In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
from sklearn.datasets import load_iris
```

In [3]:

```
dataset = load_iris()
```

In [4]:

```
print(dataset.DESCR)
```

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================

    :Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is ta
ken
from Fisher's paper. Note that it's the same as in R, but not as in the UC
I
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature.  Fisher's paper is a classic in the field
and
is referenced frequently to this day.  (See Duda & Hart, for example.)  Th
e
data set contains 3 classes of 50 instances each, where each class refers
to a
type of iris plant.  One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

.. topic:: References

   - Fisher, R.A. "The use of multiple measurements in taxonomic problems"
     Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions t
o
     Mathematical Statistics" (John Wiley, NY, 1950).
   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Anal
ysis.
     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.
```

```
    - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System
      Structure and Classification Rule for Recognition in Partially Expose
d
      Environments".  IEEE Transactions on Pattern Analysis and Machine
      Intelligence, Vol. PAMI-2, No. 1, 67-71.
    - Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule".  IEEE Transac
tions
      on Information Theory, May 1972, 431-433.
    - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLASS I
I
      conceptual clustering system finds 3 classes in the data.
    - Many, many more ...
```

In [5]:

```
X = dataset.data
```

In [6]:

```
Y = dataset.target
```

In [7]:

```
Y
```

Out[7]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [8]:

```
X
```

Out[8]:

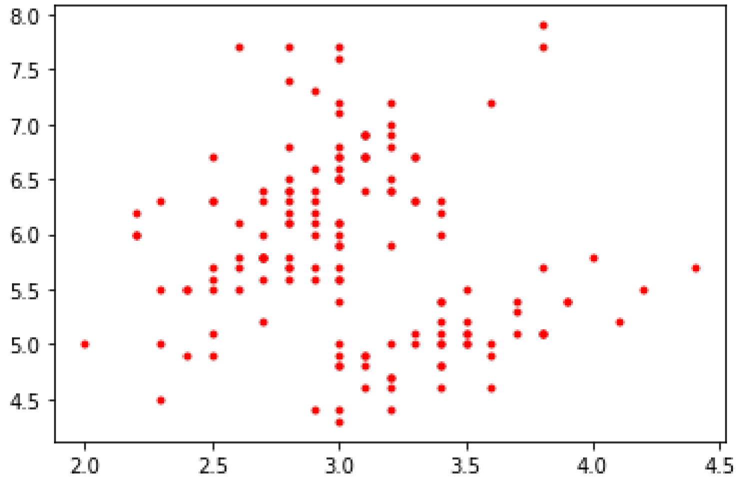
```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
```

In [10]:

```
plt.plot(X[:, 1], X[:,0],'r.')
```

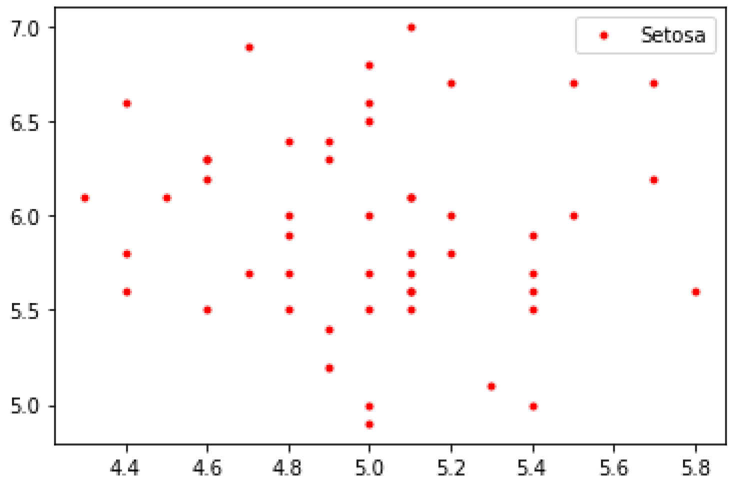
Out[10]:

```
[<matplotlib.lines.Line2D at 0x25d6e1421c0>]
```



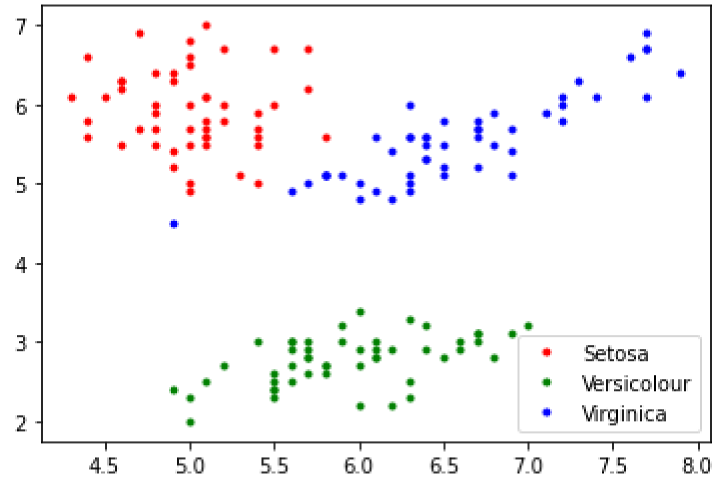
In [12]:

```
plt.plot(X[:, 0][Y ==0], X[:,0][Y ==1],'r.', label='Setosa')
plt.legend()
plt.show()
```



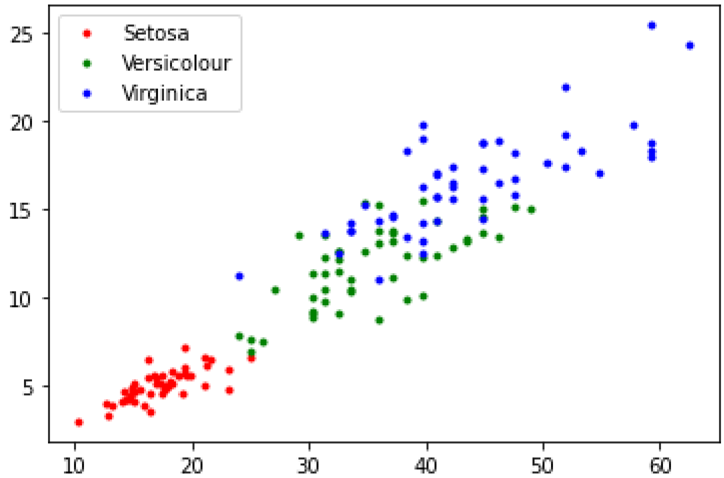
In [14]:

```
plt.plot(X[:, 0][Y ==0], X[:,0][Y ==1],'r.', label='Setosa')
plt.plot(X[:, 0][Y ==1], X[:,1][Y ==1],'g.', label='Versicolour')
plt.plot(X[:, 0][Y ==2], X[:,2][Y ==2],'b.', label='Virginica')
plt.legend()
plt.show()
```



In [15]:

```
plt.plot(X[:, 0][Y == 0]* X[:, 1][Y == 0], X[:, 1][Y == 0]* X[:, 2][Y == 0], 'r.', label
plt.plot(X[:, 0][Y == 1]* X[:, 0][Y == 1], X[:, 1][Y == 1]* X[:, 2][Y == 1], 'g.', label
plt.plot(X[:, 0][Y == 2]* X[:, 0][Y == 2], X[:, 1][Y == 2]* X[:, 2][Y == 2], 'b.', label
plt.legend()
plt.show()
```



In [16]:

```
from sklearn.preprocessing import StandardScaler
X = StandardScaler().fit_transform(X)
```

In [17]:

```
from sklearn.model_selection import train_test_split
```

In [18]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y)
```

In [19]:

```
from sklearn.linear_model import LogisticRegression
```

In [20]:

```
log_reg = LogisticRegression()
```

In [21]:

```
log_reg.fit(X_train, Y_train)
```

Out[21]:

```
LogisticRegression()
```

In [22]:

```
log_reg.score(X_test, Y_test)
```

Out[22]:

```
0.9736842105263158
```

In [ ]:

```

```