In [1]:

```
import numpy as np
```

In [2]:

```
import matplotlib.pyplot as plt
```

In [6]:

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

In [7]:

```
iris=load_iris()
```

In [8]:

```
from sklearn.cluster import KMeans
```

In [11]:

```
iris.DESCR
```

Out[11]:

```
'.. _iris_dataset:\n\nIris plants dataset\n--------------------\n\n**Data Se
t Characteristics:**\n\n    :Number of Instances: 150 (50 in each of three c
lasses)\n    :Number of Attributes: 4 numeric, predictive attributes and the
class\n    :Attribute Information:\n        - sepal length in cm\n        -
sepal width in cm\n        - petal length in cm\n        - petal width in cm
\n        - class:\n                - Iris-Setosa\n                - Iris-Ve
rsicolour\n                - Iris-Virginica\n                \n    :Summary
Statistics:\n\n    ============== ==== ==== ======= ===== ==================
==\n                    Min  Max   Mean    SD   Class Correlation\n    =====
========= ==== ==== ======= ===== ====================\n    sepal length:
4.3  7.9   5.84   0.83    0.7826\n    sepal width:    2.0  4.4   3.05   0.43
-0.4194\n    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)\n
petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)\n    ============
= ==== ==== ======= ===== ====================\n\n    :Missing Attribute Val
ues: None\n    :Class Distribution: 33.3% for each of 3 classes.\n    :Creat
or: R.A. Fisher\n    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
\n    :Date: July, 1988\n\nThe famous Iris database, first used by Sir R.A.
Fisher. The dataset is taken\nfrom Fisher\'s paper. Note that it\'s the same
as in R, but not as in the UCI\nMachine Learning Repository, which has two w
rong data points.\n\nThis is perhaps the best known database to be found in
the\npattern recognition literature.  Fisher\'s paper is a classic in the fi
eld and\nis referenced frequently to this day.  (See Duda & Hart, for exampl
e.)  The\ndata set contains 3 classes of 50 instances each, where each class
refers to a\ntype of iris plant.  One class is linearly separable from the o
ther 2; the\nlatter are NOT linearly separable from each other.\n\n.. topi
c:: References\n\n   - Fisher, R.A. "The use of multiple measurements in tax
onomic problems"\n     Annual Eugenics, 7, Part II, 179-188 (1936); also in
"Contributions to\n     Mathematical Statistics" (John Wiley, NY, 1950).\n
- Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysi
s.\n     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.\n
- Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System\n
Structure and Classification Rule for Recognition in Partially Exposed\n
Environments".  IEEE Transactions on Pattern Analysis and Machine\n     Inte
lligence, Vol. PAMI-2, No. 1, 67-71.\n   - Gates, G.W. (1972) "The Reduced N
earest Neighbor Rule".  IEEE Transactions\n     on Information Theory, May 1
972, 431-433.\n   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et a
l"s AUTOCLASS II\n     conceptual clustering system finds 3 classes in the d
ata.\n   - Many, many more ...'
```

In [12]:

```
iris.data
```

Out[12]:

```
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 [13]:

```
x=iris.data[:, :2]
```

In [14]:

```
x
```

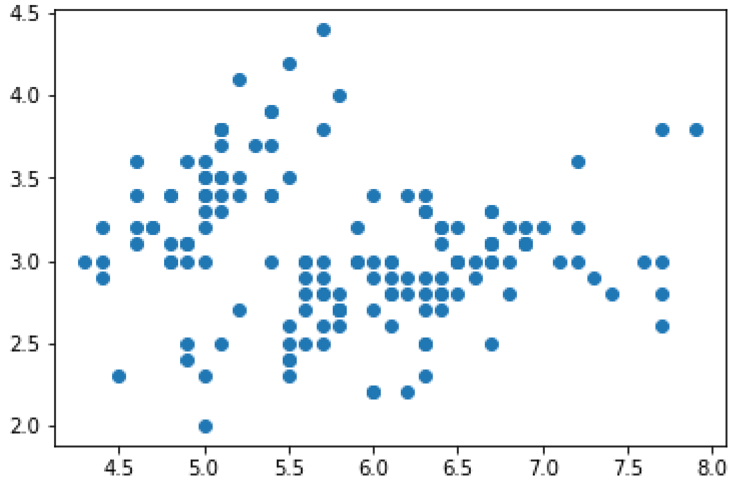
```
       [7.9, 3.8],
       [6.4, 2.8],
       [6.3, 2.8],
       [6.1, 2.6],
       [7.7, 3. ],
       [6.3, 3.4],
       [6.4, 3.1],
       [6. , 3. ],
       [6.9, 3.1],
       [6.7, 3.1],
       [6.9, 3.1],
       [5.8, 2.7],
       [6.8, 3.2],
       [6.7, 3.3],
       [6.7, 3. ],
       [6.3, 2.5],
       [6.5, 3. ],

       [6.2, 3.4],
       [5.9, 3. ]])
```

In [15]:

```
plt.scatter(x[:,0],x[:,1])
```

Out[15]:

```
<matplotlib.collections.PathCollection at 0x15c045b3fa0>
```



In [23]:

```
kmeans=KMeans(n_clusters=3)
```

In [24]:

```
y_pred=kmeans.fit_predict(x)
print(y_pred)
```

```
[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 0 0 0 1 0 1 0 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
 0 0 0 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0
 0 0 1 1 0 0 0 0 1 0 1 0 1 0 0 1 1 0 0 0 0 0 1 1 0 0 0 1 0 0 0 1 0 0 0 1 0
 0 1]
```

In [25]:

```
clusters = kmeans.cluster_centers_
```

In [26]:

```
clusters
```

Out[26]:

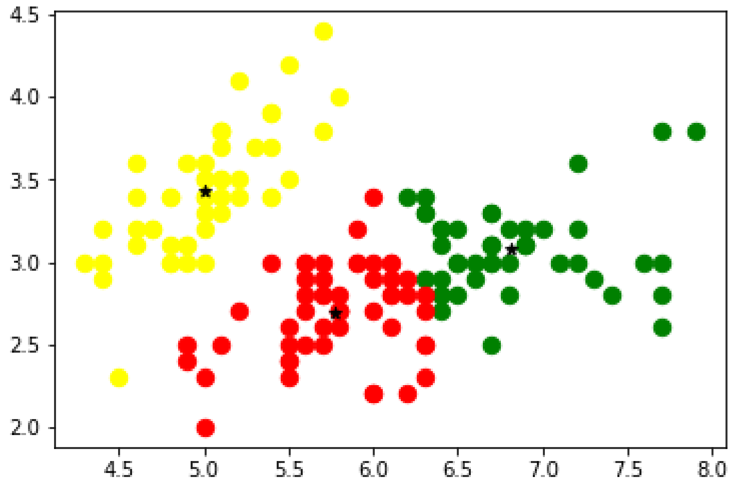
```
array([[6.81276596, 3.07446809],
       [5.77358491, 2.69245283],
       [5.006     , 3.428     ]])
```

In [27]:

```
plt.scatter(x[y_pred==0,0], x[y_pred==0,1], s=70, color='green')
plt.scatter(x[y_pred==1,0], x[y_pred==1,1], s=70, color='red')
plt.scatter(x[y_pred==2,0], x[y_pred==2,1], s=70, color='yellow')

plt.scatter(clusters[0][0],clusters[0][1], marker='*', color='black')
plt.scatter(clusters[1][0],clusters[1][1], marker='*', color='black')
plt.scatter(clusters[2][0],clusters[2][1], marker='*', color='black')
```

Out[27]:

```
<matplotlib.collections.PathCollection at 0x15c052a9970>
```

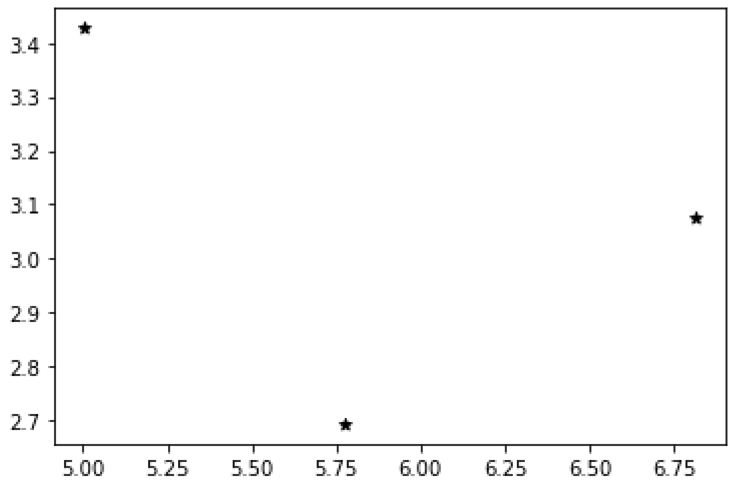


In [28]:

```
plt.scatter(clusters[0][0],clusters[0][1], marker='*', color='black')
plt.scatter(clusters[1][0],clusters[1][1], marker='*', color='black')
plt.scatter(clusters[2][0],clusters[2][1], marker='*', color='black')
```

Out[28]:

```
<matplotlib.collections.PathCollection at 0x15c0531f250>
```



In [ ]: