TP2 Analyse de donn?es - Matthieu Rousseau Antoine Marvier

October 16, 2018

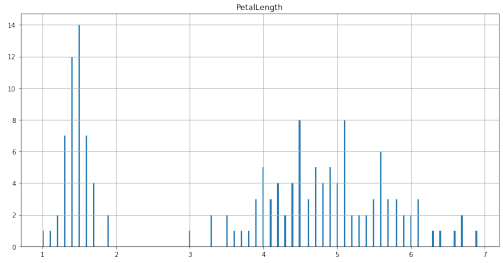
1 Exercice 1

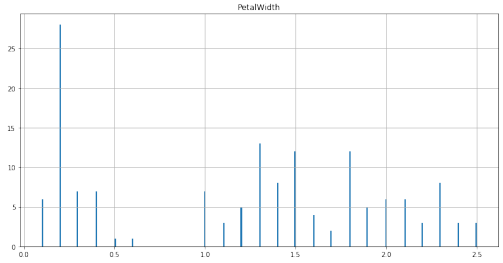
```
In [20]: import pandas as pd
         import matplotlib.pyplot as plt
         import numpy as np
```

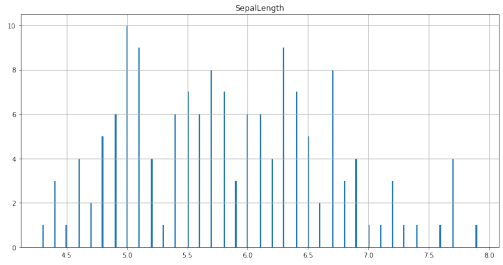
```
In [14]: df = pd.read_csv("iris.csv")
         df.head()
         SepalLength=df['SepalLength']
         SepalWidth=df['SepalWidth']
         PetalLength=df['PetalLength']
         PetalWidth=df['PetalWidth']
         Class=df['Class']
```

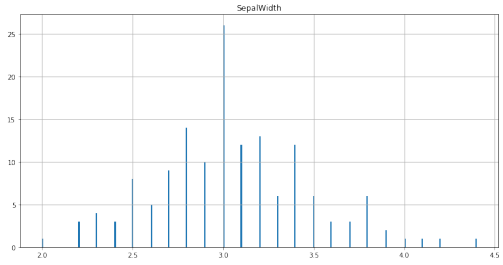
```
In [60]: df.hist(figsize=(30,15),bins=300)
```

```
Out[60]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000002358D70ACC0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002358E0F5B38>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x000002358E5E8860>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000002358E5DADD8>]],
               dtype=object)
```









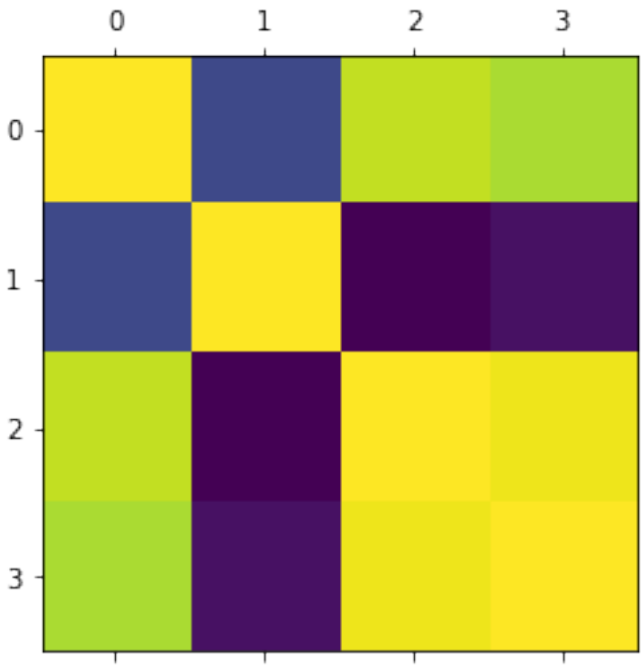
Description de la distribution...
Correlation entre les différents attributs :
SepalLenght et SepalWidth
SepalLenght et PetalLenght
SepalLenght et PetalWidth
SepalLenght et Class
SepalWidth et PetalLenght
SepalWidth et PetalWidth
SepalWidth et Class
PetalLength et PetalWidth
PetalLength et Class
PetalWidth et class

```
In [42]: def is_correlated(A,B):

             meanA = np.mean(A)
             meanB=np.mean(B)
             stdA = np.std(A)
             stdB=np.std(B)
             covariance =  (1/(len(A)-1))*sum(np.subtract(A,meanA)*np.subtract(B,meanB))

             correlation = covariance / (stdA*stdB)
             return correlation
```

```
In [43]: print("Coefficient de correlation ",is_correlated(SepalLength,SepalWidth))

         corr_matrix = df.corr()
         plt.matshow(corr_matrix)

         #Calculer le coefficient de corrélation pour les autres valeurs...
         #Détailler le schéma ci-dessous
```

```
Coefficient de correlation  -0.11010327176239866
```

```
Out[43]: <matplotlib.image.AxesImage at 0x235fae97ba8>
```




```
In [44]: def confidence_interval(A,B):
             if len(A) == len(B):
                 n=len(A)
             else:
                 return "error"
             r = is_correlated(A,B)
             Z = (np.log(1+r)-np.log(1-r))/2
             sz = np.sqrt(1/(n-3))
             Zinf = Z-1.96*sz
             Zsup = Z+1.96*sz
             icinf,icsup = ((np.exp(2*Zinf)-1)/(np.exp(2*Zinf)+1)),((np.exp(2*Zsup)-1)/(np.exp(2

             return icinf,icsup


In [45]: print(confidence_interval(SepalLength,SepalWidth))
         #Confidence intervalle pour chaque attribut + commentaires

(-0.265679606306267, 0.05106217325847706)
```

2 Exercice 2

```
In [48]: df_mansize = pd.read_csv("mansize.csv",sep=";")
```

```
In [52]: df_mansize.head()
```

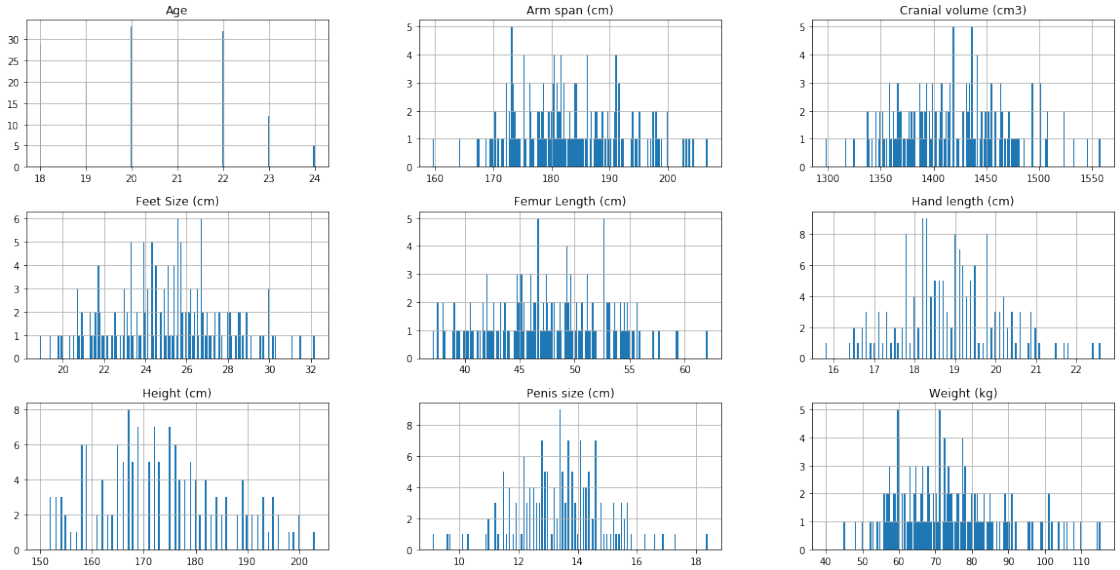
Out[52]:

	Age	Height (cm)	Weight (kg)	Femur Length (cm)	Feet Size (cm)	\
0	21	195	71.0	59.4	30.0	
1	21	184	82.4	54.3	24.3	
2	18	169	96.7	45.1	21.5	
3	21	166	68.2	42.4	21.3	
4	18	175	56.5	46.9	24.9	

	Arm span (cm)	Hand length (cm)	Cranial volume (cm3)	Penis size (cm)
0	203.2	22.6	1442	11.7
1	192.1	18.6	1366	12.8
2	176.2	16.6	1436	13.8
3	181.6	18.1	1375	14.8
4	183.9	19.1	1376	13.4

```
In [62]: df_mansize.hist(figsize=(20,10),bins=200)
```

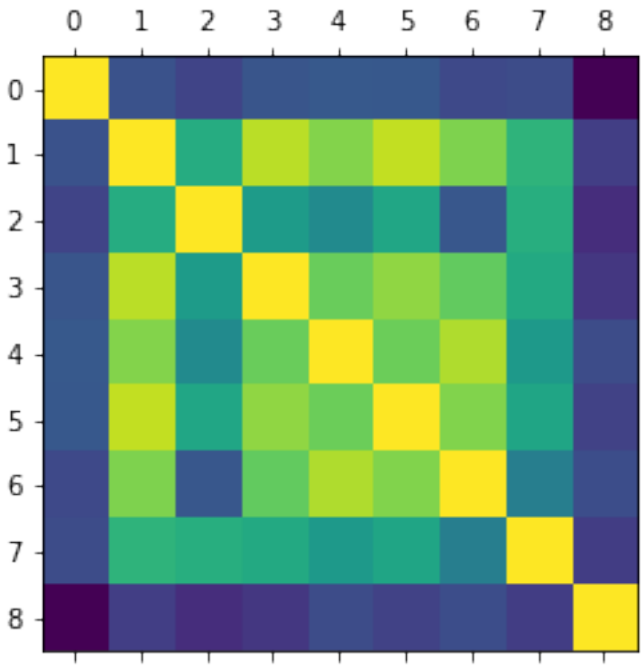
```
Out[62]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000023592743470>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592D35358>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592704A90>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x0000023592783E10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592DC40F0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592FBB9B0>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x0000023592F31F28>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592F23518>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x0000023592F23550>]],
               dtype=object)
```



```
In [71]: age = df_mansize['Age']
         height = df_mansize['Height (cm)']
         weight = df_mansize['Weight (kg)']
         femur = df_mansize['Femur Length (cm)']
         feetsize = df_mansize['Feet Size (cm)']
         armspan = df_mansize['Arm span (cm)']
         handlenght = df_mansize['Hand length (cm)']
         cranialvolume = df_mansize['Cranial volume (cm3)']
         penissize = df_mansize['Penis size (cm)']

         corr_matrix_mansize = df_mansize.corr()
         print(corr_matrix_mansize)
         plt.matshow(corr_matrix_mansize)
```

```
                          Age  Height (cm)  Weight (kg)  Femur Length (cm)  \
Age                  1.000000     0.198026     0.146802           0.212554
Height (cm)          0.198026     1.000000     0.591516           0.890573
Weight (kg)          0.146802     0.591516     1.000000           0.517094
Femur Length (cm)    0.212554     0.890573     0.517094           1.000000
Feet Size (cm)       0.226708     0.802437     0.439485           0.754205
Arm span (cm)        0.221791     0.903203     0.560522           0.823258
Hand length (cm)     0.166387     0.791568     0.218642           0.742029
Cranial volume (cm3) 0.178993     0.624609     0.599918           0.580032
Penis size (cm)     -0.071679     0.127375     0.068403           0.100553

                     Feet Size (cm)  Arm span (cm)  Hand length (cm)  \
Age                        0.226708       0.221791          0.166387
Height (cm)                0.802437       0.903203          0.791568
Weight (kg)                0.439485       0.560522          0.218642
Femur Length (cm)          0.754205       0.823258          0.742029
Feet Size (cm)             1.000000       0.758345          0.871076
Arm span (cm)              0.758345       1.000000          0.795127
Hand length (cm)           0.871076       0.795127          1.000000
Cranial volume (cm3)       0.504662       0.559920          0.386198
Penis size (cm)            0.176398       0.140155          0.182780

                     Cranial volume (cm3)  Penis size (cm)
Age                              0.178993        -0.071679
Height (cm)                      0.624609         0.127375
Weight (kg)                      0.599918         0.068403
Femur Length (cm)                0.580032         0.100553
Feet Size (cm)                   0.504662         0.176398
Arm span (cm)                    0.559920         0.140155
Hand length (cm)                 0.386198         0.182780
Cranial volume (cm3)             1.000000         0.124220
Penis size (cm)                  0.124220         1.000000
```

```
Out[71]: <matplotlib.image.AxesImage at 0x235927fa7b8>
```




```
In [68]: print (is_correlated(age,weight))
         #Ajouter les autres corrélation
```

```
0.147719890694635
```

```
In [72]: print(confidence_interval(age,weight))
         #test avec les autres
```

```
(-0.007120452423739459, 0.2956424155552849)
```

3 Test d'indépendance et variables catégorielles

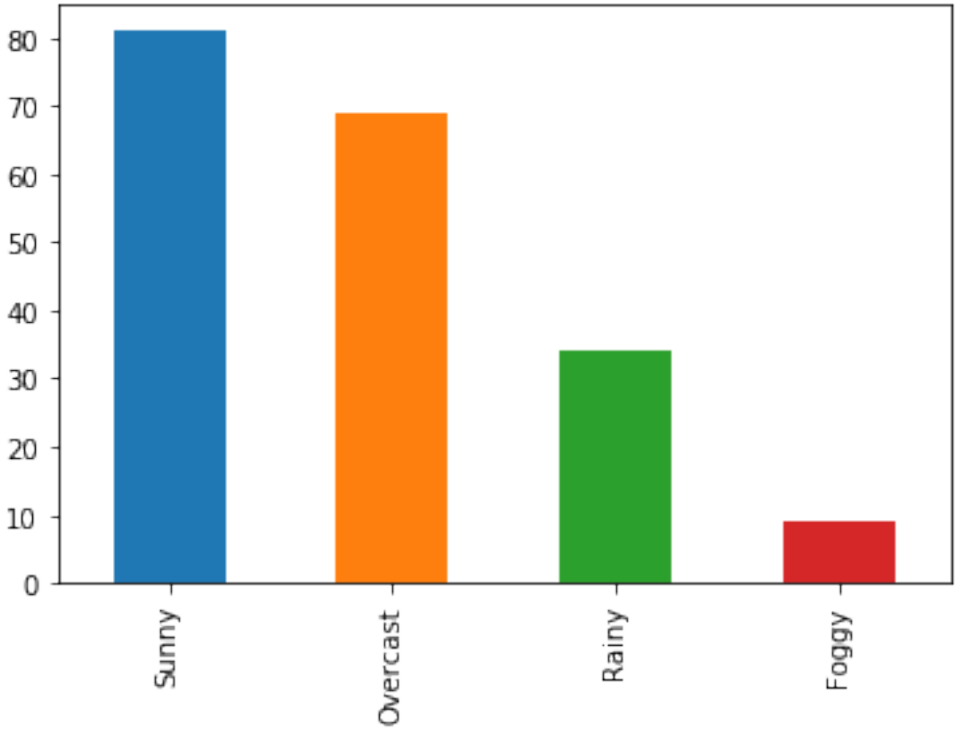
```
In [96]: df_weather = pd.read_csv("weather.csv",sep=";")
```

```
In [97]: df_weather.head()
```

```
Out[97]:
           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
```

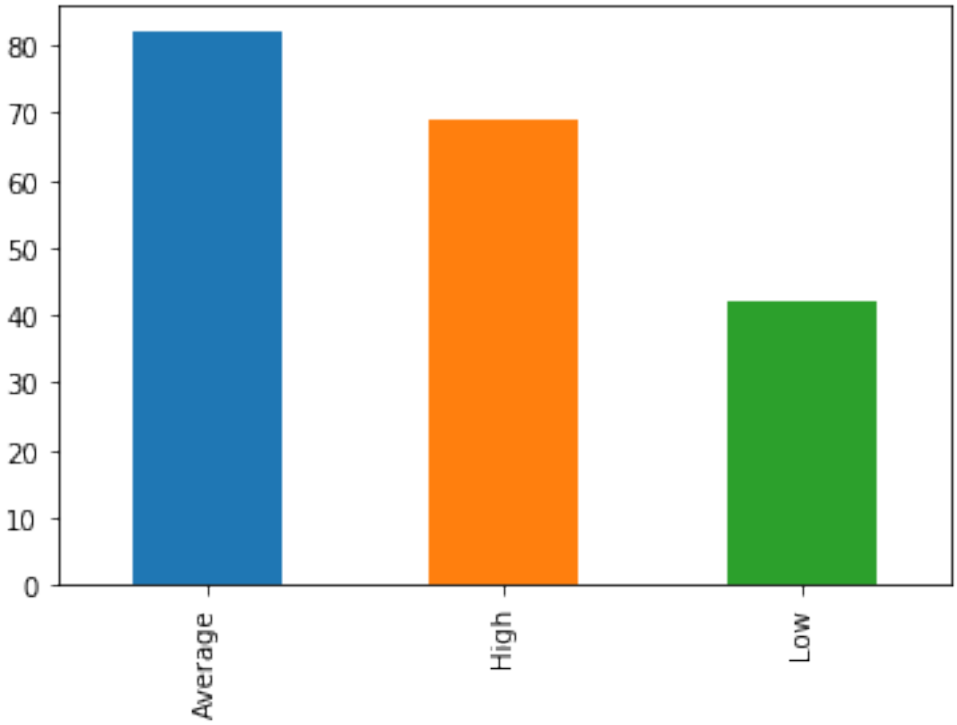
In [110]:
```
#City pas affichable car 1 valeur pour chaque ville (réflechir à un autre affichage)
outlooknbval = df_weather['Outlook'].value_counts()
humiditynbval = df_weather['Humidity'].value_counts()
temperaturenbval = df_weather['Temperature'].value_counts()


outlooknbval.plot(kind='bar')
```

Out[110]: <matplotlib.axes._subplots.AxesSubplot at 0x2359bf69668>



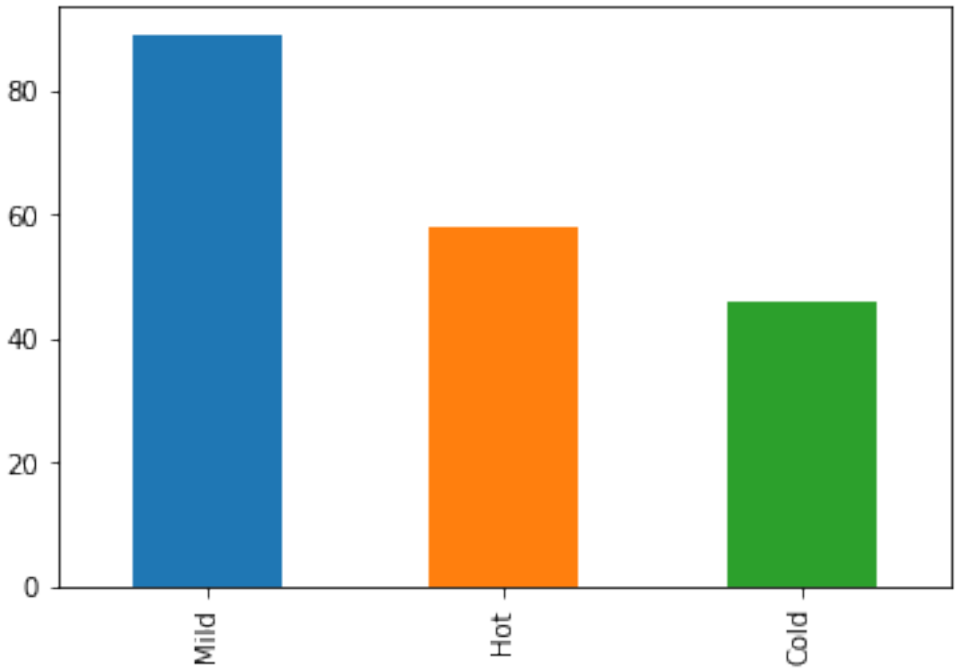
In [109]:
```
humiditynbval.plot(kind='bar')
```

Out[109]: <matplotlib.axes._subplots.AxesSubplot at 0x2359bfec550>




```
In [111]: temperaturenbval.plot(kind='bar')
```

```
Out[111]: <matplotlib.axes._subplots.AxesSubplot at 0x2359be1d390>
```



```
In [149]: crosstab = pd.crosstab(df_weather['Outlook'],df_weather['Temperature'])
          print(crosstab.shape)
```

```
(4, 3)
```

```
In [151]: def degre_de_liberte(df):
              return((df.shape[0]-1)*(df.shape[1]-1))
          print(degre_de_liberte(crosstab))
```

```
6
```

```
In [152]: from scipy.stats import chisquare
          chisquare(crosstab)
```

```
Out[152]: Power_divergenceResult(statistic=array([15.04347826, 26.55172414, 34.37078652]), pvalu
```