http://scikit-learn.org/stable/modules/clustering.html

ESEMPIO K-MEANS

```
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> X = iris.data
>>> from sklearn.cluster import KMeans
>>> km = KMeans(n_clusters = 3,init='random')
>>> km.fit(X)
KMeans(copy_x=True, init='k-means++', k=None,
max_iter=300, n_clusters=3,
n_init=10, n_jobs=1, precompute_distances=True,
random_state=None, tol=0.0001, verbose=0)
>>> km.cluster_centers_
array([[ 5.006 , 3.418 , 1.464 , 0.244 ],
[ 6.85 , 3.07368421, 5.74210526, 2.07105263],
[ 5.9016129 , 2.7483871 , 4.39354839, 1.43387097]])
>>> km.labels_
…
>>> km.inertia_
78.940841426146022
```

SCATTERING

```
>>> import pylab as pl
>>> pl.scatter(X[:, 0], X[:, 1], c=iris.target)
<matplotlib.collections.PathCollection object at
0x0000000010C0B0F0>
>>> pl.scatter(km.cluster_centers_[:, 0],
km.cluster_centers_[:, 1], marker='o', s=100)
<matplotlib.collections.PathCollection object at
0x0000000010C0B898>
>>> pl.show()
```

ESEMPIO HIERACHICAL CLUSTERING

```
>>> import scipy.cluster.hierarchy as h
>>> import pylab as pl
>>> pl.plot()
>>> H = h.single(X)
>>> h.dendrogram(H)
>>> pl.show()
```

Provare con altri tipi di linkage.
complete(), average(), centroid()

FLAT...

```
>>> C = h.fcluster(H,0.6,criterion='distance')
>>> C # per verificare l assegnamento
>>> len(np.unique(C)) # per verificare il numero di
cluster
EVAL..
>>> from sklearn import metrics
>>> Y = iris.target
>>> metrics.adjusted_rand_score(C,-Y)
>>> metrics.adjusted_mutual_info_score(C,Y)
```