LA03_Ex3_DataUnderstanding

April 26, 2018

1 Hochschule Bonn-Rhein-Sieg

2 Learning and Adaptivity, SS18

3 Assignment 03 (24-April-2018)

3.1 Sathiya Ramesh, Pradheep Krishna Muthukrishnan Padmanabhan, Naresh Kumar Gurulingan

4 Data Understanding

Iris dataset

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        from __future__ import print_function
        url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
        names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
        dataset = pd.read_csv(url, names=names)
```

4.1 Task 1: Summary of the Dataset

- Dimensions of the dataset.
- Peek at the data itself.
- Statistical summary of all attributes.
- Breakdown of the data by the class variable.

```
In [2]: print('{} is the dimensions of the dataset'.format(dataset.shape))
```

```
(150, 5) is the dimensions of the dataset
```

```
In [3]: print('Peek at the dataset:')
        dataset.head()
```

```
Peek at the dataset:
```

Out[3]:
```
   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
```

In [4]:
```
print('Statistical summary of all attributes in the dataset:')
dataset.describe()
```

```
Statistical summary of all attributes in the dataset:
```

Out[4]:
```
       sepal-length  sepal-width  petal-length  petal-width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
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
```

In [5]:
```
for key in dataset.groupby('class').groups.keys():
    print(dataset[dataset['class']==key].head())
```

```
   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
    sepal-length  sepal-width  petal-length  petal-width            class
50           7.0          3.2           4.7          1.4  Iris-versicolor
51           6.4          3.2           4.5          1.5  Iris-versicolor
52           6.9          3.1           4.9          1.5  Iris-versicolor
53           5.5          2.3           4.0          1.3  Iris-versicolor
54           6.5          2.8           4.6          1.5  Iris-versicolor
     sepal-length  sepal-width  petal-length  petal-width           class
100           6.3          3.3           6.0          2.5  Iris-virginica
101           5.8          2.7           5.1          1.9  Iris-virginica
102           7.1          3.0           5.9          2.1  Iris-virginica
103           6.3          2.9           5.6          1.8  Iris-virginica
104           6.5          3.0           5.8          2.2  Iris-virginica
```

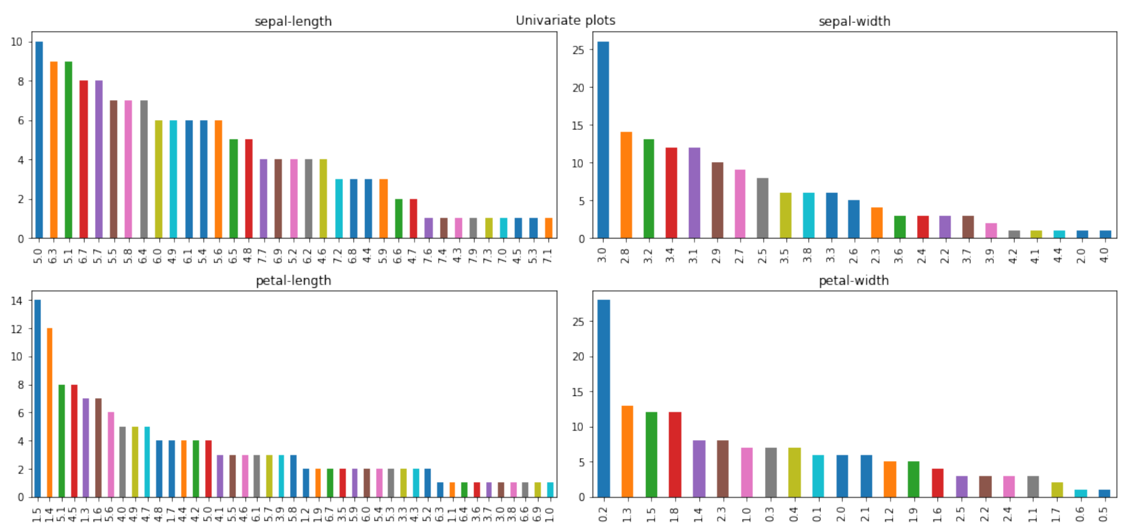
4.2 Task 2: Data Visualization

- Univariate plots, visualisation of each individual feature for better understand.

- Multivariate plots, visualisation relationships between attributes.

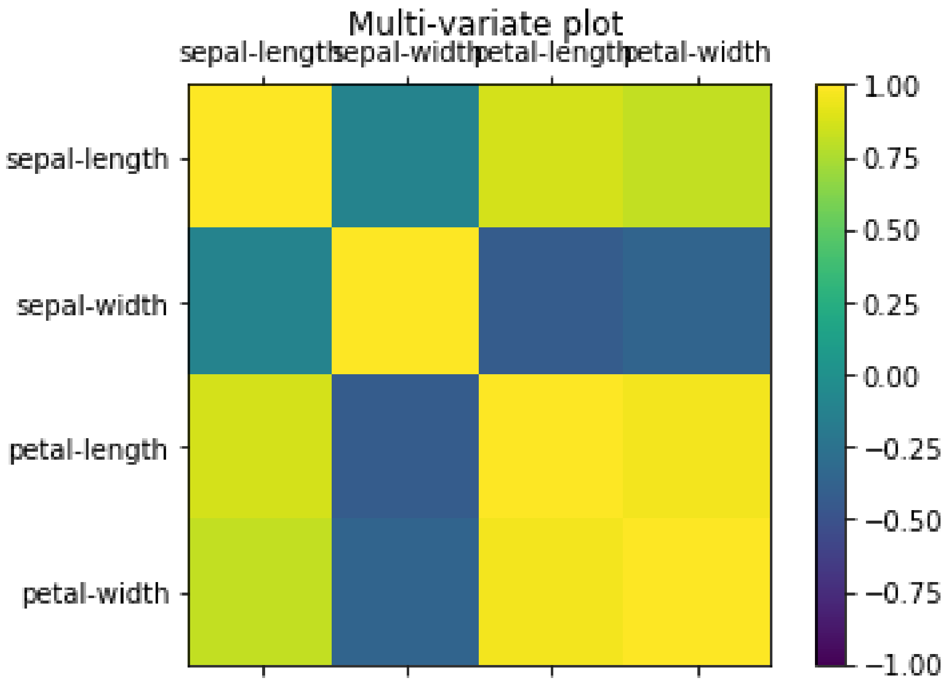
Reference: Visualize Machine Learning Data in Python With Pandas

In [6]:
```
fig = plt.figure()
fig.set_figheight(7)
fig.set_figwidth(15)
plt.tight_layout()
fig.add_subplot(221)
dataset['sepal-length'].value_counts().plot.bar()
plt.title('sepal-length')
fig.add_subplot(222)
dataset['sepal-width'].value_counts().plot.bar()
plt.title('sepal-width')
fig.add_subplot(223)
dataset['petal-length'].value_counts().plot.bar()
plt.title('petal-length')
fig.add_subplot(224)
dataset['petal-width'].value_counts().plot.bar()
plt.title('petal-width')
plt.tight_layout()
fig.suptitle('Univariate plots')
plt.show()
```



In [7]:
```
correlations = dataset.corr()
# plot correlation matrix
fig = plt.figure()
fig.suptitle('Multi-variate plot')
ax = fig.add_subplot(111)
cax = ax.matshow(correlations, vmin=-1, vmax=1)
```

```
fig.colorbar(cax)
ticks = np.arange(0,4,1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(names)
ax.set_yticklabels(names)
plt.show()
```



4.3 Task 3: Validation set

We will split the loaded dataset into two, 80% of which we will use to train our models and 20% that we will hold back as a validation dataset.

```
In [8]: from sklearn.model_selection import train_test_split

        print (np.shape(dataset))
        print ("So 80% of train dataset means we need=", dataset.shape[0]*80/100)
        print ("So 20% of train dataset means we need=", dataset.shape[0]*20/100)
        train, test = train_test_split(dataset, test_size=0.2)
        #Now the dataset is split, the reading for the same is given here
        print("Train dataset after splitting",train.shape[0])
        print("Test dataset after splitting",test.shape[0])
```

```
(150, 5)
So 80% of train dataset means we need= 120.0
```

```
So 20% of train dataset means we need= 30.0
Train dataset after splitting 120
Test dataset after splitting 30
```