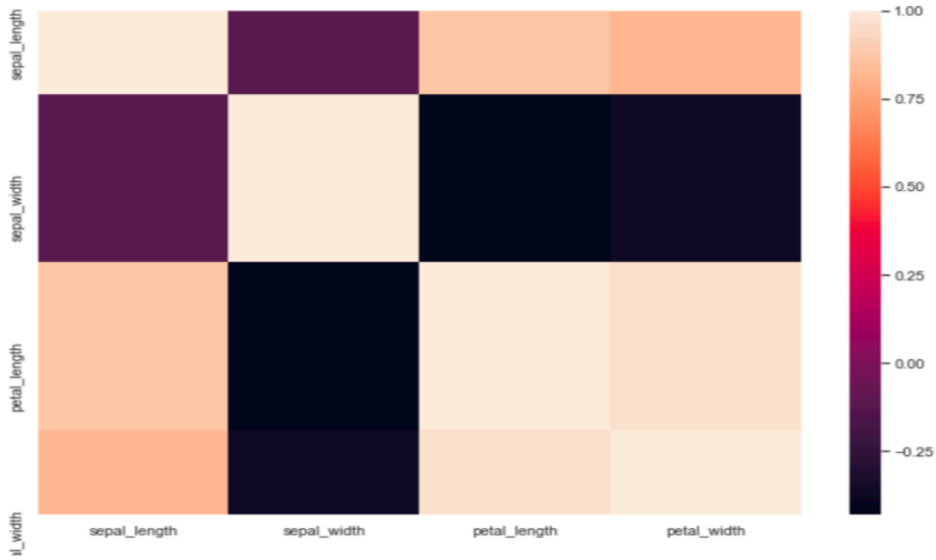
We check the result with the made-in function from Pandas library, and yes the results are correct. Let's visualize our data before going into a deeper interpretation

Question 4 - Visualize the correlation between the different variables.

```
df_corr = dataframe.corr()
figure,axe = plt.subplots(figsize=(12,8))
sns.heatmap(df_corr)
```

Out[68]:

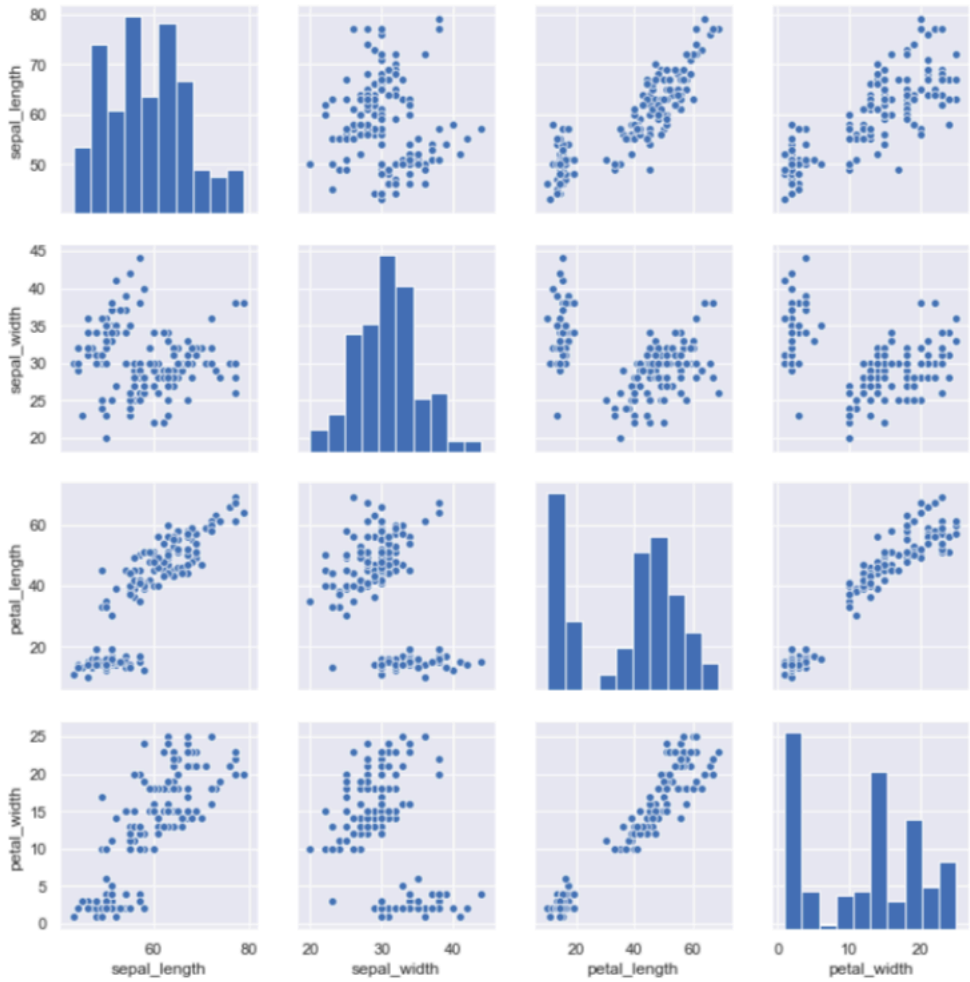
<matplotlib.axes._subplots.AxesSubplot at 0x1a2030d6d8>



```
sns.pairplot(dataframe)
```

Out[17]:

<seaborn.axisgrid.PairGrid at 0x1a1f078ac8>



Comment your results :

We used to kind of visualisation, to have a better understanding of our data The results are the same using the Python function. Using plot, we can see that we have the same intuition looking at :

- Graph 1-3
- Graph 1-4
- Graph 3-4

The petal lengths and widths have a good correlation between them, and it's the same for the sepals lengths and widths. If we have correlation between those attributes we could classify flowers depending on their petals and sepals

Question 5 - Compute the confidence intervals for the correlation coefficients

We will suppose that the attributes are following a normal distribution

In [93]:

```
def IC95(a,b):
    r = cov(a,b) / (np.std(a) * np.std(b))
    Z = np.log(abs(1 + r)) - np.log(abs(1 - r))/2
    sz = sqrt(1/(len(a) - 3))
    Zinf = Z - 1.96*sz
    Zsup = Z + 1.96*sz
    ic_interval = [(exp(2*Zinf) - 1)/(exp(2*Zinf) + 1), (exp(2*Zsup) - 1)/(exp(2
*Zsup) + 1)]
    return ic_interval

# Calcul des 16 intervalles de confiance à 95% :
for i in range(dataframe.shape[1]):
    for j in range(dataframe.shape[1]):
        print(IC95(dataframe.values[:,i],dataframe.values[:,j]))
```

We can confirm that we got good result because our correlation coefficient are well computed : they each sit between their distinct inferior and superior bounds.

We observe that two variables with a low coefficient of correlation like the sepal length and the sepal width, r12 = --0.1093692 , the confidence interval is wider.

The span of Confidence Interval (95%) varies from 0.32 to 0.07 when considering the high correlation between the sepal length and the petal length.

Wider the IC95 is, highest the correlation is.

Exercice B - Multivariate data set : Anthropometric data

In this exercice, we study the "mansize" data set. These data described anthropo- metric features acquired in a famous medicine University based on a population of Bachelor students.

Question 1 - Open mansize.csv as a matrice

We use the semicolon separator because we saw in the text editor that the data was separated by semicolon.

In [4]:

```
dataframe = pd.read_csv("data/mansize.csv", sep=';')
print(dataframe.shape)
```

```
(161, 9)
```

Question 2 - Apply the function describe(·) to your data set. What does this function do ? Comment the results on your data.

Lab 1 – Data Analysis

```
print(dataframe.describe())
```

```
                Age  Height (cm)  Weight (kg)  Femur Length (cm)  \
count  161.000000   161.000000   161.000000         161.000000   
mean    20.447205   173.223602    73.357143          47.516149   
std      1.676681    12.346546    14.160746           5.210949   
min     18.000000   150.000000    40.000000          37.100000   
25%     19.000000   165.000000    63.100000          43.600000   
50%     20.000000   172.000000    71.500000          47.400000   
75%     22.000000   181.000000    81.100000          51.300000   
max     24.000000   203.000000   115.200000          62.100000   

       Feet Size (cm)  Arm span (cm)  Hand length (cm)  Cranial volu
me (cm3)  \
count      161.000000     161.000000        161.000000            16
1.000000   
mean        24.967702     183.040994         18.885093           141
8.105590   
std          2.703530       8.989101          1.247258             4
9.010535   
min         18.900000     159.600000         15.800000           129
8.000000   
25%         23.100000     176.300000         18.200000           138
2.000000   
50%         25.100000     181.700000         18.900000           141
8.000000   
75%         26.700000     188.900000         19.800000           145
0.000000   
max         32.200000     206.900000         22.600000           155
8.000000   

       Penis size (cm)  
count       161.000000  
mean         13.394410  
std           1.481986  
min           9.100000  
25%          12.500000  
50%          13.400000  
75%          14.300000  
max          18.400000  
```

Function pandas.DataFrame.describe :

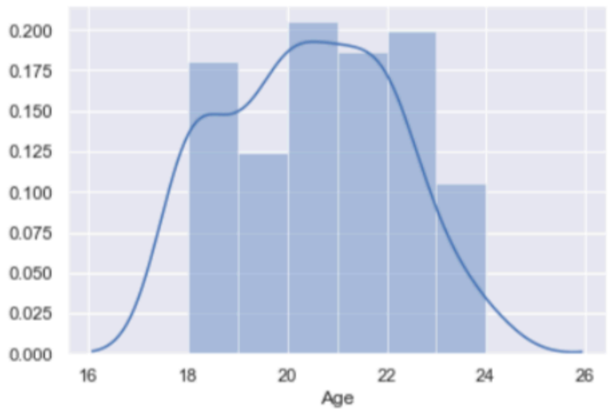
- Generate descriptive statistics that summarize the central tendency, dispersion and shape of a dataset's distribution, excluding NaN values.
- Analyzes both numeric and object series, as well as DataFrame column sets of mixed data types. The output will vary depending on what is provided. Refer to the notes below for more detail.

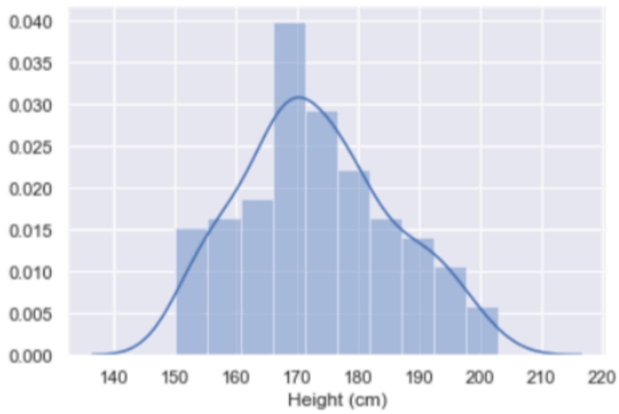
About mansize.csv data ?

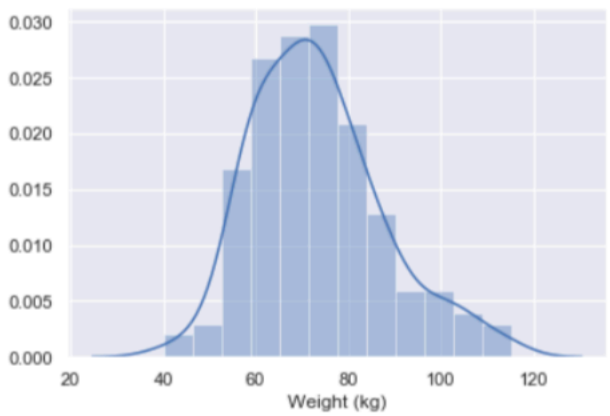
- We observe that most of the variables have an average value close to the median and the 1st and 3rd quartiles are almost symmetrical around the mean. This means that the data set is quite homogenous and that most of these variables follow a normal distribution.

Lab 1 – Data Analysis

For the first three :







Lab 1 – Data Analysis

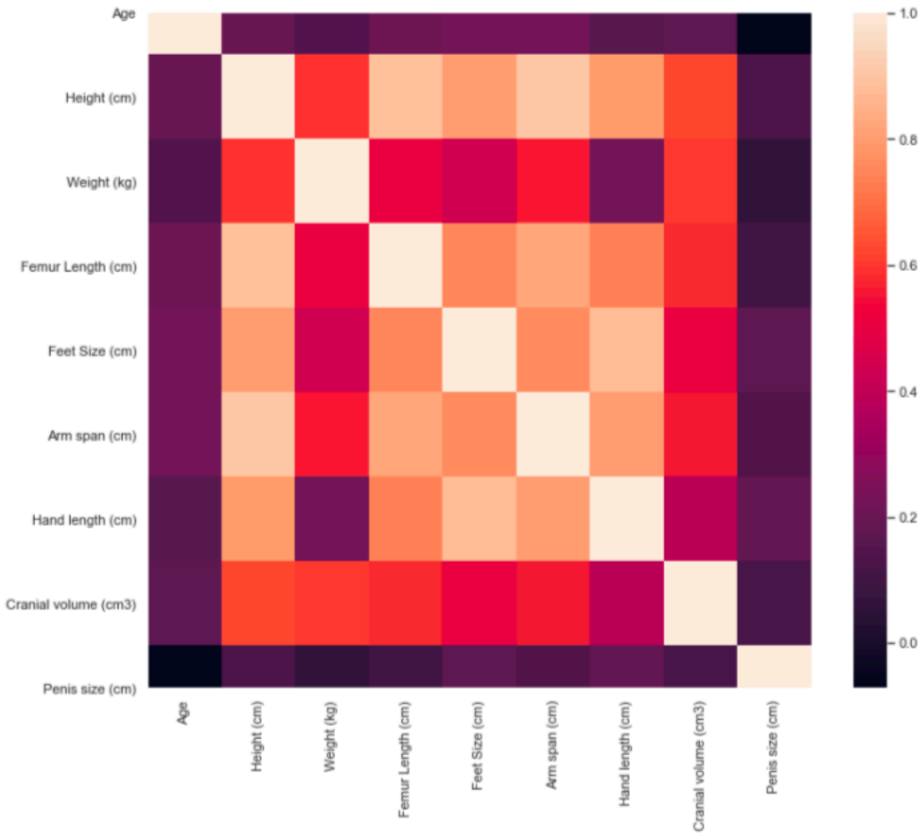
Question 4 - Use the commands corr(), subplots(), heatmap() and pairplot() to visualize the correlation between the different variables. Comment your results. In particular, what can you say about the use in archaeology of the femur length to predict the height of an individual ?

In [34]:

```
df_corr = dataframe.corr()
figure,axe = plt.subplots(figsize=(12,10))
sns.heatmap(df_corr)
```

Out[34]:

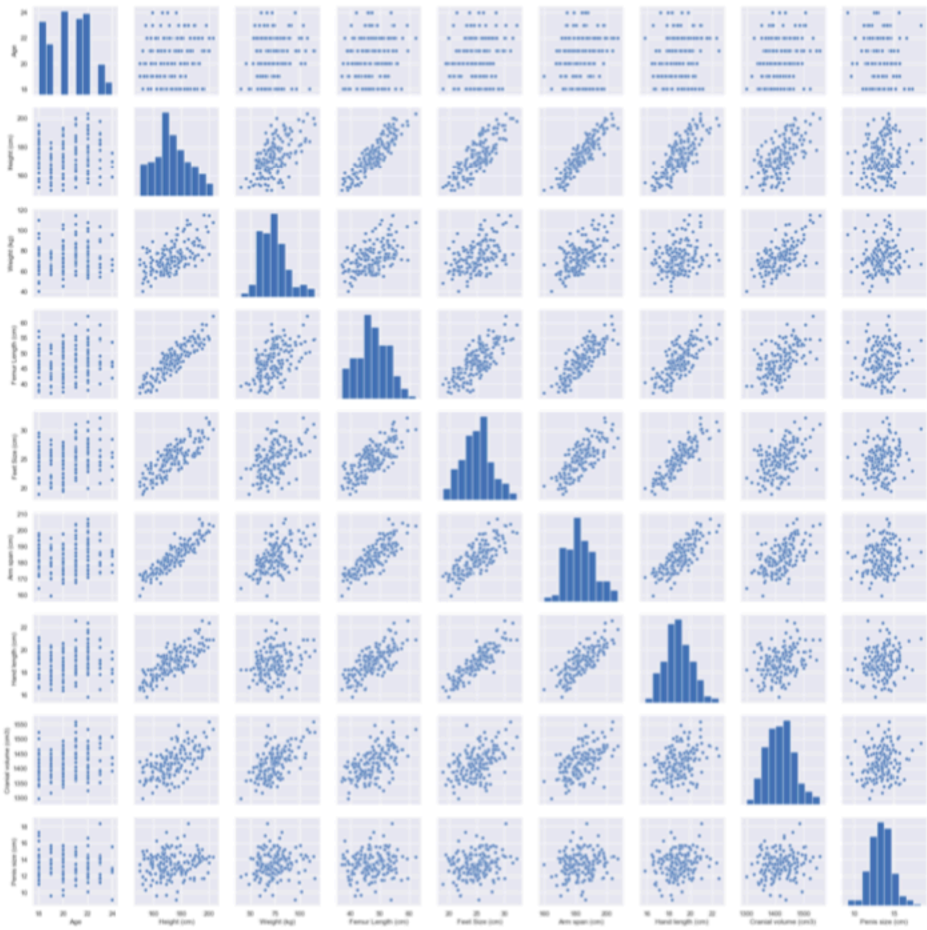
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a2d2d9748>
```



```
sns.pairplot(dataframe)
```

Out[26]:

```
<seaborn.axisgrid.PairGrid at 0x1a273495c0>
```



```
def IC95(a,b):
    r = pearsonr(a,b)[0]
    Z = np.log(abs(1 + r)) - np.log(abs(1 - r))/2
    sz = sqrt(1/(len(a) - 3))
    Zinf = Z - 1.96*sz
    Zsup = Z + 1.96*sz
    ic_interval = [(exp(2*Zinf) - 1)/(exp(2*Zinf) + 1), (exp(2*Zsup) - 1)/(exp(2
*Zsup) + 1)]
    return ic_interval

# Calcul des 16 intervalles de confiance à 95% :
for i in range(dataframe.shape[1]):
    for j in range(dataframe.shape[1]):
        print(IC95(dataframe.values[:,i],dataframe.values[:,j]))
```

It appears that there is a strong link between the height and the length of the femur as well as with the span of the arm which have a coefficient of determination of around 0.8.

There is also a 0.64 and 0.62 coefficient of determination between the height and the size of the feet as well as with the length of the hand, making them less strongly linked.

We observe about a 0.55 determination between the length of the femur and the size of the feet as well as with the length of the hand. It shows are strong they're linked.

Furthermore, penis' size is definitely not linked with attributes provided.

Exercice C - Chi-squared test of independence and categorial variables

In this exercice, we study the Weather data set.

Question 1 - Open weather.csv

In [3]:

```
dataframe = pd.read_csv("data/weather.csv",sep=';')
print(dataframe.shape)

dataframe.head()
```

```
(193, 4)
```

Out[3]:

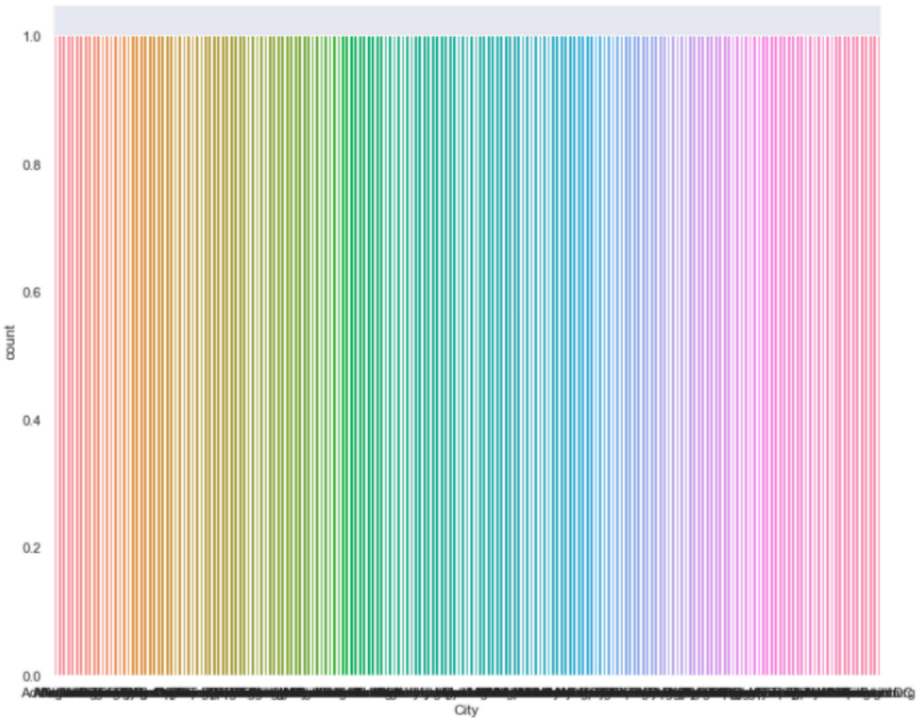
	City	Outlook	Humidity	Temperature
0	Abidjan	Rainy	High	Hot
1	Addis-Abeba	Rainy	Average	Mild
2	Algiers	Overcast	Average	Mild
3	Amsterdam	Sunny	Average	Mild
4	Anchorage	Sunny	Average	Cold

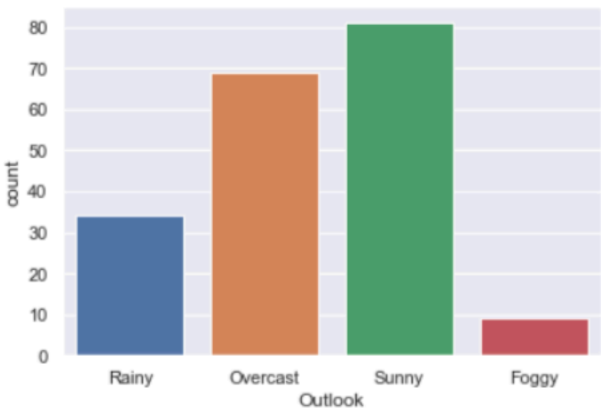
In [4]:

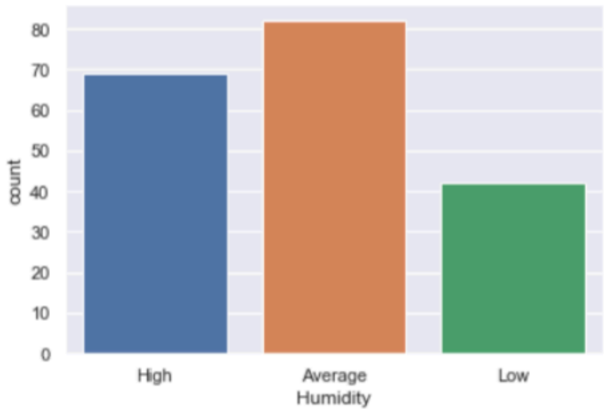
```
dataframe.describe()
```

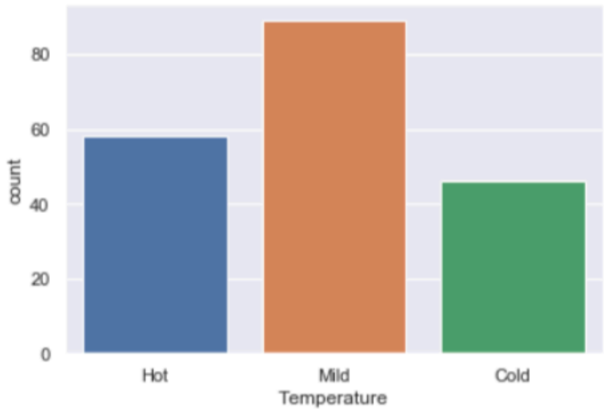
Out[4]:

	City	Outlook	Humidity	Temperature
count	193	193	193	193
unique	193	4	3	3
top	Lhasa	Sunny	Average	Mild
freq	1	81	82	89


1.0
0.8
0.6
0.4
0.2
0.0
count
City


80
70
60
50
40
30
20
10
0
count
Rainy
Overcast
Sunny
Foggy
Outlook





We have three categorical variables that describe the outlook, the temperature and the humidity in different cities around the world.

The outlook can be : foggy, overcast, rainy or sunny.

The temperature can be : cold, mild or hot.

The humidity can be : low, average or high.

Question 2 - Create the contingency table between the variables “outlook” and “temperature"

```
pd.crosstab(dataframe['Outlook'], dataframe['Temperature'],  margins = True)
```

Out[6]:

Temperature	Cold	Hot	Mild	All
Outlook				
Foggy	4	3	2	9
Overcast	19	14	36	69
Rainy	6	11	17	34
Sunny	17	30	34	81
All	46	58	89	193

Comments

More measures were done in situations of overcast and sunny outlooks. If the data collected is accurate, foggy days or rainy days are rare. Therefore we have more exhaustive data for the corresponding temperatures in these situations.

The outlook data has a span of 4 and the temperature data takes 3 different values.

We consider the following formula to calculate the degrees of freedom of this problem :

Degrees of freedom = (r – 1)(c – 1)

Degrees of freedom = (4-1)(3-1) = 6

Question 3 - Use the command chi2 contingency(·)

In [7]:

```
contingency_table_outlook_temperature = pd.crosstab(dataframe['Outlook'], datafr
ame['Temperature'],  margins = False)
```

```
print('This function computes the chi-square statistic and p-value for the hypot
hesis test of independence of the observed frequencies in the contingency table
 observed. \n')
stats.chi2_contingency(contingency_table_outlook_temperature)
```

```
This function computes the chi-square statistic and p-value for the
hypothesis test of independence of the observed frequencies in the c
ontingency table observed.
```

Out[8]:

```
(8.493304645048728,
 0.2041427335538749,
 6,
 array([[ 2.14507772,  2.70466321,  4.15025907],
        [16.44559585, 20.7357513 , 31.81865285],
        [ 8.10362694, 10.21761658, 15.67875648],
        [19.30569948, 24.34196891, 37.35233161]]))
```

Comments :

Using the chi2_contingency() command we obtain the value of Chi square : X^2 = 8,493 .

We also get the p-value : p = 0,2041 .

As the p value is way higher than 0.05 we cannot reject the independence hypothesis between the outlook and the temperature.

Question 4 - Assess whether there is a link between the other variables

i.e. : Outlook/humidity, temperature/humidity

Dependency between the outlook and the humidity :

At first we generate the contingency table of these two variables :

In [9]:

```
contingency_table_outlook_humidity = pd.crosstab(dataframe['Outlook'], dataframe
['Humidity'],  margins = False)
print(contingency_table_outlook_humidity)
```

```
Humidity  Average  High  Low
Outlook
Foggy           3     6    0
Overcast       34    30    5
Rainy          10    24    0
Sunny          35     9   37
```

Then we proceed to compute the p value and the chi square for this table using the chi2_contingency() command.

In [10]:

```
stats.chi2_contingency(contingency_table_outlook_humidity)
```

Out[10]:

```
(68.48971179065708,
 8.339864804368143e-13,
 6,
 array([[ 3.8238342 ,  3.21761658,  1.95854922],
        [29.31606218, 24.66839378, 15.01554404],
        [14.44559585, 12.15544041,  7.39896373],
        [34.41450777, 28.95854922, 17.62694301]]))
```

Using the chi2_contingency() command we obtain the value of Chi square : X^2 = 68,490 .

We also get the p-value : p = 8,340 .

As the p value is very close to zero we can reject the independence hypothesis on these two variables. It is therefore relevant to compute the value of a contingency coefficient.

As the two variables do not have the same span, we chose to compute the Cramer contingency coefficient for these variables : ρ = sqrt(x^^2 /N min((c−1)(r−1)))

We obtain Cramer's V = 0.42132306 We cannot conclude that the two variables are dependant as almost 60% of the data's contingency cannot be explained.

Dependency between the temperature and the humidity :

At first we generate the contingency table of these two variables :

In [13]:

```
contingency_table_temp_humidity = pd.crosstab(dataframe['Temperature'], dataframe['Humidity'], margins = False)
print(contingency_table_temp_humidity)
```

```
Humidity     Average  High  Low
Temperature
Cold              20    22    4
Hot               24    15   19
Mild              38    32   19
```

Then we proceed to compute the p value and the chi square for this table using the chi2_contingency() command.

```
stats.chi2_contingency(contingency_table_temp_humidity)
```

Out[14]:

```
(10.330736992441599,
 0.03521016044258168,
 4,
 array([[19.54404145, 16.44559585, 10.01036269],
        [24.64248705, 20.7357513 , 12.62176166],
        [37.8134715 , 31.81865285, 19.36787565]]))
```

Using the chi2_contingency() command we obtain the value of Chi square : X^2 = 10.331 .

We also get the p-value : p = 0,0352 .

As the p value is very close to zero we can reject the independence hypothesis on these two variables. It is therefore relevant to compute the value of a contingency coefficient.

As the two variables have the same span, we chose to compute the Chuprov contingency coefficient for these variables :

p = sqrt((x^^2)/(N*sqrt((c-1)(r-1)))

We have ptemphum = 0.1635978

The Chuprov coefficient is close to zero, we can therefore conclude that the two variables are independent.