Unsupervised Learning

Shyue Ping Ong

University of California, San Diego

NANO281

Overview

Preliminaries

- Here, we will take a digression from the **supervised learning** that we have focused on so far, and go into **unsupervised learning**.
- In supervised learning, model development is carried out with a set of input/output examples (training data).
- In unsupervised learning, the goal is to infer the properties of a set of data (e.g., its distribution) without training examples.
- We will include dimensionality reduction techniques within the umbrella of unsupervised learning.

Supervised vs Unsupervised Learning

Supervised learning

- Learn from example inputs and outputs (labels).
- Clear metrics of success (e.g., maximum likelihood, MSE, MAE, etc.)
- Computationally efficient.

Unsupervised Learning

- Learn only from inputs.
- No rigorously-defined metric of success.
- Computationally complex.

Principal Component Analysis (PCA)

- Briefly alluded to in lecture on Linear Methods (regressing on derived input directions).
- Consider a dataset that has dimension p. The principal components provide a sequence of best linear approximations to that data, of all ranks $q \leq p$.
- Let the observations be $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}$. The rank q linear model for representing this data is given by:

$$f(\lambda) = \mu + \mathbf{V_q}\lambda$$

- μ is a location vector, $\mathbf{V_q}$ is a $p \times q$ matrix with q orthogonal vectors, λ is a length q vector of parameters.
- We want to minimize the "reconstruction error",

$$\min_{\mu, \mathbf{V_q}, \lambda} \sum_{i=1}^{N} ||\mathbf{x_i} - \mu - \mathbf{V_q}\lambda_\mathbf{i}||^2$$

Solution

- Minimizing wrt to μ and λ_i gives

$$\begin{aligned} \hat{\mu} &= \bar{\mathbf{x}} \\ \lambda_i &= \mathbf{V_q}^T(\mathbf{x_i} - \bar{\mathbf{x}}) \end{aligned}$$

- Need to solve:

$$\min_{\mathbf{V_q}} \sum_{i=1}^{N} ||(\mathbf{x_i} - \bar{\mathbf{x}}) - \mathbf{V_q}\mathbf{V_q}^T(\mathbf{x_i} - \bar{\mathbf{x}})||^2$$

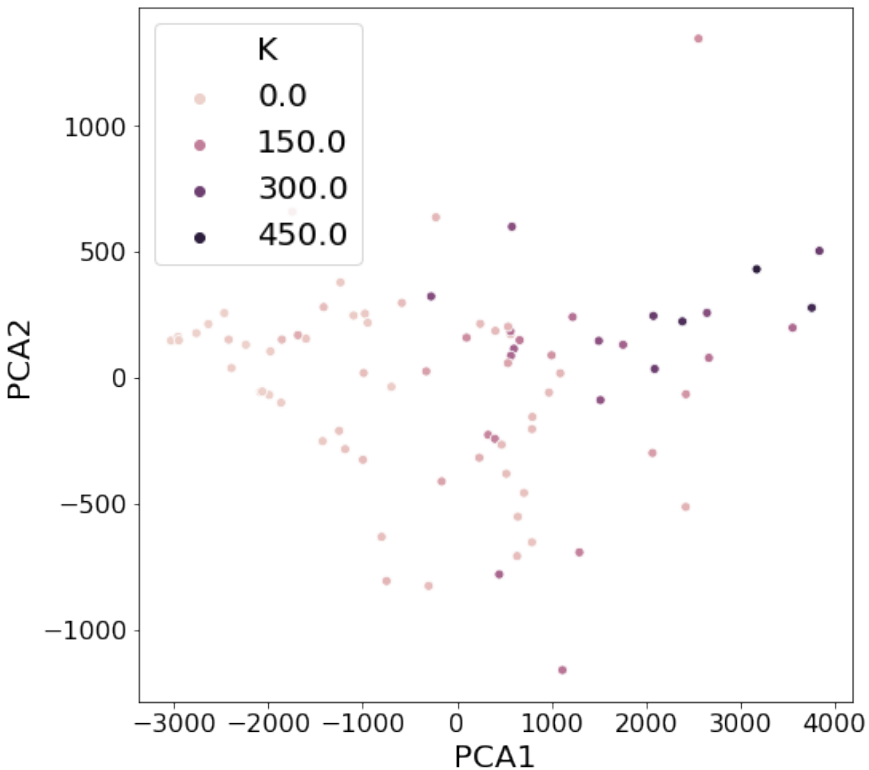
Solution, contd.

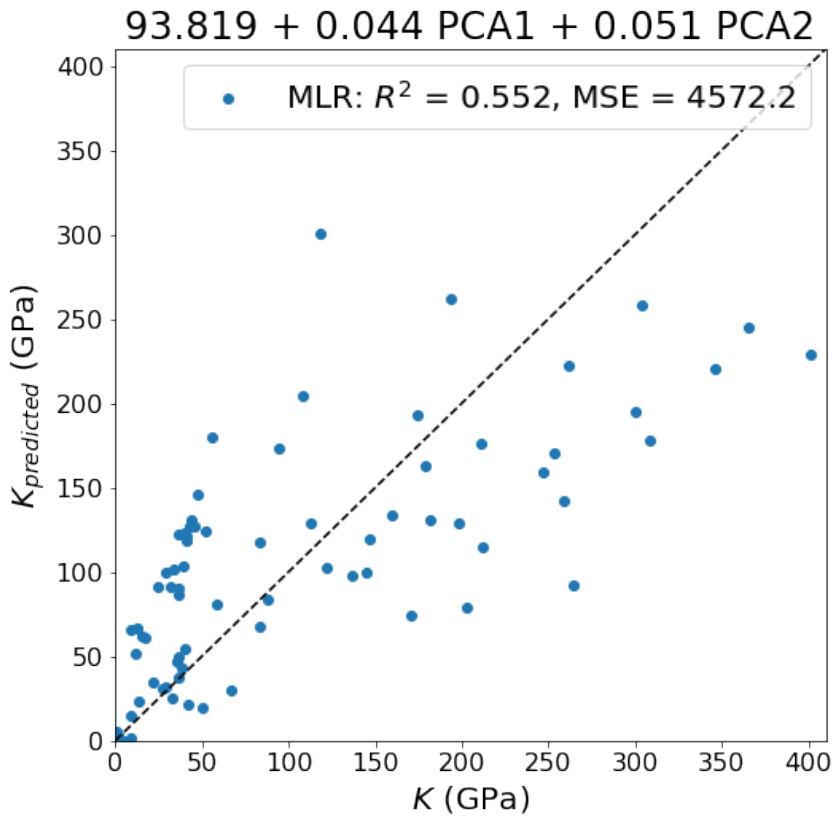
- Construct singular value decomposition (SVD) of $\mathbf{X}$, the $N \times p$ matrix of centered $\mathbf{x_i}$.

$$\mathbf{X} = \mathbf{UDV}^T$$

- $\mathbf{U}$ is an $N \times p$ orthogonal matrix, $\mathbf{D}$ is a $p \times p$ diagonal matrix with singular values $d_1 > d_2 > ... > d_p$ and $\mathbf{V}$ is $p \times p$ orthogonal matrix with right singular vectors $\mathbf{v_1}, \mathbf{v_2}, ...\mathbf{v_p}$ as columns.
- $\mathbf{UD}$ are the principal components.
- $\mathbf{Xv_1}$ has highest variance among all linear combination of features, followed by $\mathbf{Xv_2}$, $\mathbf{Xv_3}$, etc.

Linear regression of bulk modulus on first two PCAs of elemental features





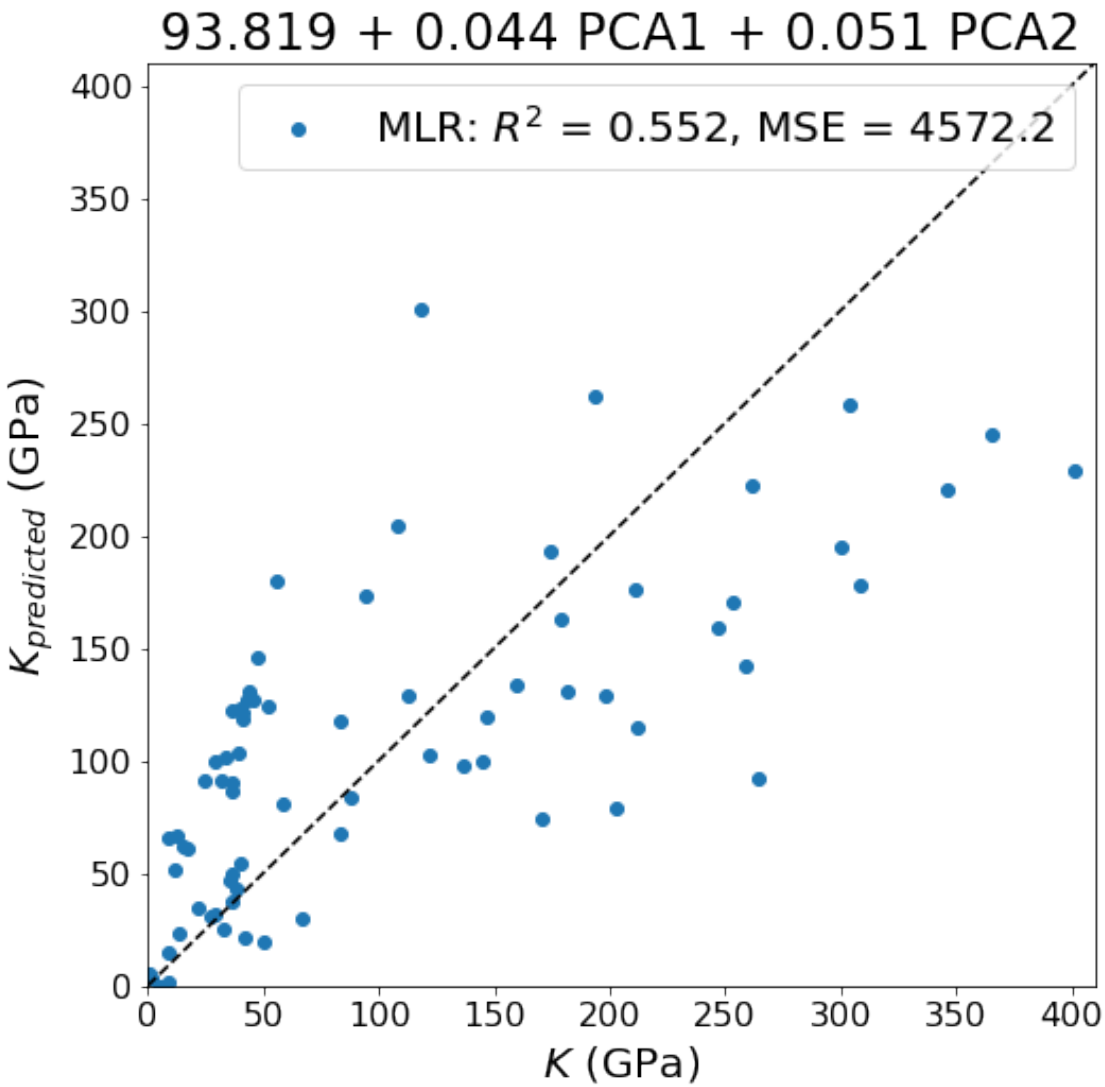
Code

```
from sklearn.decomposition import PCA
pca = PCA()
pca.fit(x)
x_pca = pca.transform(x)
print(pca.explained_variance_)

# Linear regression using PCA components
from sklearn import linear_model
from sklearn.model_selection import cross_val_predict, KFold

kfold = KFold(n_splits=5, shuffle=True, random_state=42)
mlr = linear_model.LinearRegression()
yhat_mlr = cross_val_predict(mlr, x_pca[:, 0:2], y, cv=kfold)
```

Example of linear regression on PCA components



Extensions to PCA

- Principal curves: smooth 1D curved approximation to data.
- Principal surfaces: curved 2D manifold approximation to data.

Cluster Analysis

- Cluster observations into groups so that pairwise differences within cluster tend to be smaller than differences between clusters.

  Combinatorial algorithms Model observed data with no underlying probability model.

  Mixture modeling Assumes samples are i.i.d. from some population with a probability density function.

  Mode seekers Estimate modes from PDF.

K-means

- One of the most popular iterative descent clustering methods.
- Often used with the Euclidean distance as the dissimlarity measure.

$$d(x_i, x_i') = ||x_i - x_i'||^2$$

- Classic k-means measure distance to centroids of clusters.
- Other distance metrics are possible: weighted Euclidean, periodic boundary condition distance, etc.
- Variants:
  - K-medoids: Use one of points as cluster center instead of centroid. Removes influence of large outliers that produce large distances.

K-means Algorithm

1. Initialize a set of k means (centroids), e.g., choosing k observations to be the initial means (Forgy algo) or randomly assigns a cluster to each observation (random partition).
2. Assign each observation to the cluster with the smallest distance, i.e., partition the observations using the Voronoi diagram generated by means.
3. Recalculate the new means of the observations in the new clusters.
4. Algorithm is converged when assignment no longer changes.

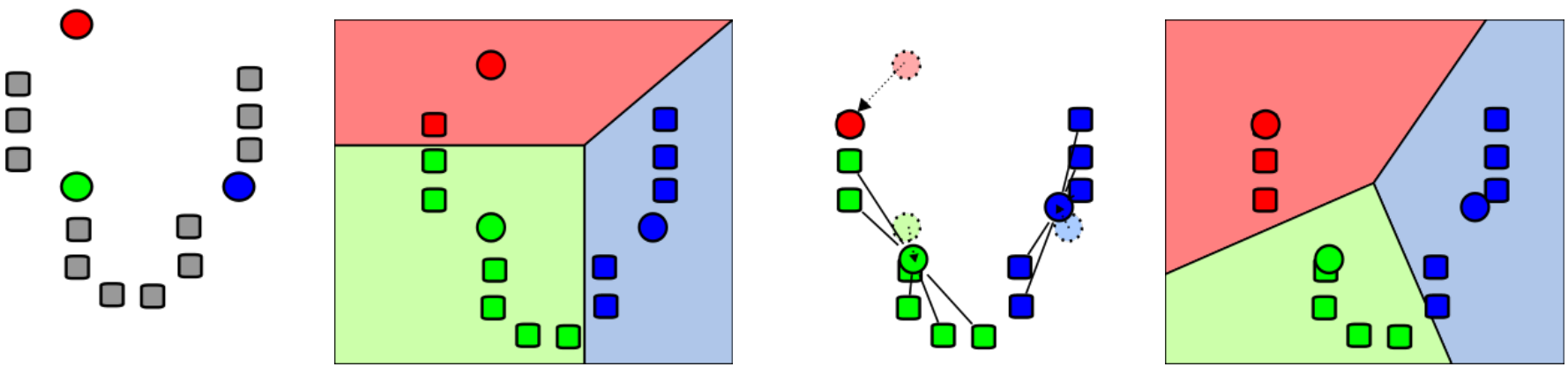
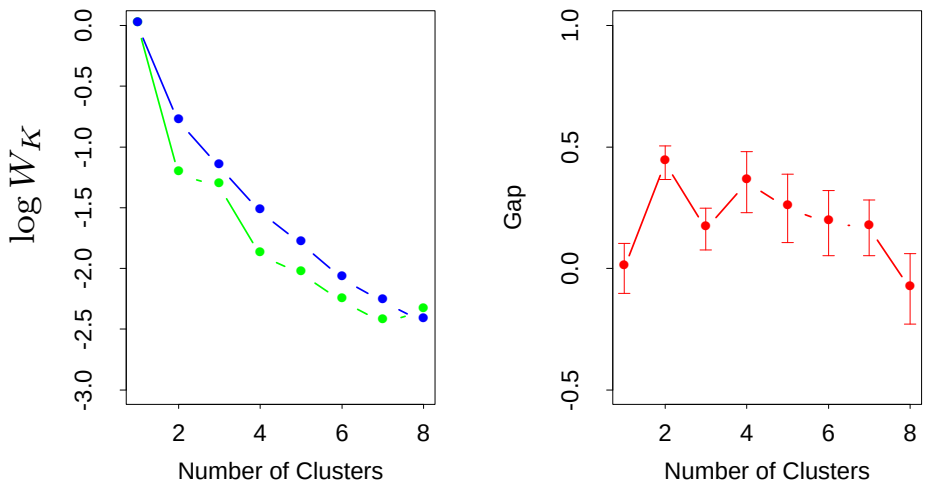


Figure: Four steps of K-means algorithm. Source: Wikipedia

Practical considerations

- K-means is often used for vector quantization, i.e., determining
- Determining K. Sometimes K is based on goals, e.g., if you have a scientific / practical reason for having K clusters, e.g., only K instruments available or you want to bin the compounds in K chemical classes.
- Gap analysis:



Hierarchical Clustering

- Does not require specific of number of clusters.
- Require dissimilarity measure.
- Produce hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level.
- Two paradigms: agglomerative (bottom-up merging) and divisive (top-down splitting).
- Typically shown in a dendrogram.

Recent application: Lithium Superionic Conductors

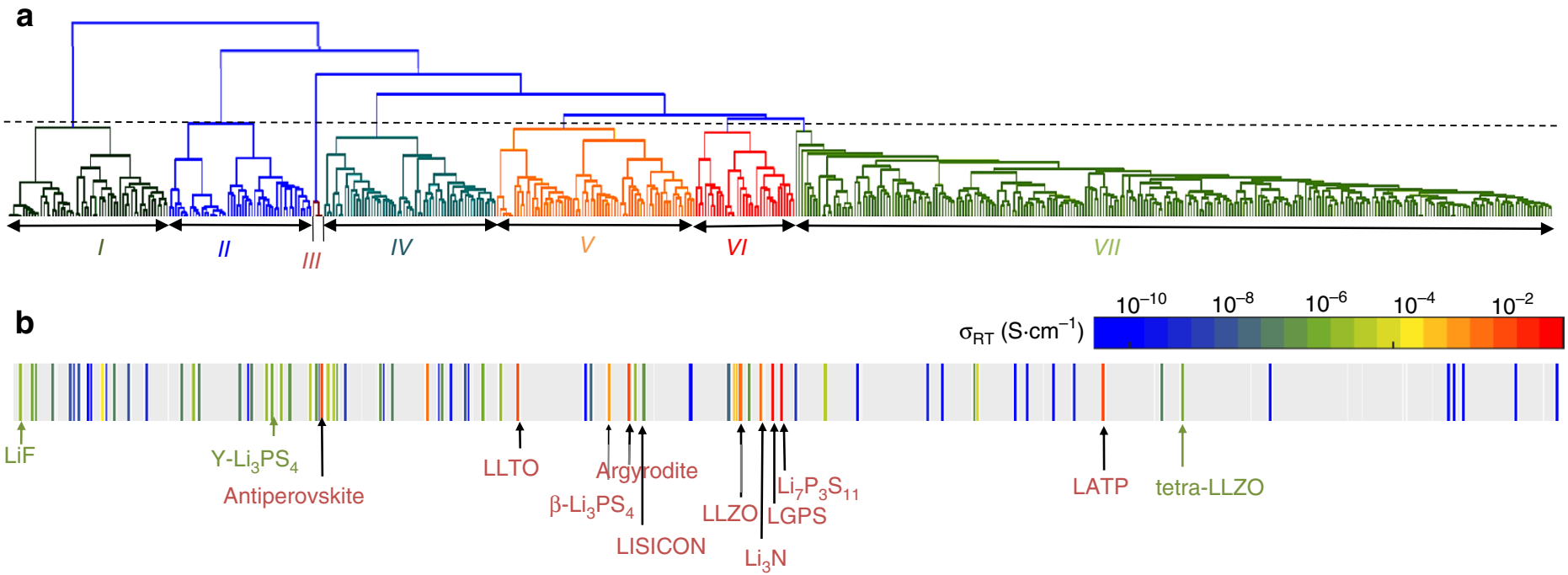


Figure: a. Dendrogram generated using the agglomerative hierarchical clustering method. The dashed line shows the position where all compounds are partitioned into seven groups, marked as I–VII from left to right and distinguished by different colors. b Mapping the dendrogram to the conductivity reveals the grouping of known solid-state Li-ion conductors in group V and VI. Reproduced from [1].

Density-based Clustering

- Clusters are defined as areas of higher density.
- Most popular variant is Density-based spatial clustering of applications with noise (DBSCAN).[2]
- DBSCAN groups together points that are closely packed together and marks points that lie alone in low-density regions as outliers.
- Key parameters of the DBSCAN algorithm are:
  - eps: Max distance between two samples for one to be considered as in the neighborhood of the other.
  - min_samples: No. of samples in a neighborhood for a point to be considered as a core point.

K-means and DBSCAN in scikit-learn

```
# Reading images using matplotlib
from matplotlib import image
import matplotlib.pyplot as plt
import numpy as np
# load image as numpy array.
data = image.imread('example.png')
# Display image
plt.imshow(data)

from sklearn.cluster import kmeans, DBSCAN
clustering = KMeans(k).fit(X)
print(clustering.labels_)
clustering = DBSCAN(eps=3, min_samples=2).fit(X)
print(clustering.labels_)
```

Bibliography


Ying Zhang, Xingfeng He, Zhiqian Chen, Qiang Bai, Adelaide M. Nolan, Charles A. Roberts, Debasish Banerjee, Tomoya Matsunaga, Yifei Mo, and Chen Ling.
Unsupervised discovery of solid-state lithium ion conductors.
Nature Communications, 10(1):5260, December 2019.


Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu.
A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.
page 6.

The End