W4995 Applied Machine Learning

Evaluating Clustering

03/25/17

Andreas Müller

Evaluation Measures

Supervised evaluation

- Compare clustering against “ground truth”.
- Usually classification datasets.
- Often used for papers on clustering algorithms.
- Unclear what the assumptions are.
- Impossible to do in practice.
- Compare partitions, not labels!

Thinking in Partitions

- Y=[0, 0, 0, 1, 1, 1] → C={{0, 1, 2}, {3, 4, 5}}
- Y=[0, 1, 0, 1, 2, 2] → C={{0, 2}, {1, 3}, {4, 5}}

- Partition given by [1, 0, 1, 0] and [0, 1, 0, 1] are the same!

Contingency matrix

- Same as confusion matrix – but different!
- Rows correspond to one partition, columns the other.
- Doesn't need to be square.
- Both of these mean identical partitions:

10	0
0	10

0	10
10	0

Rand Index

- Computed of partitions C1, C2 over the same elements (say {0, ..., n-1})

Number of pairs of points in the same set in C1 and in C2

Number of pairs of points in different sets in C1 and in C2

$$R(C_1, C_2) = \frac{a + b}{\binom{n}{2}}$$

Number of possible pairs

Between 0 and 1

Rand Index Example

$$
\begin{aligned}
&R([0,0,1,1],[1,1,0,0]) \\
=&R\big\{\{0,1\},\{2,3\}\big\},\big\{\{0,1\},\{2,3\}\big\} \\
=&\frac{2+4}{6} = 1
\end{aligned}
$$

$$
\begin{aligned}
&R([0,1,0,1],[1,2,0,0]) \\
=&\big\{\{0,2\},\{1,3\}\big\},\big\{\{0,1\},\{2\},\{3\}\big\}) \\
=&\frac{0+3}{6} = \frac{1}{2}
\end{aligned}
$$

Adjusted Rand Index (ARI)

$$\text{AdjustedIndex} = \frac{\text{Index} - \text{ExpectedIndex}}{\text{MaxIndex} - \text{ExpectedIndex}}$$

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}$$

Maximum of 1, 0 on expectation (can become negative!)
[Don’t learn the formula]

Mutual Information

$$MI(U,V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} P(i,j) \log \frac{P(i,j)}{P(i)P'(j)}$$

$$= \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{|U_i \cap V_j|}{|U_i||V_j|}$$

Between 0 and 1 but not normalized or adjusted.
If V is subpartition of U still = 1

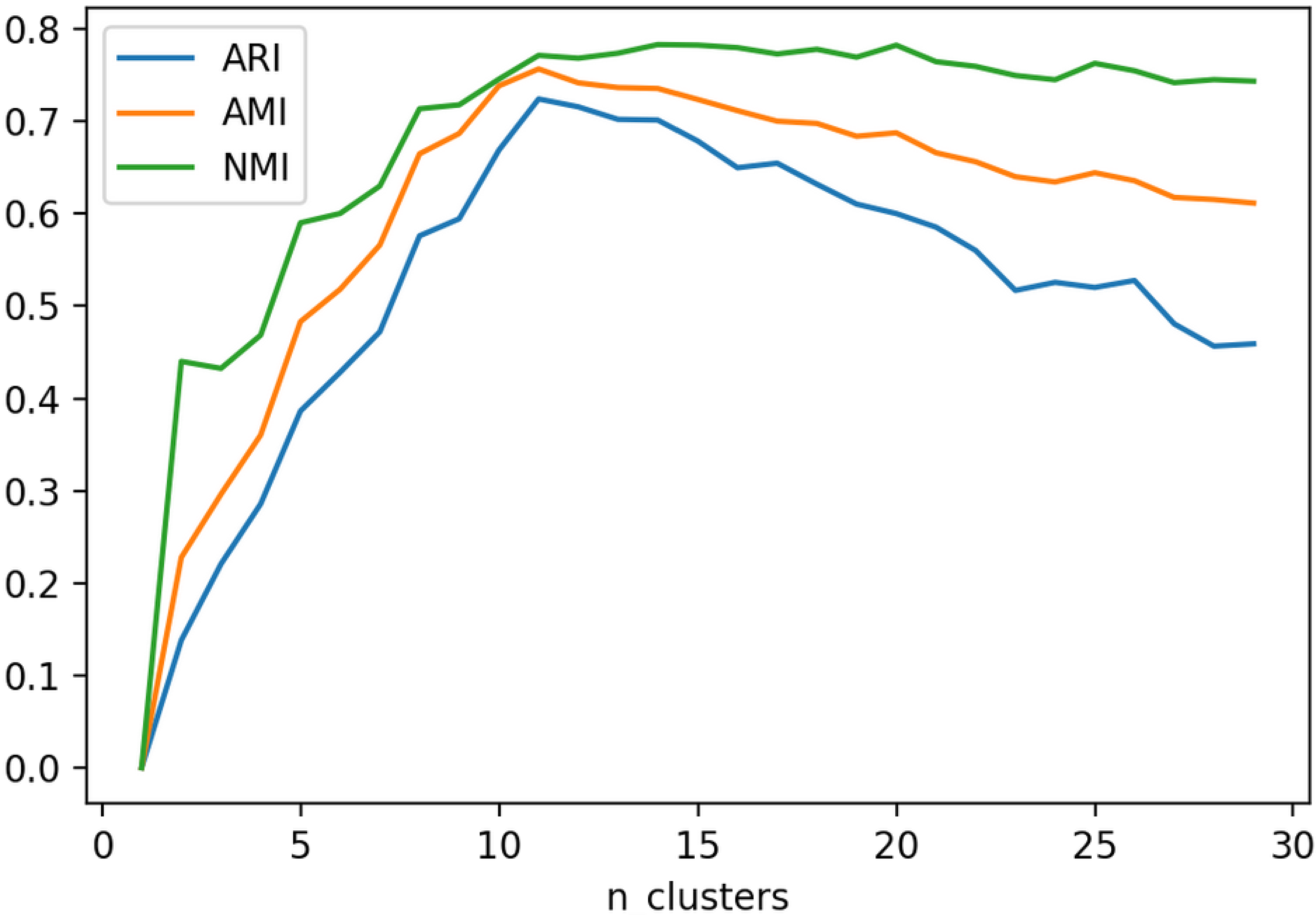
NMI and AMI

$$\mathrm{NMI}(U,V) = \frac{\mathrm{MI}(U,V)}{\sqrt{H(U)H(V)}}$$

Penalizes overpartitioning (via entropy)

$$\mathrm{AMI} = \frac{\mathrm{MI} - E[\mathrm{MI}]}{\max(H(U),H(V)) - E[\mathrm{MI}]}$$

Adjusts for chance – any two random partitions have expected AMI of 0



ARI, AMI and NMI for k-Means on “digits” dataset. ARI and AMI penalize too many clusters.

Unsupervised Evaluation

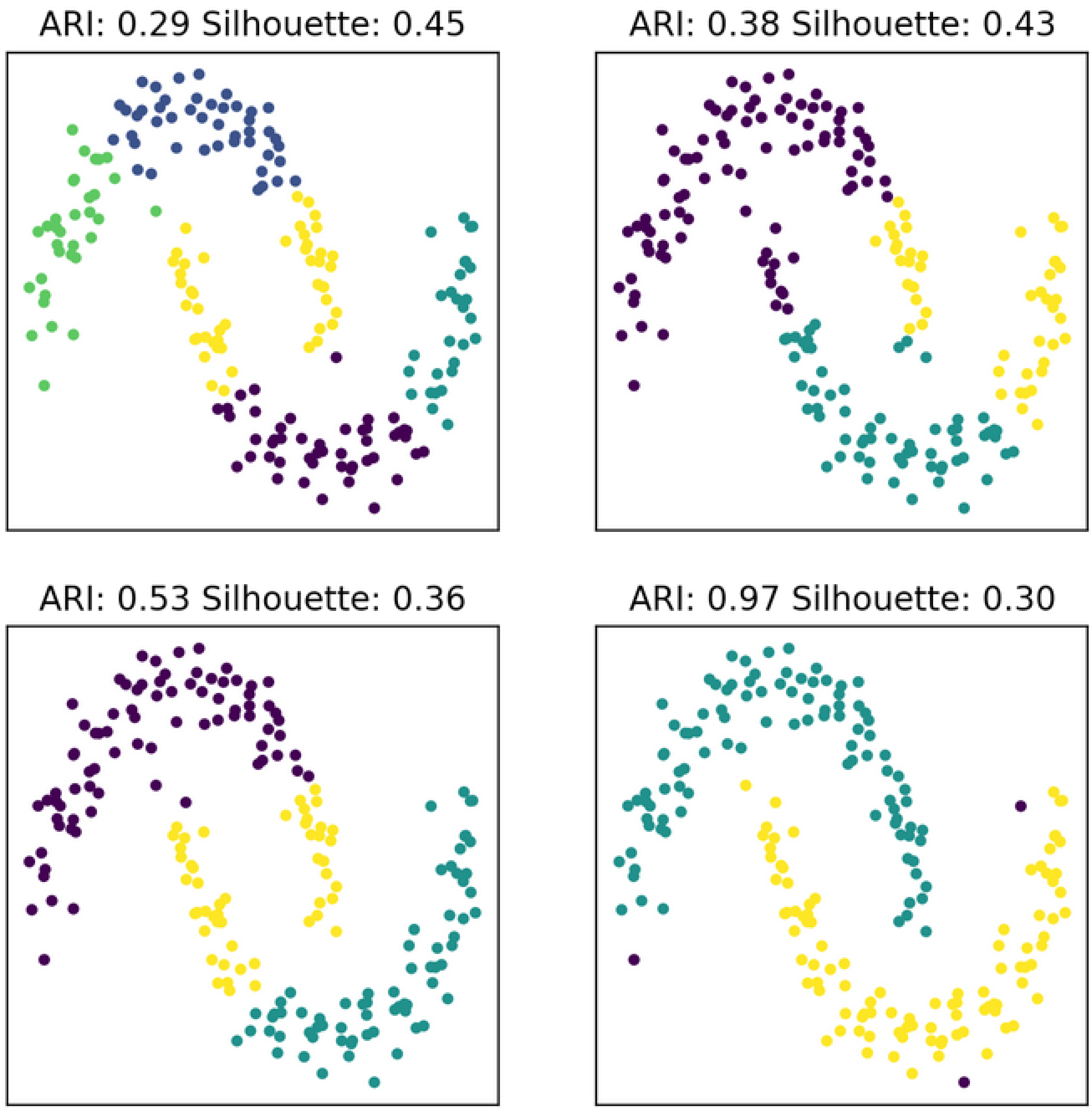
Silhouette Score

- For each sample:

$$s = \frac{b - a}{max(a, b)}$$

- a is mean distance to samples in same cluster
- b is mean distance to samples in nearest cluster

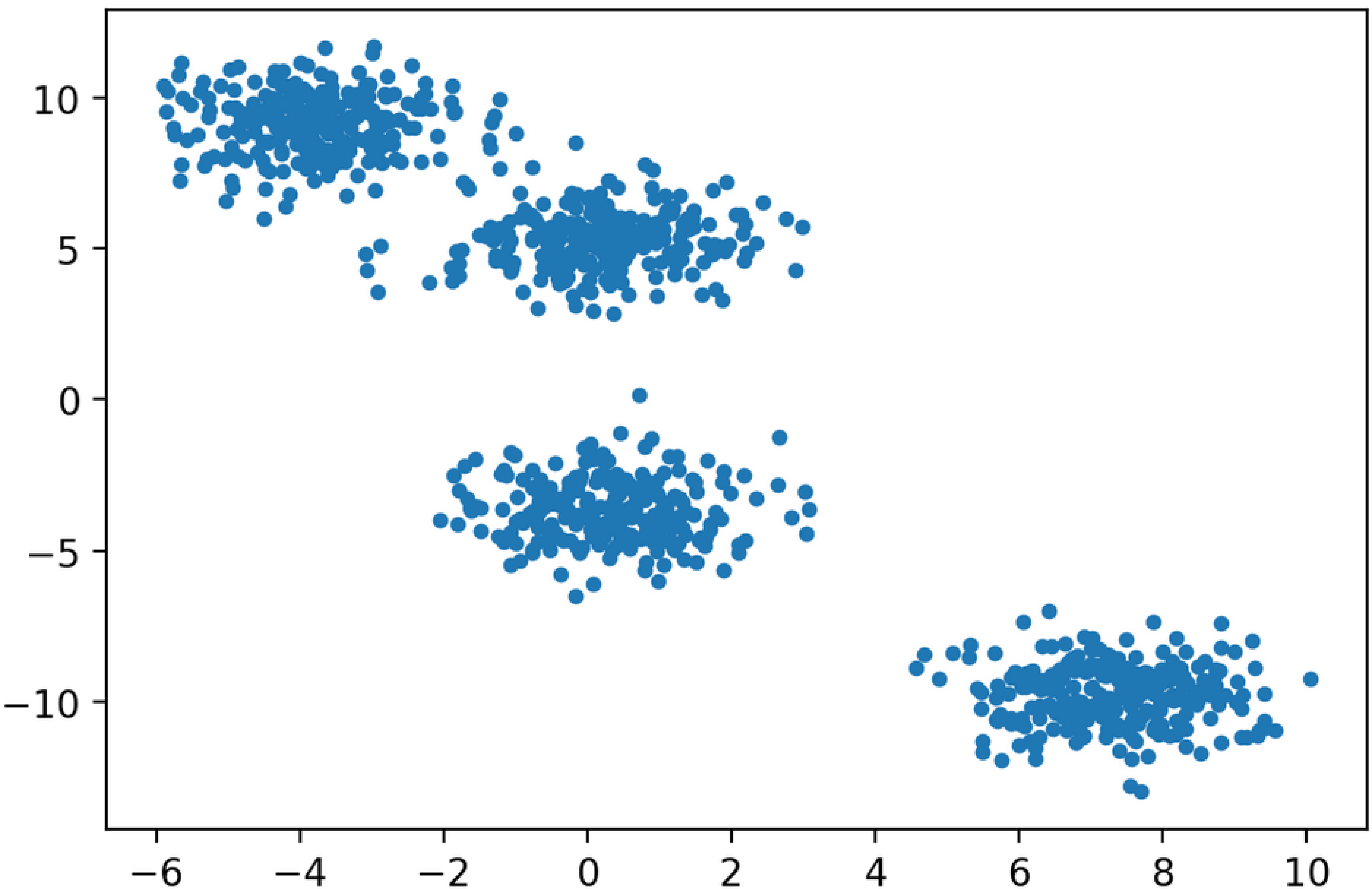
- For whole clustering: average over all samples
- Prefers compact clusters (like k-means)


ARI: 0.29 Silhouette: 0.45
ARI: 0.38 Silhouette: 0.43
ARI: 0.53 Silhouette: 0.36
ARI: 0.97 Silhouette: 0.30

Picking the number of clusters

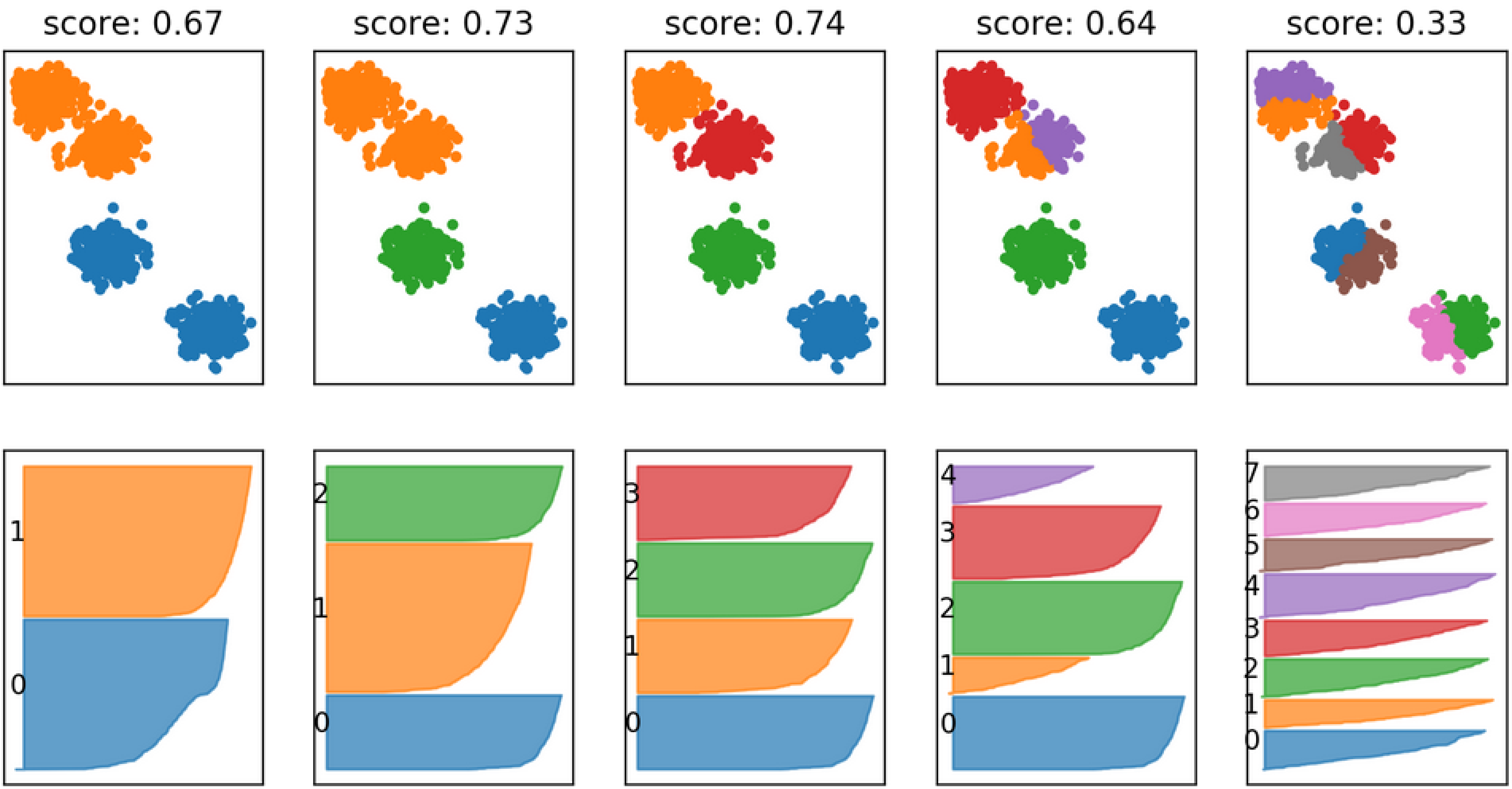
With metrics

- Either cross-validation or evaluated on whole dataset.
- If you have the label, why are your doing clustering?

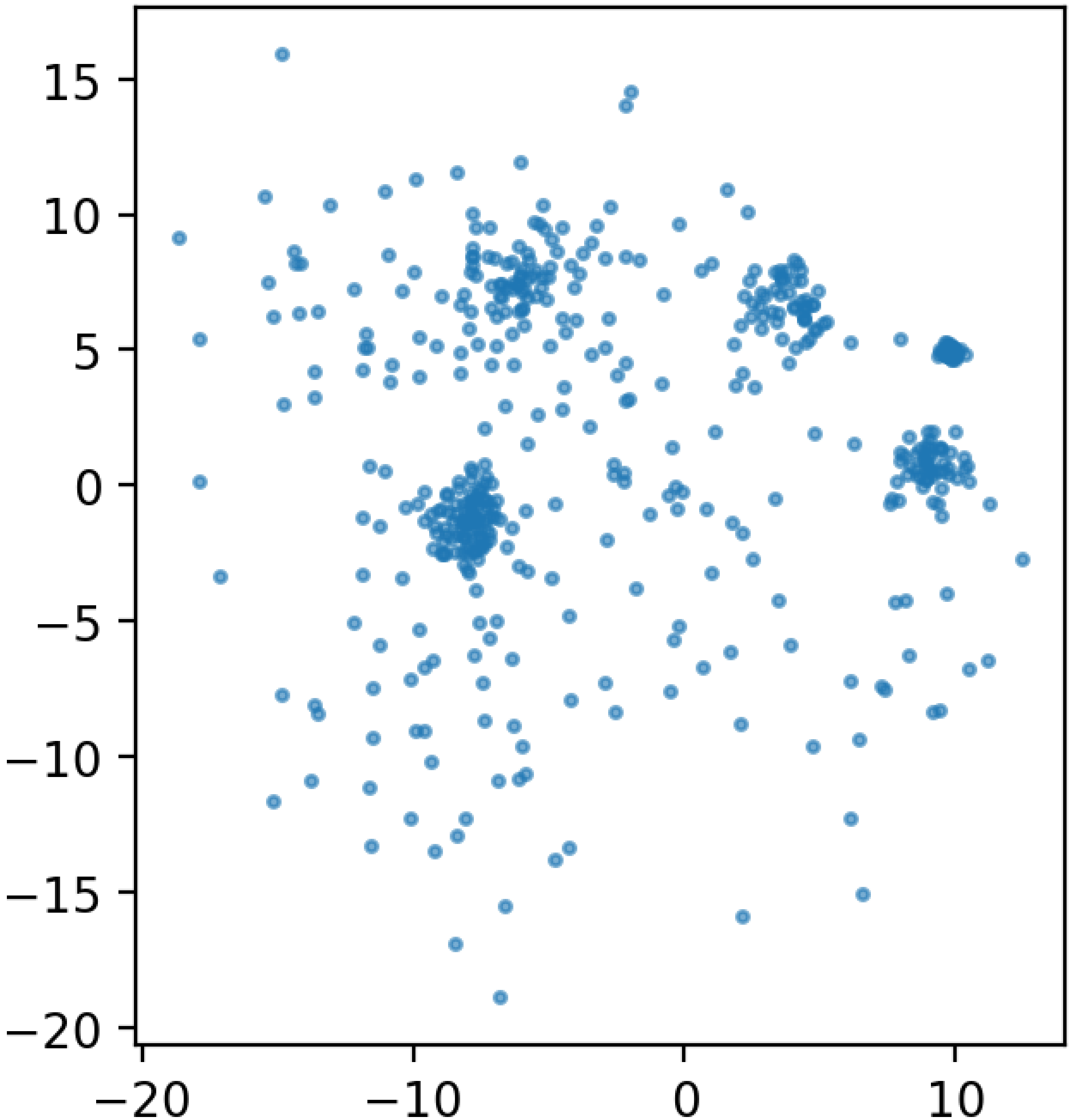


What's the right number of clusters?

Silhouette Plots

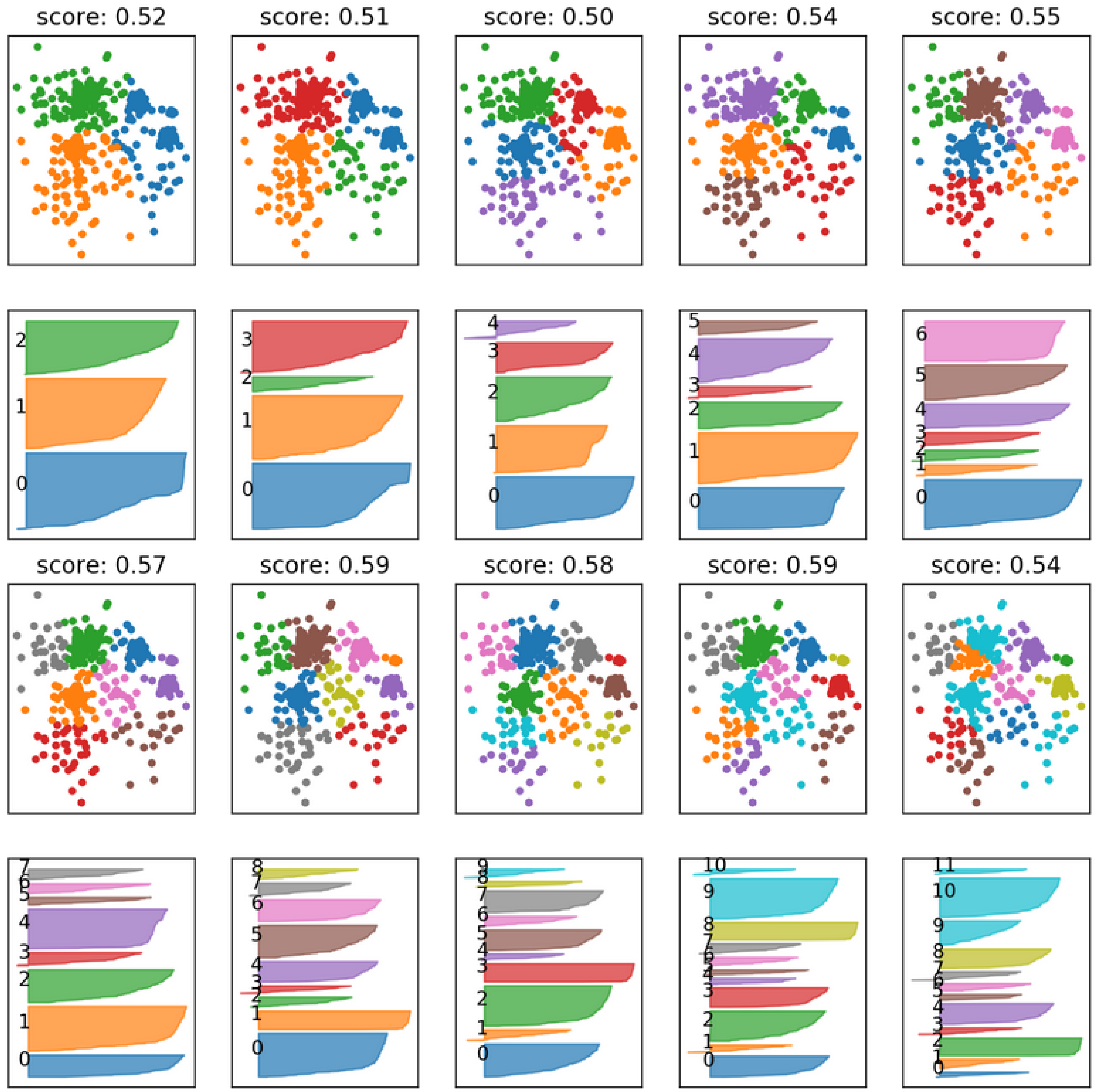


Plot sorted silhouette score for each sample in each cluster
http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

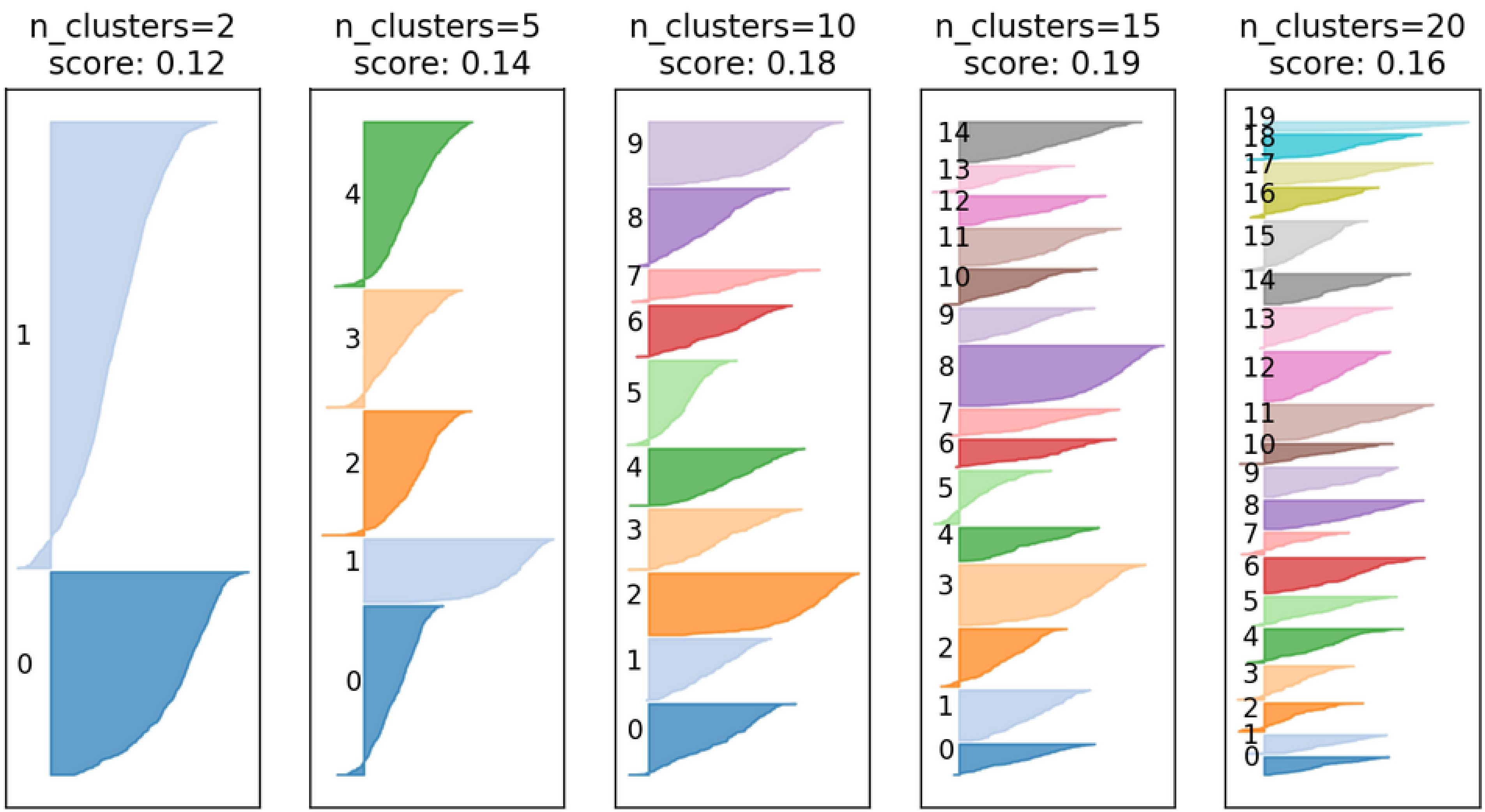


What's the right number of clusters?

Generated as mixture of 10 Gaussians.
So what?


score: 0.52
score: 0.51
score: 0.50
score: 0.54
score: 0.55
score: 0.57
score: 0.59
score: 0.58
score: 0.59
score: 0.54

Digits dataset

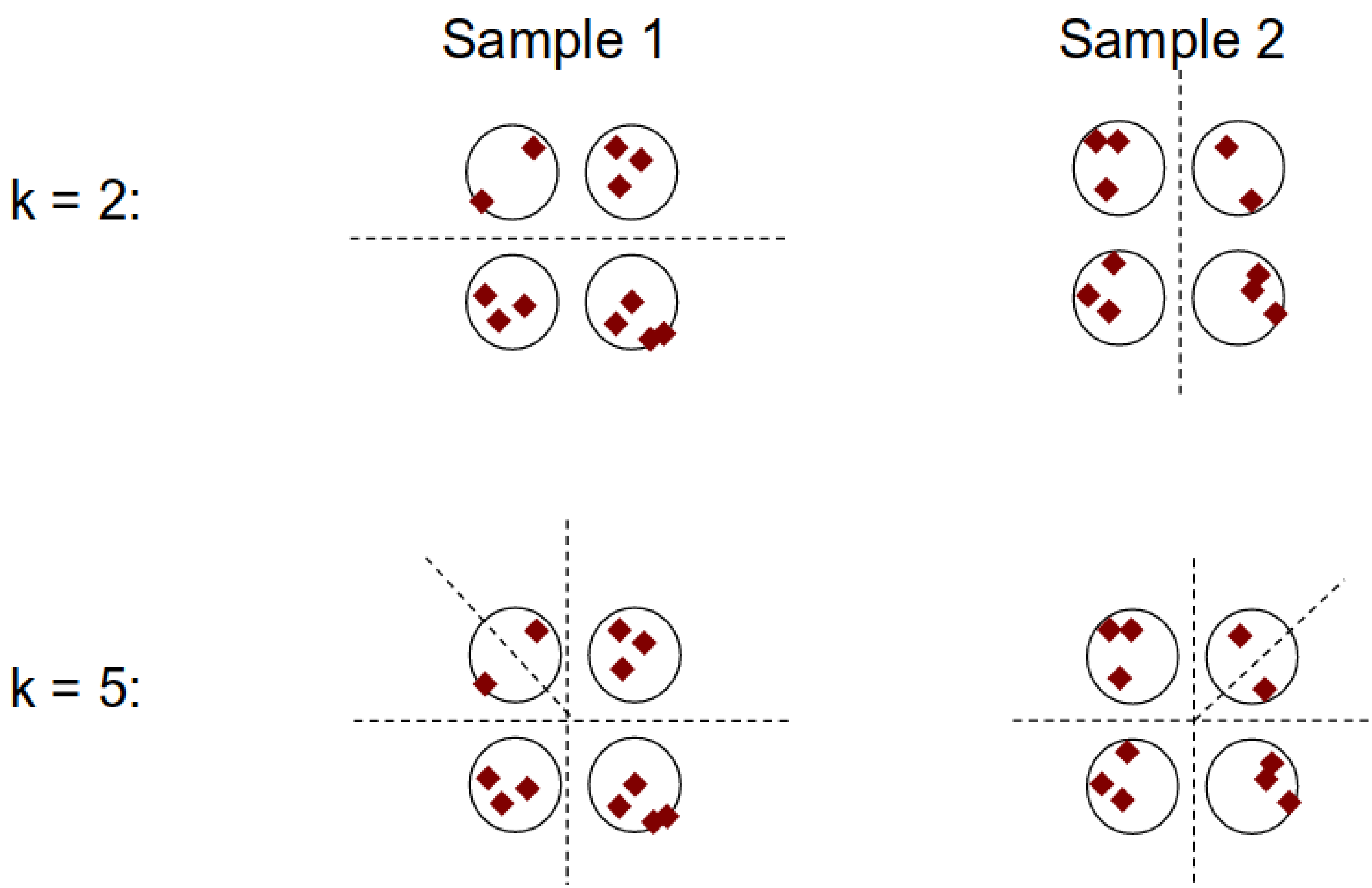


Custer Stability

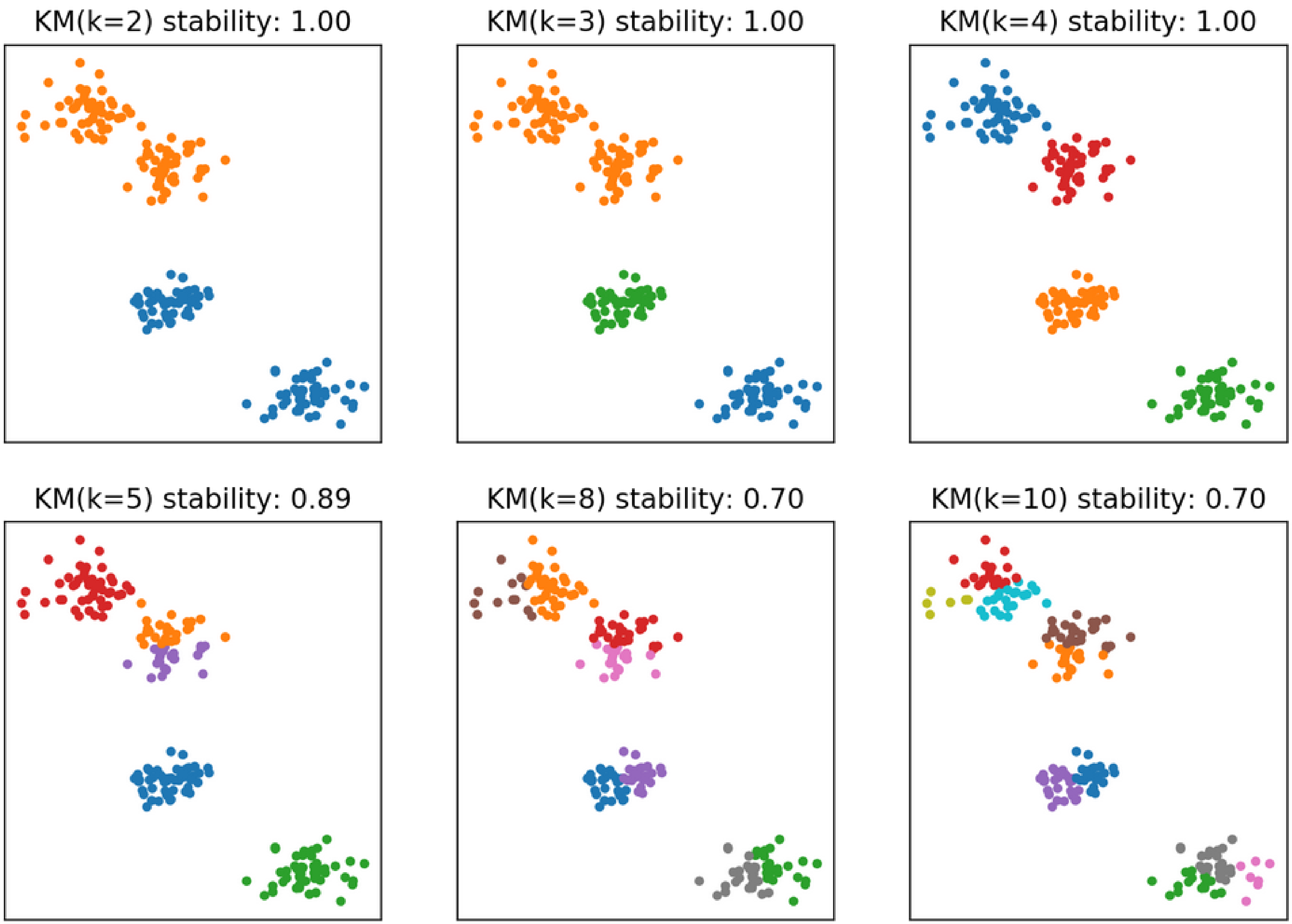
- For comparing clustering algorithms / parameters (with different number of clusters for example)
- Clustering stability: an overview (U. v. Luxburg)

  https://arxiv.org/abs/1007.1075
- Try multiple random samplings / perturbations
- The configuration that yields the most consistent result among perturbations is best.

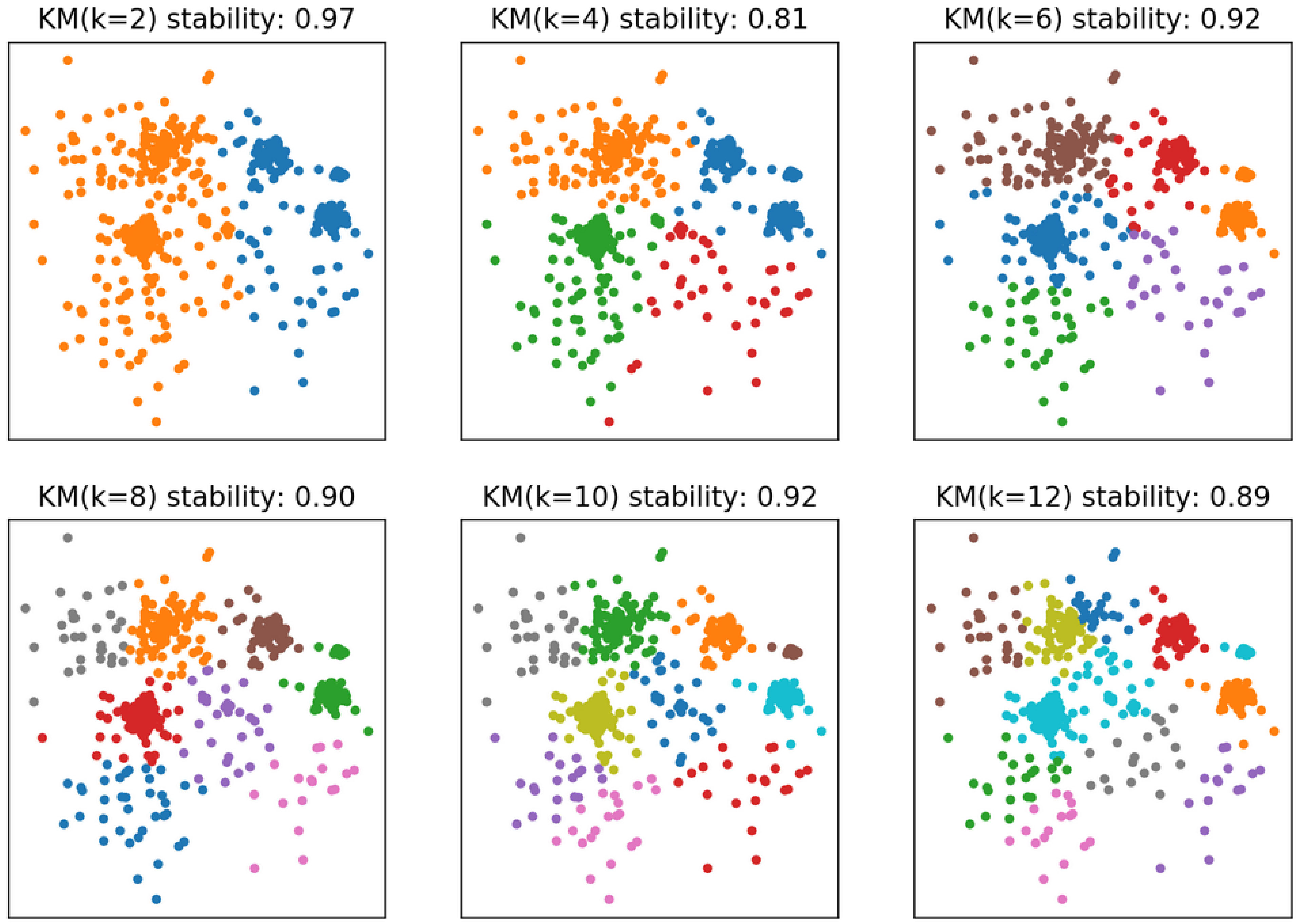
Stability Intuition



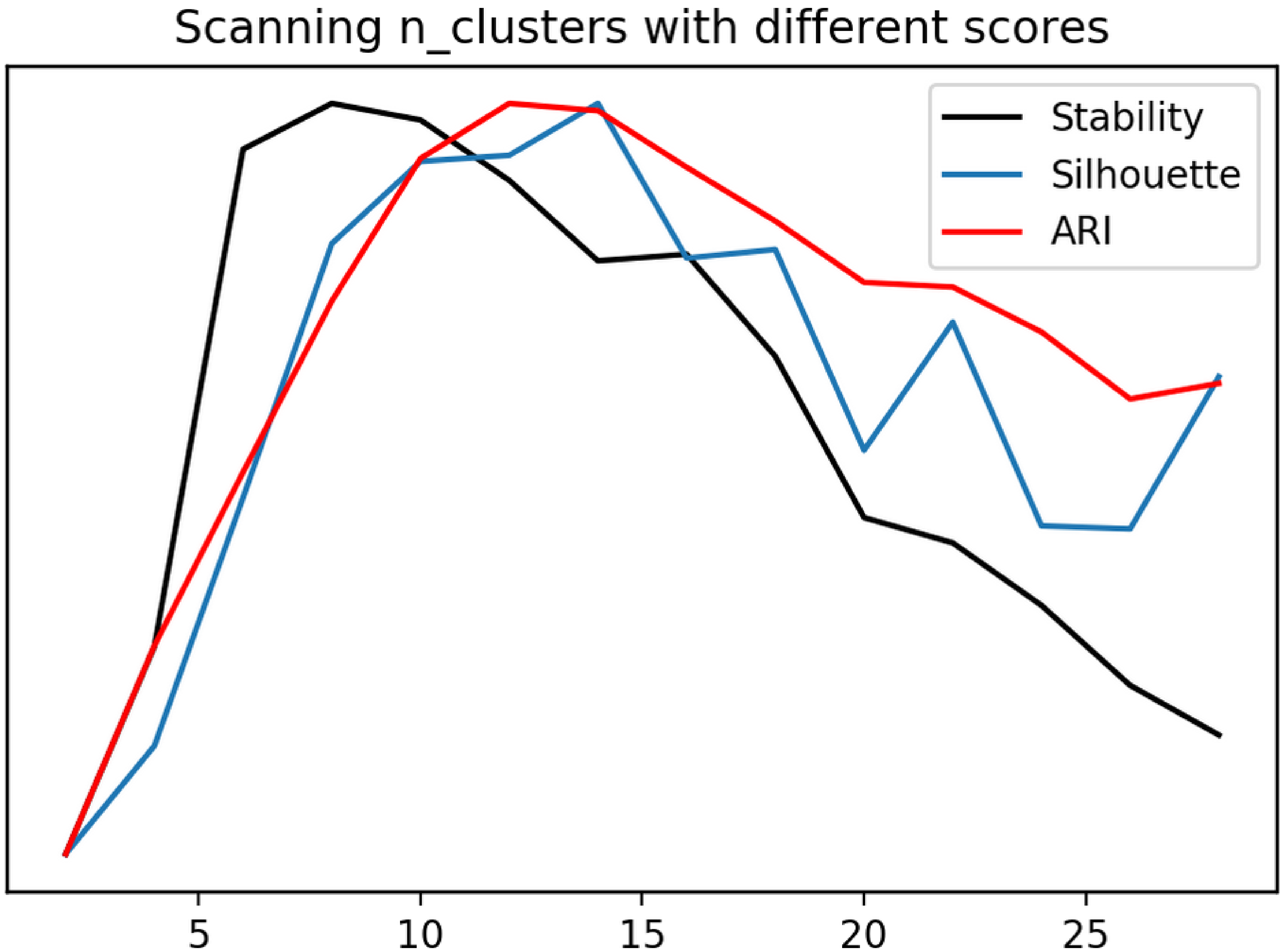
```python
def cluster_stability(X, est, n_iter=20, random_state=None):
    labels = []
    indices = []
    for i in range(n_iter):
        # draw bootstrap samples, store indices
        sample_indices = rng.randint(0, X.shape[0], X.shape[0])
        indices.append(sample_indices)
        est = clone(est)
        if hasattr(est, "random_state"):
            # randomize estimator if possible
            est.random_state = rng.randint(1e5)
        X_bootstrap = X[sample_indices]
        est.fit(X_bootstrap)
        # store clustering outcome using original indices
        relabel = -np.ones(X.shape[0], dtype=np.int)
        relabel[sample_indices] = est.labels_
        labels.append(relabel)
    scores = []
    for l, i in zip(labels, indices):
        for k, j in zip(labels, indices):
            # we also compute the diagonal which is a bit silly
            in_both = np.intersect1d(i, j)
            scores.append(adjusted_rand_score(l[in_both], k[in_both]))
    return np.mean(scores)
```



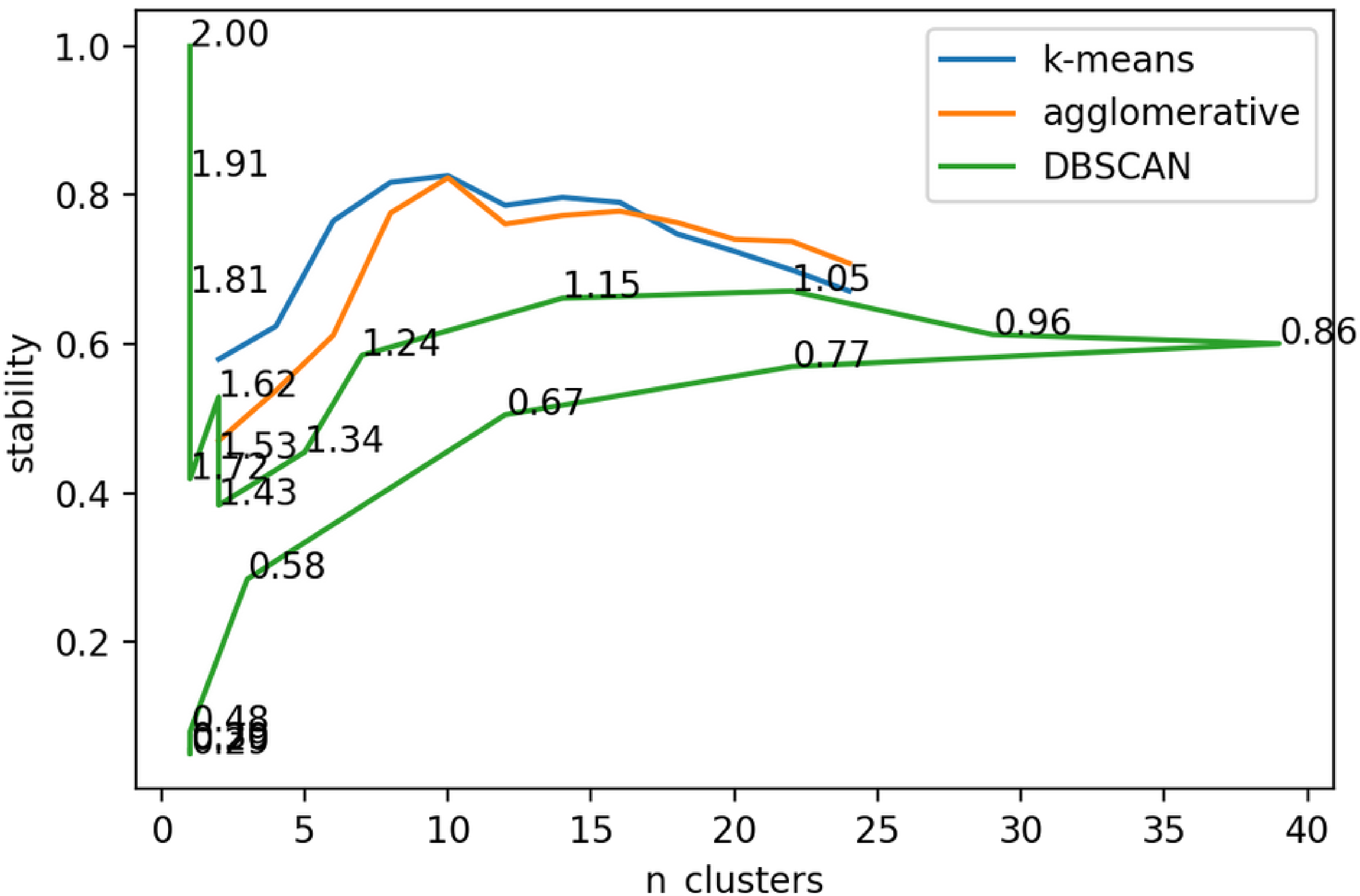
Outcome depends critically on initialization and n_iter.
This is kmeans++ with n_iter=10


KM(k=2) stability: 0.97
KM(k=4) stability: 0.81
KM(k=6) stability: 0.92
KM(k=8) stability: 0.90
KM(k=10) stability: 0.92
KM(k=12) stability: 0.89

digits dataset

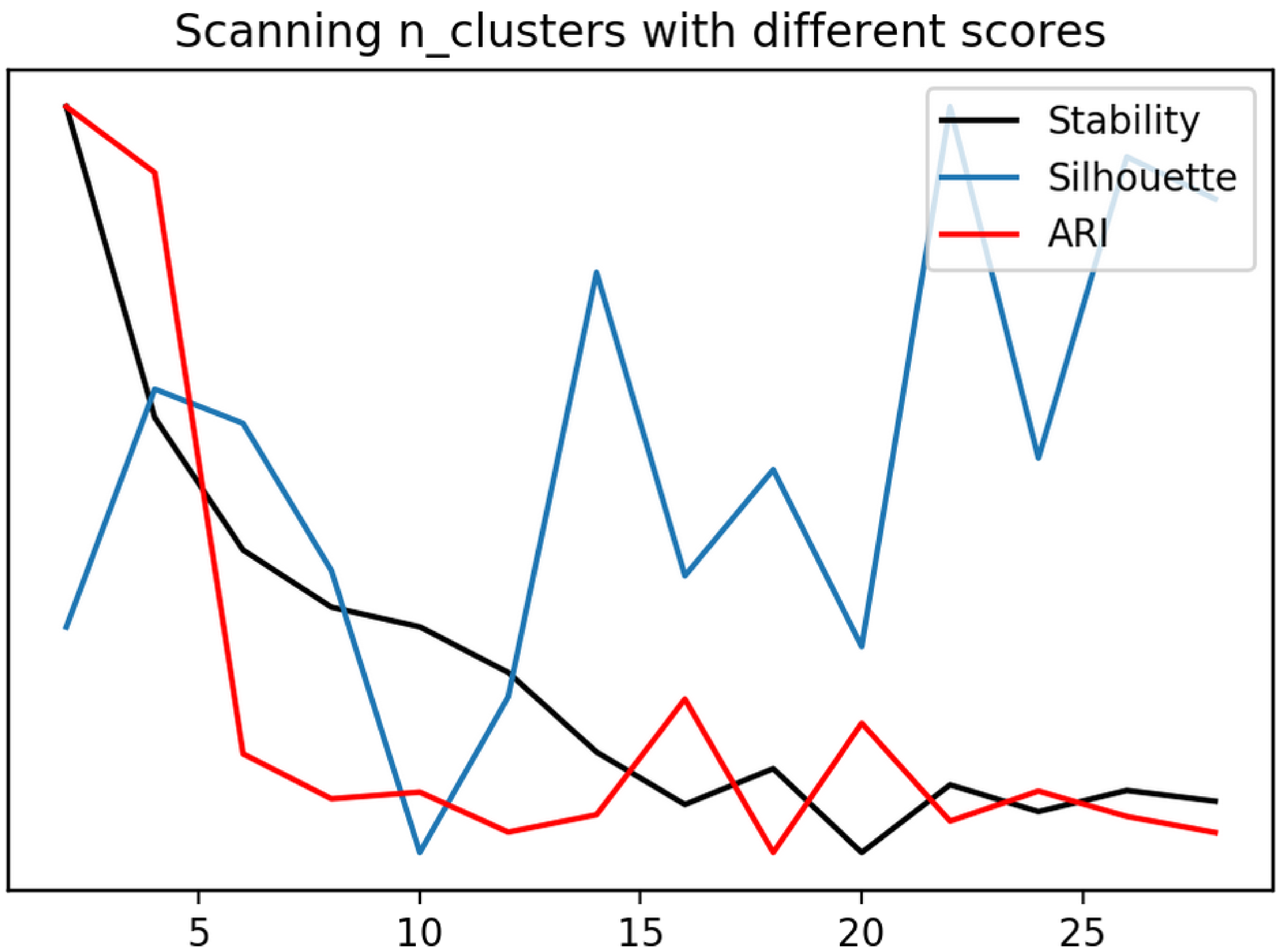


Stability of different algorithms

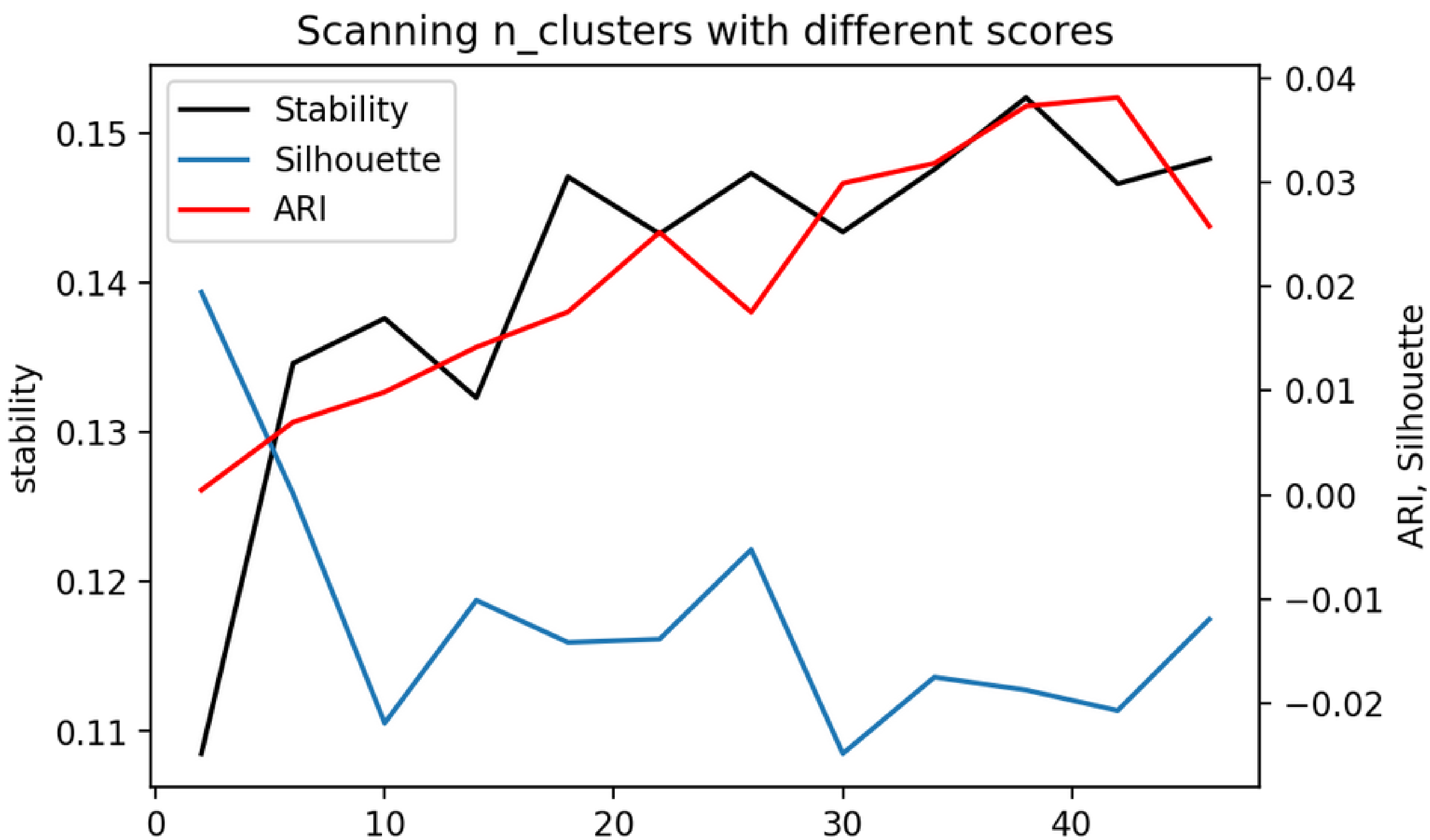


Digits dataset

Adult Dataset



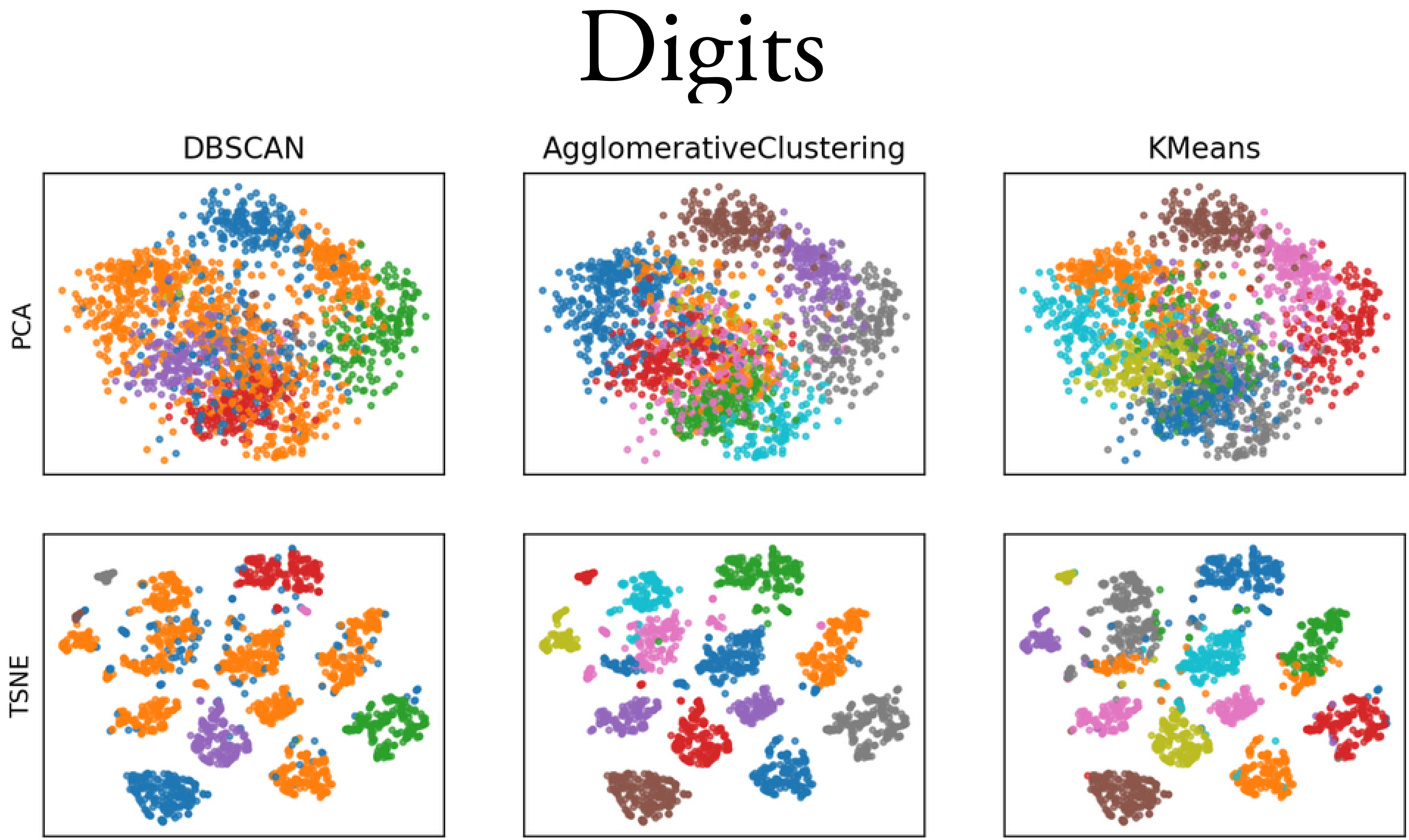
Labeled Faces in the Wild



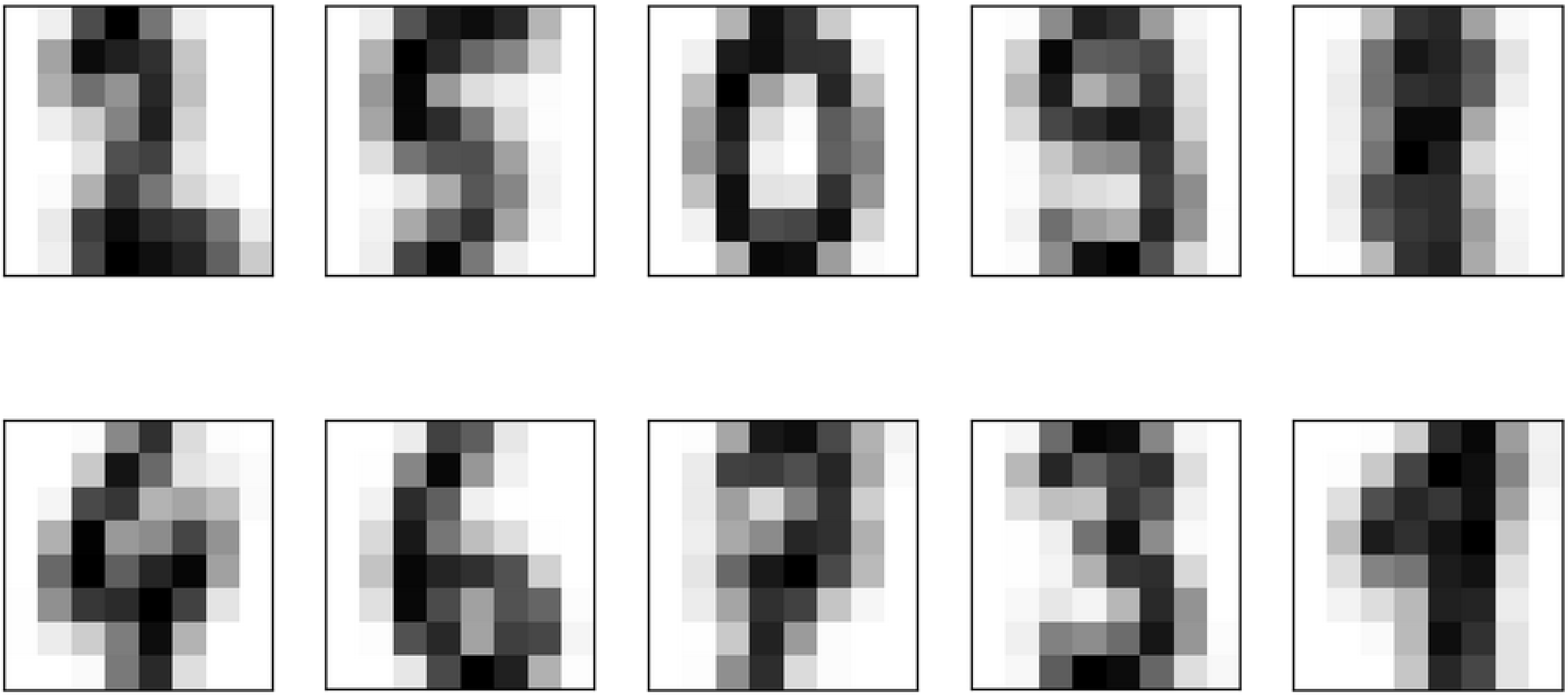
With PCA, 100 components. Without whitening: no stable clusters!

Qualitative Evaluation (aka eyeballing)

- Look at low-dim visualization
- Look at individual points
- Look at cluster centers (if available)

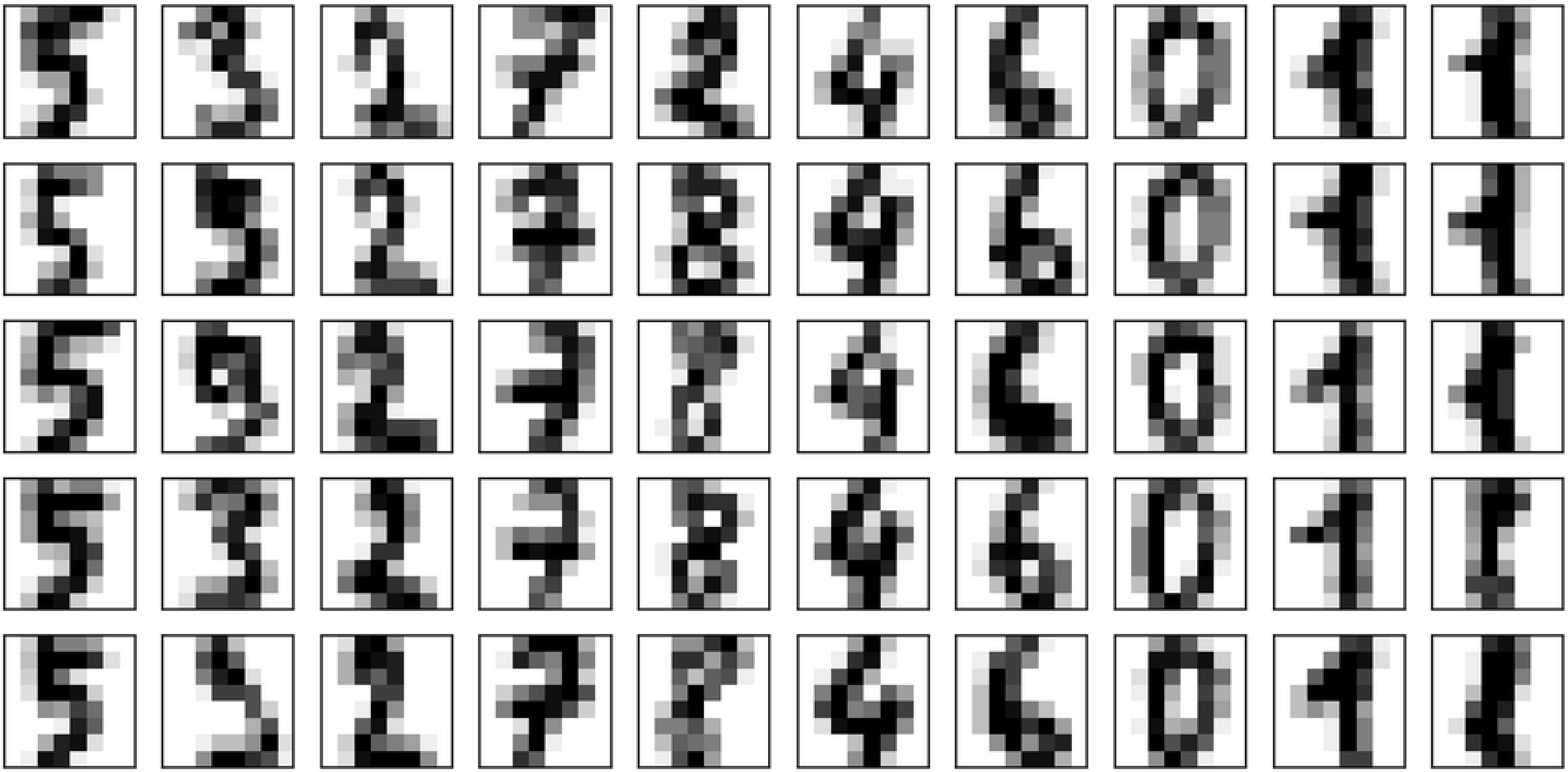

Digits
DBSCAN
AgglomerativeClustering
KMeans
PCA
TSNE

K-Means cluster centers



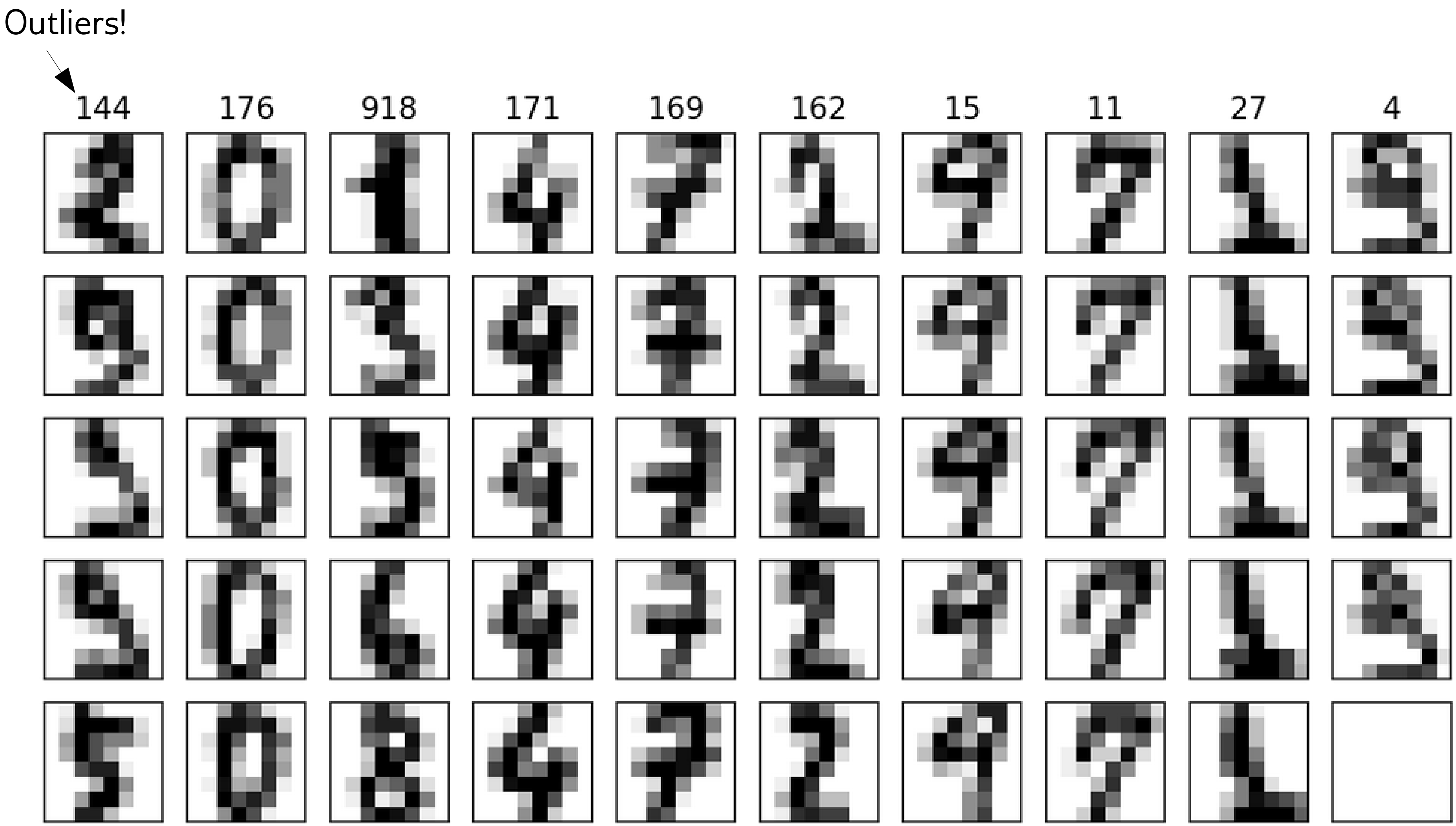
```python
fig, axes = plt.subplots(2, 5, subplot_kw={'xticks': (), 'yticks': ()}, figsize=(10, 5))
km = KMeans(n_clusters=10).fit(digits.data)
for ax, center in zip(axes.ravel(), km.cluster_centers_):
    ax.imshow(center.reshape(8, 8), cmap='gray_r')
fig.suptitle("K-Means cluster centers")
```

Agglomerative Clusters



Each column corresponds to one of 10 clusters, “first” 5 samples are shown.

DBSCAN Clusters



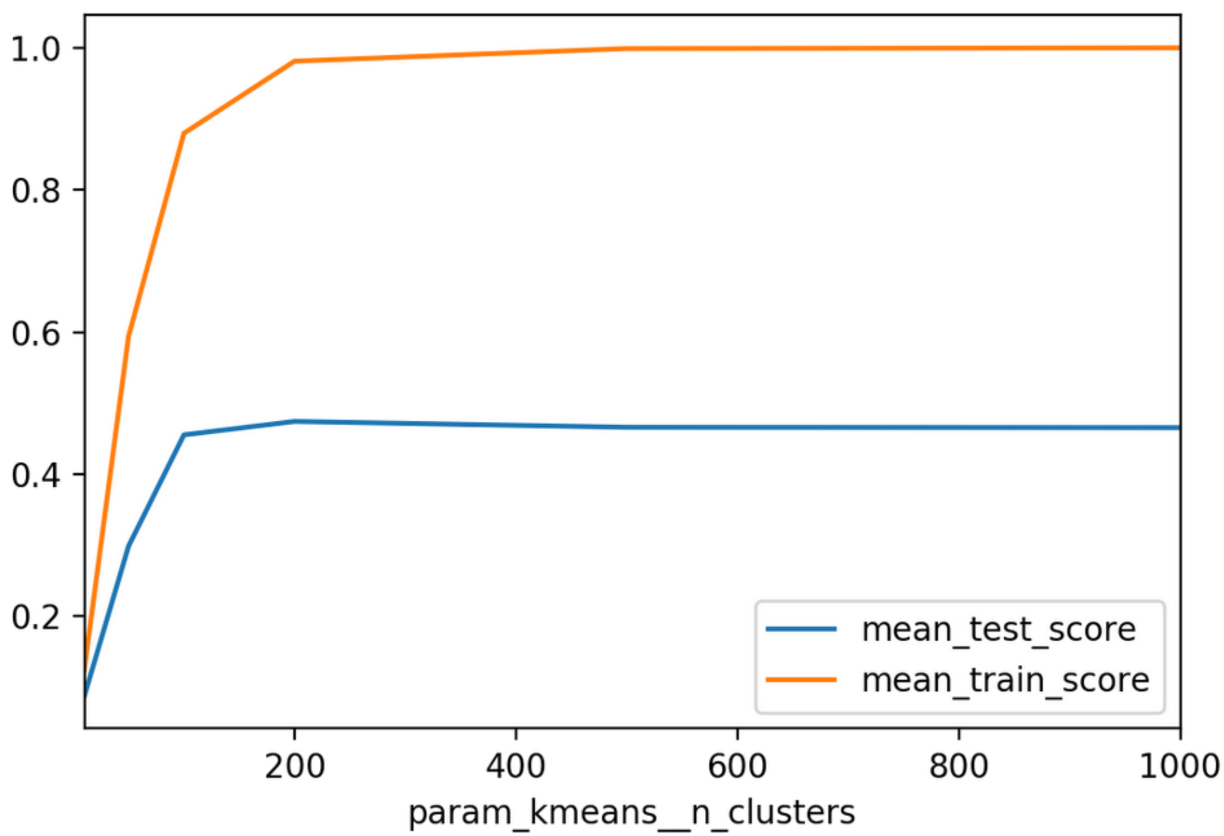
Each column corresponds to one of 10 clusters, “first” 5 samples are shown.
Number on top gives clusters size

For feature extraction

```python
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV

km = KMeans(n_init=1, init="random")
pipe = make_pipeline(km, LogisticRegression())

param_grid = {'kmeans__n_clusters': [10, 50, 100, 200, 500]}
grid = GridSearchCV(pipe, param_grid, cv=5, verbose=True)
grid.fit(X, y_people)
```



So what is n_clusters?

- As preprocessing
  - The one that makes the whole pipeline work best.
  - Larger is often better.
- For exploratory analysis:
  - The one that tells you the most about the data.

- For most datasets, there is no “correct” number of clusters.

If you want semantics from clustering,
you need manual confirmation.

Clustering might not pick up
on what you thought it would.