Politecnico di Torino

Data Science Lab

Scikit-learn

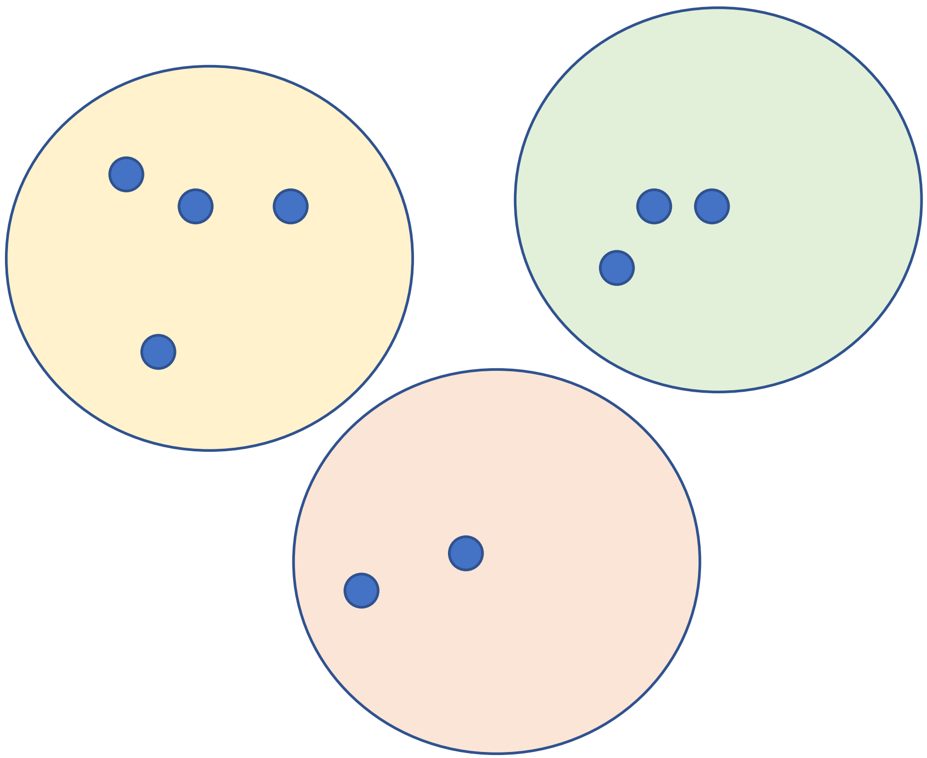
Clustering

Andrea Pasini
Flavio Giobergia

DataBase and Data Mining Group

Clustering

- **Unsupervised** technique that analyzes the data distribution to generate N partitions
  - Unsupervised = it only requires a features matrix



- Import a model

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

- Build model object

```
km = KMeans(n_clusters = 5)
```

  - The hyperparameter **n_clusters** specifies the number of centroids (= number of clusters)
    - Default is 8 (buy may change across different library versions)

Clustering

- Apply clustering to input data

In [1]:
```
y_pred = km.fit_predict(X)
```

Out[1]:
```
[3, 1, 1, 1, 2, 2, 0]
```

- This operation assigns data to their respective cluster
  - X is the 2D NumPy array with input features (**features matrix**)
  - y_pred is a 1D array with cluster labels

1.0	5	1.5
1.4	10	0.3
…	…	…

3
1
1

Clustering

- Example: DBSCAN

```
from sklearn.cluster import DBSCAN
cl_alg = DBSCAN(eps=3, min_samples=2)
```

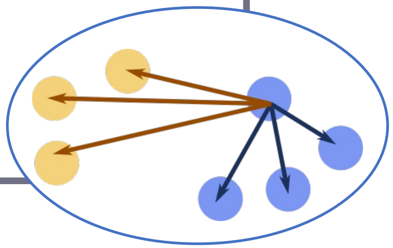
- Example: Hierarchical clustering, n_clusters=5, average linkage

```
from sklearn.cluster import AgglomerativeClustering
cl_alg = AgglomerativeClustering(5, linkage='average')
```

- Note: Agglomerative clustering and DBSCAN only support `fit()` and `fit_predict()`!
  - A “new” point cannot be predicted!

- Assessing clustering results
  - **Internal** metrics: use only the information of the features matrix
    - E.g. Silhouette, SSE

```
from sklearn.metrics import silhouette_score, silhouette_samples
silh_avg = silhouette_score(X, clusters)
silh_i = silhouette_samples(X, clusters)
```

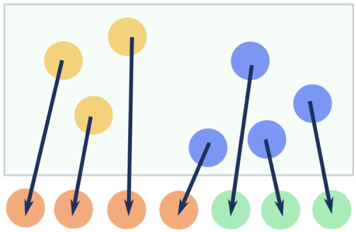


  - **Silhouette** is a number in the range [−1, 1]
  - Higher values mean higher cluster quality
    - Clusters are well separated and cohesive
  - Expensive computation! $O(n^2)$

Clustering

- Assessing clustering results
  - **External** metrics: compare a clustering result with some ground-truth labels
    - E.g. Adjusted Rand Score, Fowlkes-Mallows index

```
from sklearn.metrics import adjusted_rand_score
ars = adjusted_rand_score(c_truth, c_pred)
```



  - The ARS score ranges in "[0, 1]"
    - ~ 0: randomly assigned clusters
    - 1: perfect agreement
    - [!] Values < 0 may occur if cluster assignments are worse than random
  - It is close to 1 when data in the predicted clusters is grouped in a similar way compared with ground truth

Clustering

- Adjusted Rand Score (ARS)
  - Does not check for equality of target and predictions
  - It checks whether data are **clustered in the same way**
  - Example:
    - c_truth = [1, 1, 2, 2, 2, 1]
    - c_pred = [2, 2, 1, 1, 1, 2]
    - ARS(c_truth, c_pred) is 1

Notebook Examples

- **4d-Scikitlearn-Clustering.ipynb**