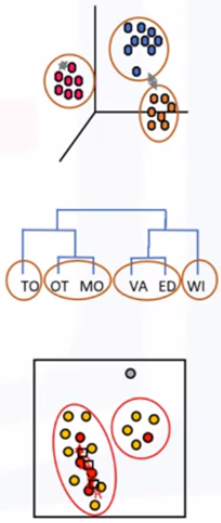
Clustering

A cluster is a group of data points or objects in a dataset that are similar to other objects in the group, and dissimilar to datapoints in other clusters

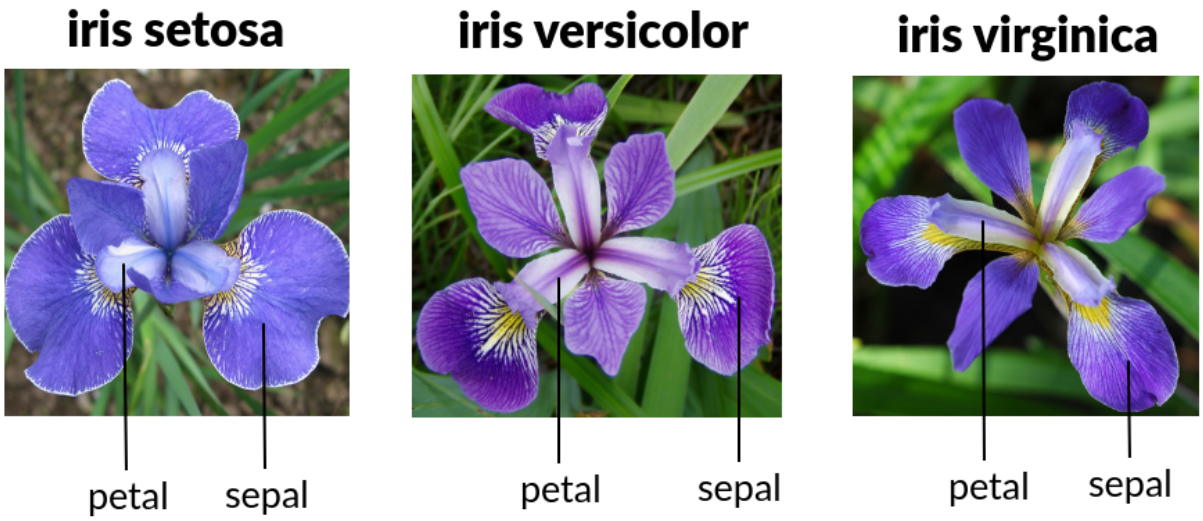
Clustering algorithms

- Partitioned-based Clustering
  - Relatively efficient
  - E.g. k-Means, k-Median, Fuzzy c-Means
- Hierarchical Clustering
  - Produces trees of clusters
  - E.g. Agglomerative, Divisive
- Density-based Clustering
  - Produces arbitrary shaped clusters
  - E.g. DBSCAN



1. Partitioned-based Clustering

KMeans



In [1]:

```
from sklearn.datasets import load_iris
iris = load_iris()
iris
```

Out[1]:

```
{'data': 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],
```

In [2]:

```
data = iris['data']
```

In [3]:

```
data = iris.data
```

In [4]:

```
data
```

Out[4]:

```
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],
```

In [5]:

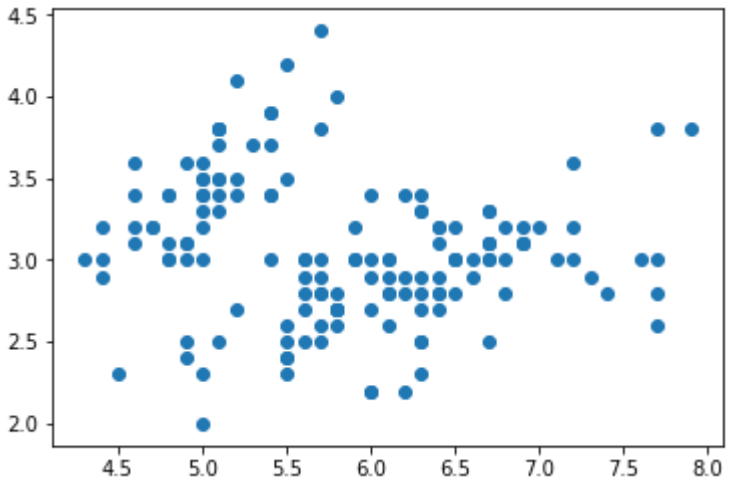
```
1 import matplotlib.pyplot as plt
```

In [6]:

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

Out[6]:

```
<matplotlib.collections.PathCollection at 0x25f3d02ada0>
```



In [7]:

```
1 req = data[:,0:2]
2 req
```

Out[7]:

```
array([[5.1, 3.5],
       [4.9, 3. ],
       [4.7, 3.2],
       [4.6, 3.1],
       [5. , 3.6],
       [5.4, 3.9],
       [4.6, 3.4],
       [5. , 3.4],
       [4.4, 2.9],
       [4.9, 3.1],
       [5.4, 3.7],
       [4.8, 3.4],
       [4.8, 3. ],
       [4.3, 3. ],
       [5.8, 4. ],
       [5.7, 4.4],
       [5.4, 3.9],
```

In [8]:

```
from sklearn.model_selection import train_test_split
cl_tr,cl_ts = train_test_split(req,test_size = 0.3,random_state = 7)

from sklearn.cluster import KMeans

model = KMeans(n_clusters = 3)

model.fit(cl_tr)
```

Out[8]:

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

In [9]:

```
y_pred = model.predict(cl_ts)
y_pred
```

Out[9]:

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

In [10]:

```
# Import pyplot
import matplotlib.pyplot as plt

# Assign the columns of new_points: xs and ys
xs = cl_ts[:,0]
ys = cl_ts[:,1]

# Make a scatter plot of xs and ys, using labels to define the colors
plt.scatter(xs,ys,c = y_pred,alpha=0.5)

# Assign the cluster centers: centroids
centroids = model.cluster_centers_

# Assign the columns of centroids: centroids_x, centroids_y
centroids_x = centroids[:,0]
centroids_y = centroids[:,1]

# Make a scatter plot of centroids_x and centroids_y
plt.scatter(centroids_x,centroids_y,marker = 'D',s = 50)
plt.show()
```