Module 3 : Data Exploration

Reference Book

- Reference the "Data Exploration" chapter in Book Introduction to Data Mining to understand some of the concepts introduced in this tutorial notebook.
- Notebook Down

Data Exploration

- Data exploration refers to the preliminary investigation of data in order to better understand its specific characteristics.
- Two motivations :
    1. To help users select the appropriate preprocessing and data analysis technique used.
    2. To make use of humans' abilities to recognize patterns in the data.

3.1 Summary Statistics

Introduction

Summary statistics are quantities, such as the mean and standard deviation, that capture various characteristics of a potentially large set of values with a single number or a small set of numbers. In this tutorial, we will use the Iris sample data, which contains information on 150 Iris flowers, 50 each from one of
three Iris species: Setosa, Versicolour, and Virginica. Each flower is characterized by five attributes:

- sepal length in centimeters
- sepal width in centimeters
- petal length in centimeters
- petal width in centimeters
- class(Setosa,Versicolour,Virginica)

Target

- Load a CSV data file into a Pandas DataFrame object;
- Compute various summary statistics from the DataFrame.

(1) Download CSV File

UCI Machine Learning Repository : Iris Dataset

```
import pandas as pd

data = pd.read_csv("http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data",header=None)
data.columns = ['sepal length','sepal width','petal length','petal width','class']

data
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width	class
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

```
data.head() # display first five row
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width	class
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

(2) Calculation

For each quantitative attribute, calculate its average, standard deviation, minimum, and maximum values.

Functions

- mean(): get the average
- std() : get the standard deviation
- min() : get the minmum
- max() : get the maxmum

```
from pandas.api.types import is_numeric_dtype
for col in data.columns:
    if is_numeric_dtype(data[col]):
        print('%s:' % (col))
        print('\t Mean = %.2f' % data[col].mean())
        print('\t Standard deviation = %.2f' % data[col].std())
        print('\t Minimum = %.2f' % data[col].min())
        print('\t Maximum = %.2f' % data[col].max())

```

```
sepal length:
     Mean = 5.84
     Standard deviation = 0.83
     Minimum = 4.30
     Maximum = 7.90
sepal width:
     Mean = 3.05
     Standard deviation = 0.43
     Minimum = 2.00
     Maximum = 4.40
```

```
petal length:
     Mean = 3.76
     Standard deviation = 1.76
     Minimum = 1.00
     Maximum = 6.90
petal width:
     Mean = 1.20
     Standard deviation = 0.76
     Minimum = 0.10
     Maximum = 2.50
```

(3) Count the Frequency

For the qualitative attribute (calss), count the frequency for each of its distinct values.

```
data['class'].value_counts()
```

```
Iris-versicolor    50
Iris-setosa        50
Iris-virginica     50
Name: class, dtype: int64
```

(4) describe() function

- It is also possible to display the summary for all the attributes simultaneously in a table using the describe() function.
- If an attribute is quantitative, it will display its mean, standard deviation and various quantiles (including minimum, median, and maximum) values.
- If an attribute is qualitative, it will display its number of unique values and the top (most frequent) values.

```
data.describe(include='all')
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width	class
count	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	Iris-versicolor
freq	NaN	NaN	NaN	NaN	50
mean	5.843333	3.054000	3.758667	1.198667	NaN
std	0.828066	0.433594	1.764420	0.763161	NaN
min	4.300000	2.000000	1.000000	0.100000	NaN
25%	5.100000	2.800000	1.600000	0.300000	NaN
50%	5.800000	3.000000	4.350000	1.300000	NaN
75%	6.400000	3.300000	5.100000	1.800000	NaN
max	7.900000	4.400000	6.900000	2.500000	NaN

(5) Multivariate Statistics

- Compute the covariance and correlation between pairs of attributes.

Function

- cov() : calculate the covariance
- corr(): calculate the correlation

```
print("Covariance:")
data.cov()
```

```
Covariance:
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width
sepal length	0.685694	-0.039268	1.273682	0.516904
sepal width	-0.039268	0.188004	-0.321713	-0.117981
petal length	1.273682	-0.321713	3.113179	1.296387
petal width	0.516904	-0.117981	1.296387	0.582414

```
print("Correlation:")
data.corr()
```

```
Correlation:
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	sepal length	sepal width	petal length	petal width
sepal length	1.000000	-0.109369	0.871754	0.817954
sepal width	-0.109369	1.000000	-0.420516	-0.356544
petal length	0.871754	-0.420516	1.000000	0.962757
petal width	0.817954	-0.356544	0.962757	1.000000

3.2 Data Visualization

- Data visualization is the display of information in a graphic or tabular format.
- Target : the characteristics of the data and the relationships among data items or attributes can be analyzed or reported.

(1) Histogram

Function : hist()

- Create the histogram for target attribute;
- Count the frequency for each bin.

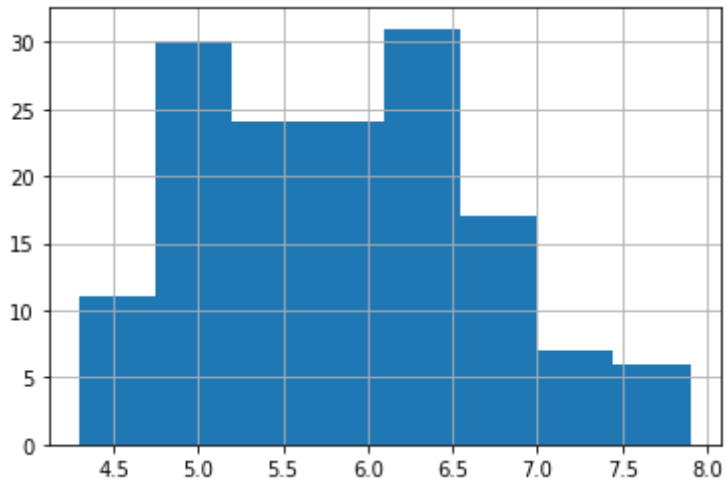
parameter :

- bin : the number of separate bins

```
# First, we will display the histogram for the sepal length attribute
# by discretizing it into 8 separate bins and counting the frequency
# for each bin.

%matplotlib inline
data['sepal length'].hist(bins = 8)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a586045548>
```



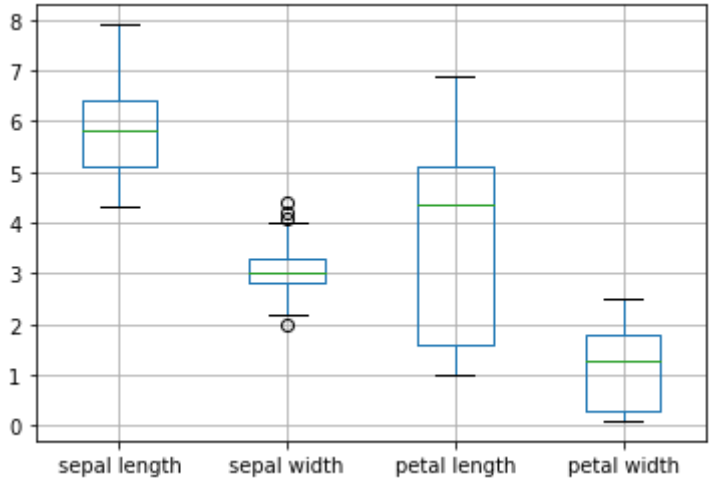
（2）Box plot

Function : boxplot()

- Show the distribution of values for each attribute.

```
data.boxplot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a58862cf08>
```



(3) Scatter Plot

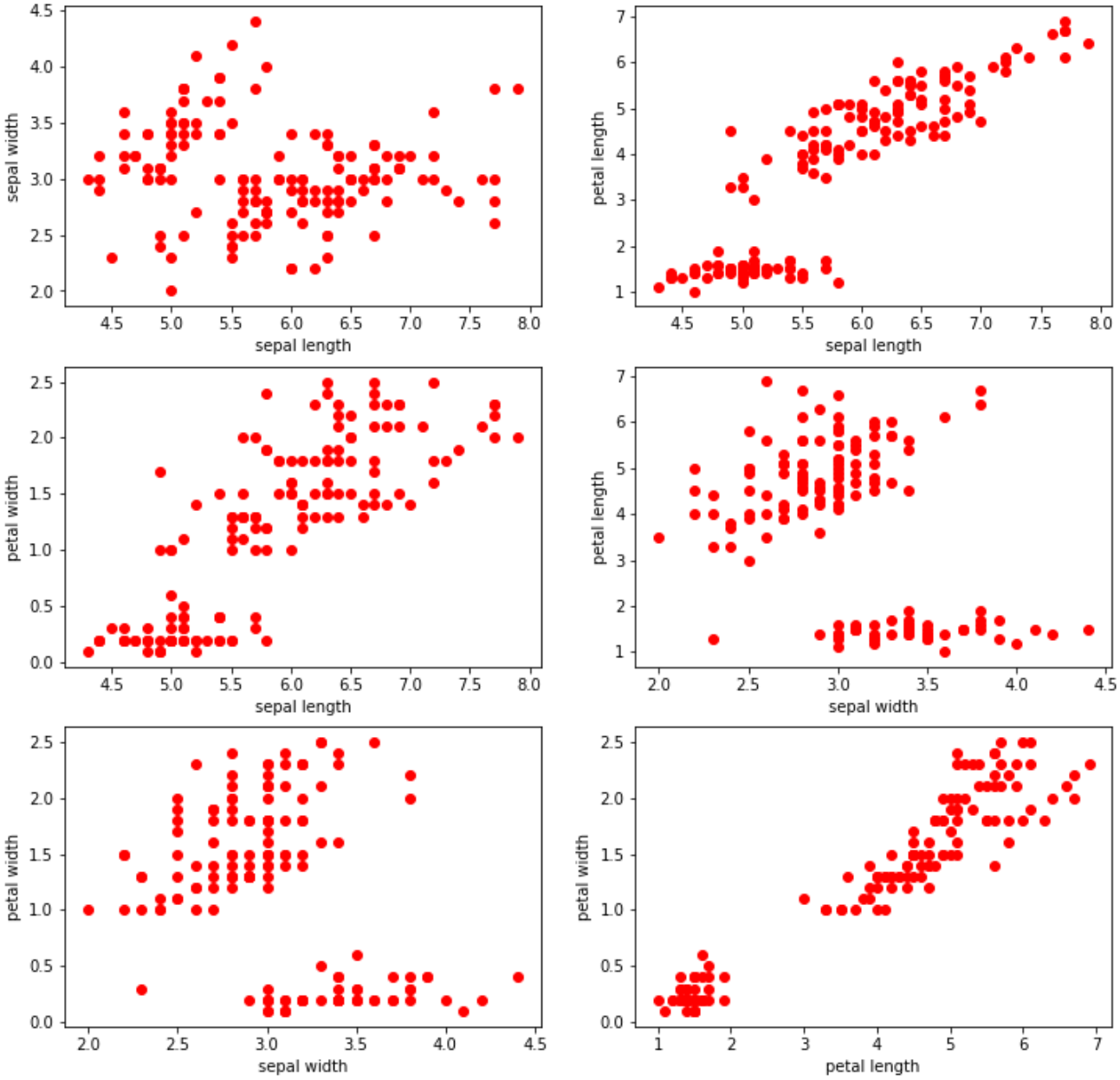
For each pair of attributes, we can use a scatter plot to visualize their joint distribution.

Function

- subplots()

```
import matplotlib.pyplot as plt

fig,axes = plt.subplots(3,2,figsize=(12,12))
index = 0
for i in range(3):
    for j in range(i+1,4):
        ax1 = int(index/2)
        ax2 = index % 2
        axes[ax1][ax2].scatter(data[data.columns[i]], data[data.columns[j]], color='red')
        axes[ax1][ax2].set_xlabel(data.columns[i])
        axes[ax1][ax2].set_ylabel(data.columns[j])
        index = index + 1
```



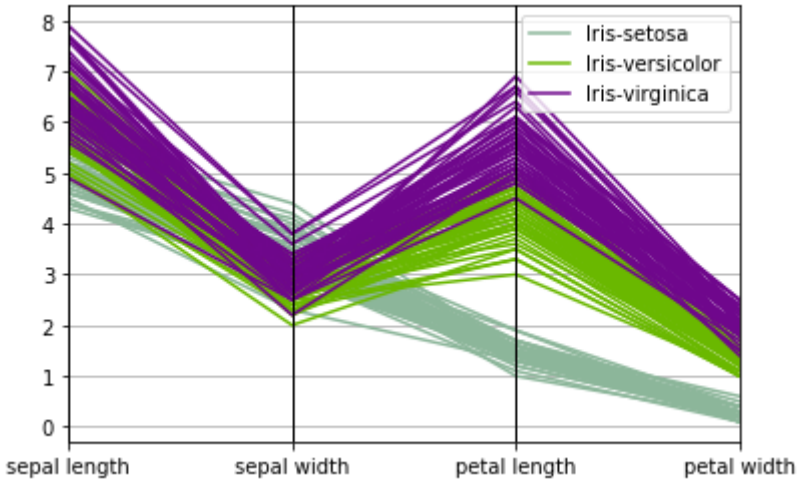
(4) Paralle Corrdinates

- Parallel coordinates can be used to display all the data points simultaneously.
- Parallel coordinates have one coordinate axis for each attribute, but the different axes are parallel to one other instead of perpendicular, as is traditional.
- Furthermore, an object is represented as a line instead of as a point. In the example below, the distribution of values for each class can be identified in a separate color.

```
from pandas.plotting import parallel_coordinates
%matplotlib inline

parallel_coordinates(data,'class')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a588ac5208>
```


Iris-setosa
Iris-versicolor
Iris-virginica
8
7
6
5
4
3
2
1
0
sepal length
sepal width
petal length
petal width