# EECS 553 Lecture 18
# Machine Learning (ECE)
# [class version]

Jeff Fessler

University of Michigan
©Laura Balzano and Jeff Fessler, UM, November 2, 2026, 13:36

F26

# Logistics

© 2026 Fessler

- HW 6 due Wed. Nov. 4
- PA3: Fri. Nov. 6: draft reproducibility report
- HW 7 released soon; due Wed. Nov. 11
  Some non-graded problems for timing flexibility
  Last HW before Exam 2
- Exam 2: Mon. Nov. 16, 6-8PM; 1013 Dow; 2 sheets again
- iClicker does not count towards grade today (election day)

# Recap/Outline

© 2026 Fessler

- Recap
  - Hyperparameter selection / Bayesian optimization
  - Cross validation
- Today
  - Bayesian optimization: Addendum
  - Labeling is hard
  - Unsupervised learning
    - Representations
    - Embedding
    - Feature extraction
    - Association/clustering/segmentation
    - Density estimation
  - Principal component analysis (PCA)
  - Embedding and dimensionality reduction

- BO uses a Gaussian process model for $F(\quad)$ with **kernel** $k(\cdot, \cdot)$.
- Specifically, the model for $F(\cdot)$ is that *any* $m$ values of $F$ at distinct locations have a joint Gaussian distribution:

$$\tilde{\boldsymbol{\lambda}}_1, \ldots, \tilde{\boldsymbol{\lambda}}_m \in \mathbb{R}^q \implies \begin{bmatrix} F(\tilde{\boldsymbol{\lambda}}_1) \\ \vdots \\ F(\tilde{\boldsymbol{\lambda}}_m) \end{bmatrix} \sim \mathcal{N}(\quad \mathbf{1}_m, \boldsymbol{K}), \; K_{ij} = k(\tilde{\boldsymbol{\lambda}}_i, \tilde{\boldsymbol{\lambda}}_j).$$

- The observation model after $t$ trials is:

$$\boldsymbol{y} = \underbrace{\boldsymbol{F} + \boldsymbol{\varepsilon}}_{\text{independent}}, \quad \boldsymbol{F} = \begin{bmatrix} F(\boldsymbol{\lambda}_1) \\ \vdots \\ F(\boldsymbol{\lambda}_t) \end{bmatrix}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma_\varepsilon^2 \boldsymbol{I}_t).$$

- Typically BO uses a plug-in estimate of the mean:

$$\hat{\mu} = \frac{1}{t} \sum_{i=1}^{t} y_i.$$

- The lower confidence bound (LCB) approach uses posterior mean and standard deviation:

$$\mu(\boldsymbol{\lambda}) \triangleq \mathbb{E}\{F(\boldsymbol{\lambda}) \mid \boldsymbol{Y} = \boldsymbol{y}\}$$
$$\mathrm{LCB}(\boldsymbol{\lambda}) \triangleq \mu(\boldsymbol{\lambda}) - \underbrace{2}_{\text{or } c} \sigma(F(\boldsymbol{\lambda}) \mid \boldsymbol{Y} = \boldsymbol{y})$$

- If $\boldsymbol{X}$, $\boldsymbol{Y}$ are jointly distributed Gaussian random vectors, then the **conditional distribution** of $\boldsymbol{X}$ given $\boldsymbol{Y} = \boldsymbol{y}$ is:

$$\boldsymbol{X} \mid \boldsymbol{Y} = \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}_{X|Y}(\boldsymbol{y}), \boldsymbol{\Sigma}_{X|Y}\right)$$
$$\boldsymbol{\mu}_{X|Y}(\boldsymbol{y}) \triangleq \mathbb{E}\{\boldsymbol{X} \mid \boldsymbol{Y} = \boldsymbol{y}\} = \boldsymbol{\mu}_X + \boldsymbol{\Sigma}_{X,Y}\boldsymbol{\Sigma}_Y^{-1}(\boldsymbol{y} - \boldsymbol{\mu}_Y)$$
$$\boldsymbol{\Sigma}_{X|Y} \triangleq \mathrm{cov}(\boldsymbol{X} \mid \boldsymbol{Y} = \boldsymbol{y}) = \boldsymbol{\Sigma}_X - \boldsymbol{\Sigma}_{X,Y}\boldsymbol{\Sigma}_Y^{-1}\boldsymbol{\Sigma}_{Y,X}.$$

- For LCB, think of $X = F(\boldsymbol{\lambda})$ for any candidate value of $\boldsymbol{\lambda} \in \mathbb{R}^q$.

The **probability of improvement** approach uses:

$$\arg\max_{\boldsymbol{\lambda}} \mathrm{PI}(\boldsymbol{\lambda}), \quad \mathrm{PI}(\boldsymbol{\lambda}) \triangleq \mathrm{P}\{F(\boldsymbol{\lambda}) < y_{\min} - \xi \mid \}$$

$$y_{\min} \triangleq \min_{i \in \{1,\ldots,t\}} y_i,$$

where $\xi \geq 0$ is an optional exploration parameter.

The value $y_{\min}$ is noisy in general.
When the measurement noise is high, a more robust approach is to use the minimum of the *posterior mean* at the already sampled points:

$$f_{\min} \triangleq \min_i \mu(\boldsymbol{\lambda}_i).$$

This approach uses the Gaussian Process to "filter out" measurement noise, providing a cleaner estimate of the best true underlying function value found so far. (Not needed in HW.)

# Bayesian optimization choices

© 2026 Fessler

- noise variance $\sigma_\varepsilon^2$
- kernel for covariance model
- kernel parameters (scale, width for Gaussian): can be optimized
- acquisition function (LCB, PI, ...) and its parameters
- number of updates
- parameter ranges
- how to optimize the acquisition function

- Gaussian kernel hyperparameters: std dev $\sigma_f$, length scale $\ell$:

$$k(x,z) = \sigma_f \exp\left(-\frac{(x-z)^2}{2\ell^2}\right).$$

- Diagonal values of covariance $\boldsymbol{K} = \boldsymbol{K}(\sigma_f, \ell)$ are $\sigma_f^2$.
- Adapt: after each update, select $(\sigma_f, \ell)$ (e.g., via grid search!) to maximize the GP log marginal likelihood:

$$\log \mathrm{p}(\boldsymbol{y}; \sigma_f, \ell) = -\frac{1}{2}\log(2\pi \det \boldsymbol{K}) - \frac{1}{2}\boldsymbol{r}^\top \boldsymbol{K}^{-1}\boldsymbol{r}, \quad \boldsymbol{r} = \boldsymbol{y} - \hat{\mu}\mathbf{1}.$$

- The implementation uses a **Cholesky decomposition** $\boldsymbol{K} = \boldsymbol{L}\boldsymbol{L}^\top$:

$$\boldsymbol{r}^\top \boldsymbol{K}^{-1}\boldsymbol{r} = \boldsymbol{r}^\top (\boldsymbol{L}\boldsymbol{L}^\top)^{-1}\boldsymbol{r} = \left\|\boldsymbol{L}^{-1}\boldsymbol{r}\right\|_2^2, \quad \det(\boldsymbol{K}) = \det(\boldsymbol{L}\boldsymbol{L}^\top) = \det(\quad)^2$$

- Demo shows better results when adapting kernel hyperparameters.

# CQ: Unsupervised learning: Warm up

© 2026 Fessler

No right answer:

Q18.1 How familiar are you with unsupervised learning?

A: I've never heard of it.

B: I've heard of it, but I don't know much about it.

C: I know a few things about it.

D: I have worked on it in my research.

Preview:

- Feature matrix
- Clustering:
- Dimensionality reduction:
-

# Supervised learning: Recap

© 2026 Fessler

- Training data points $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$, $i = 1, \ldots, n$.
- Training label $\boldsymbol{y}_i \in \mathcal{Y}$ for each point above.
- A function (classifier or regressor) $f : \mathcal{X} \mapsto \mathcal{Y}$ predicts $\boldsymbol{y}$ from $\boldsymbol{x}$.

This is a useful paradigm only when we have labeled data.
What do we do when labels

# Labeling is hard - a case study

© 2026 Fessler

- **ImageNet** is a database of 14M labeled images of $K > 20,000$ categories of nouns (classes) envisioned by Fei-Fei Li in 2006.
- "Li's first idea was to hire undergraduate students for $10 an hour to manually find images and add them to the dataset. But back-of-the-napkin math quickly made Li realize that at the undergrads' rate of collecting images it would take 90 years to complete." [3]
- They discovered **Amazon Mechanical Turk** (launched 2005) and knew it was the only way to build a dataset the size they wanted [4].

- Not everyone who wants to create a dataset can afford Mechanical Turk.
- Not every problem we want to solve can be labeled by crowd-sourcing.
  Example:

# Labeling is hard - and there are many solutions © 2026 Fessler

- **Self-supervised learning**: let some parts of the data be "labels" for other parts of the data, *e.g.*, predict masked pixels or words, or using captions of images online, or use **data augmentation** to predict known simulated changes.
- **Weak supervision** aka **semi-supervised learning**: learn from a mixture of
- **Active learning**: collect only the "most important" labels that
- **Unsupervised learning**: learn only from

These are not mutually exclusive; self-supervised learning is a type of unsupervised learning.

# Unsupervised learning

© 2026 Fessler

What can we learn from *unlabeled* data?

The goal of unsupervised learning is to within unlabeled datasets. This is a vague goal, but gets defined more precisely in each setting.

Duda & Hart [2, p. 517] enumerate 5 reasons for unsupervised learning:

- labeling is expensive, *e.g.*, speech
- labels may be unknown (*e.g.*, find common patterns, label later)
- patterns may change over time (*e.g.*, food as seasons change)
- find features that are useful for classification
- feature discovery may alter classifier design approach

# Unsupervised learning: Canonical problems

© 2026 Fessler

- 
- 
- 

# Representations

© 2026 Fessler

- All of the ML frameworks discussed so far start with $\boldsymbol{x} \in \mathbb{R}^d$.
- How do we handle other data types?
  - strings and language
  - categorical data
  - relational data, *e.g.*, graph
  - meta data, web site visit or transaction records
- Even for numerical data, the raw data may not always be the best feature vector for learning:
  - image pixels
  - uncalibrated sensor values
  - audio samples

# Embedding example: Images

© 2026 Fessler



Figure courtesy [5].
Clothing images embedding in $d = 2$ dimensions.

# Embedding example: Feature extraction

© 2026 Fessler

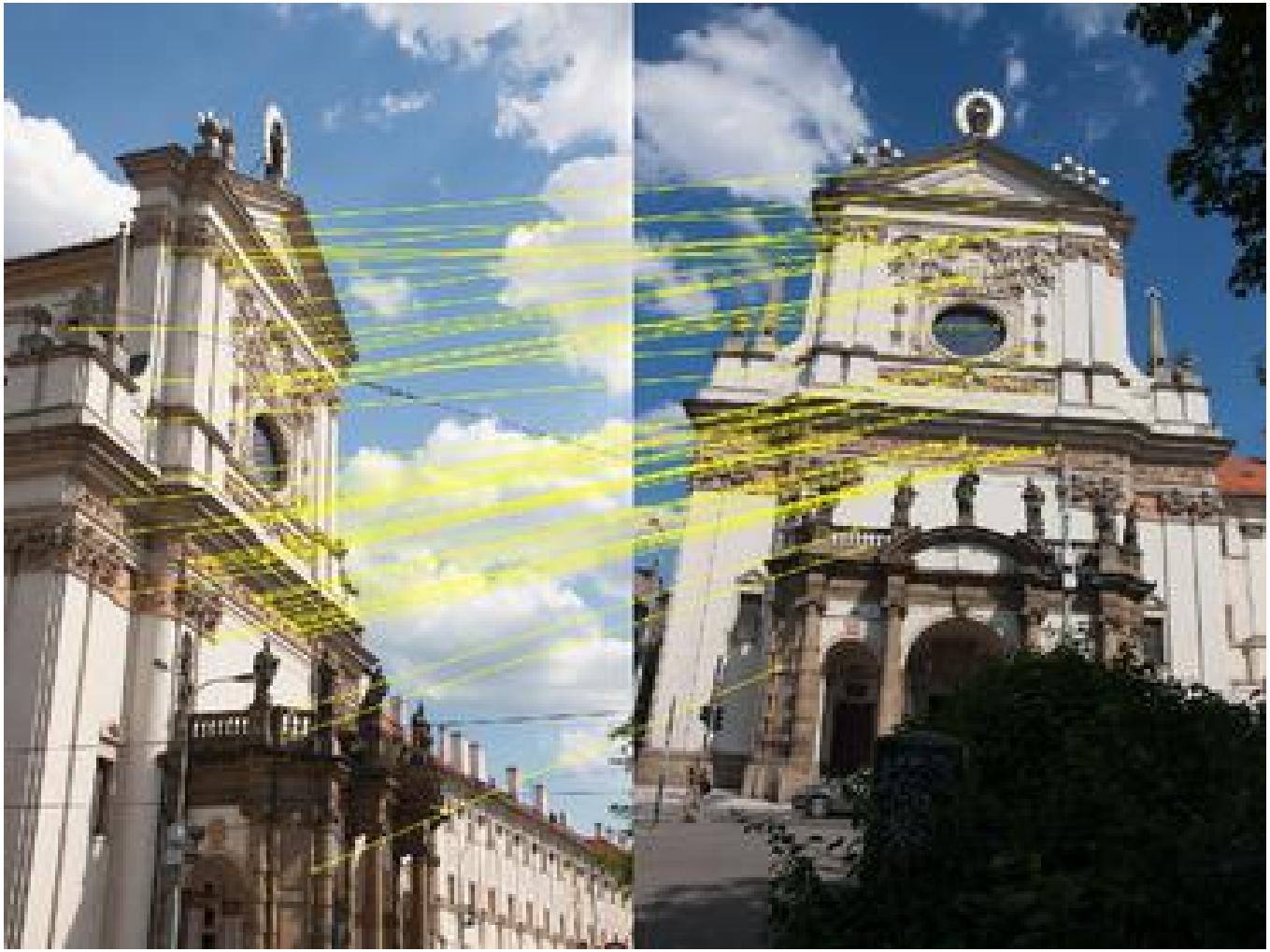


Figure courtesy [6].

# Clustering example: Image segmentation

© 2026 Fessler

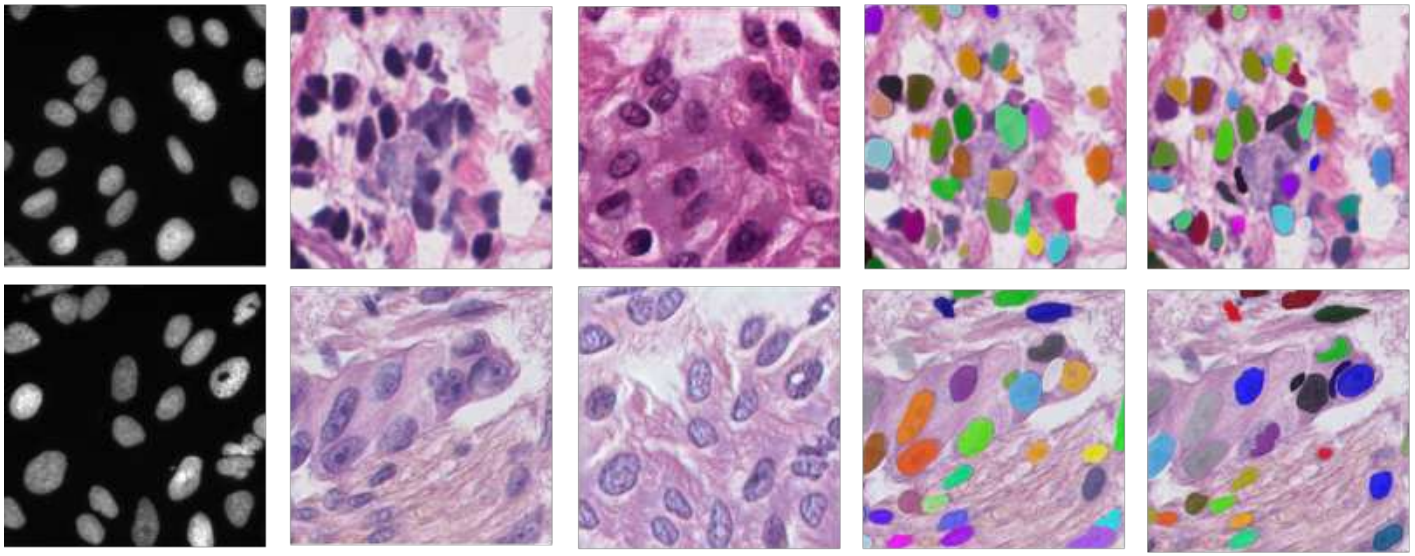
Use k-means (next class!) on the pixels of an image:




Figure courtesy of [7].

# Clustering example: Image segmentation

© 2026 Fessler



(a) (b) (c) (d) (e)

Figure 1. Example images of our proposed framework. (a) fluorescence microscopy images; (b) real histopathology images; (c) our synthesized histopathology images; (d) nuclei segmentation generated by our proposed UDA method; (e) ground truth.

Figure courtesy of [8].

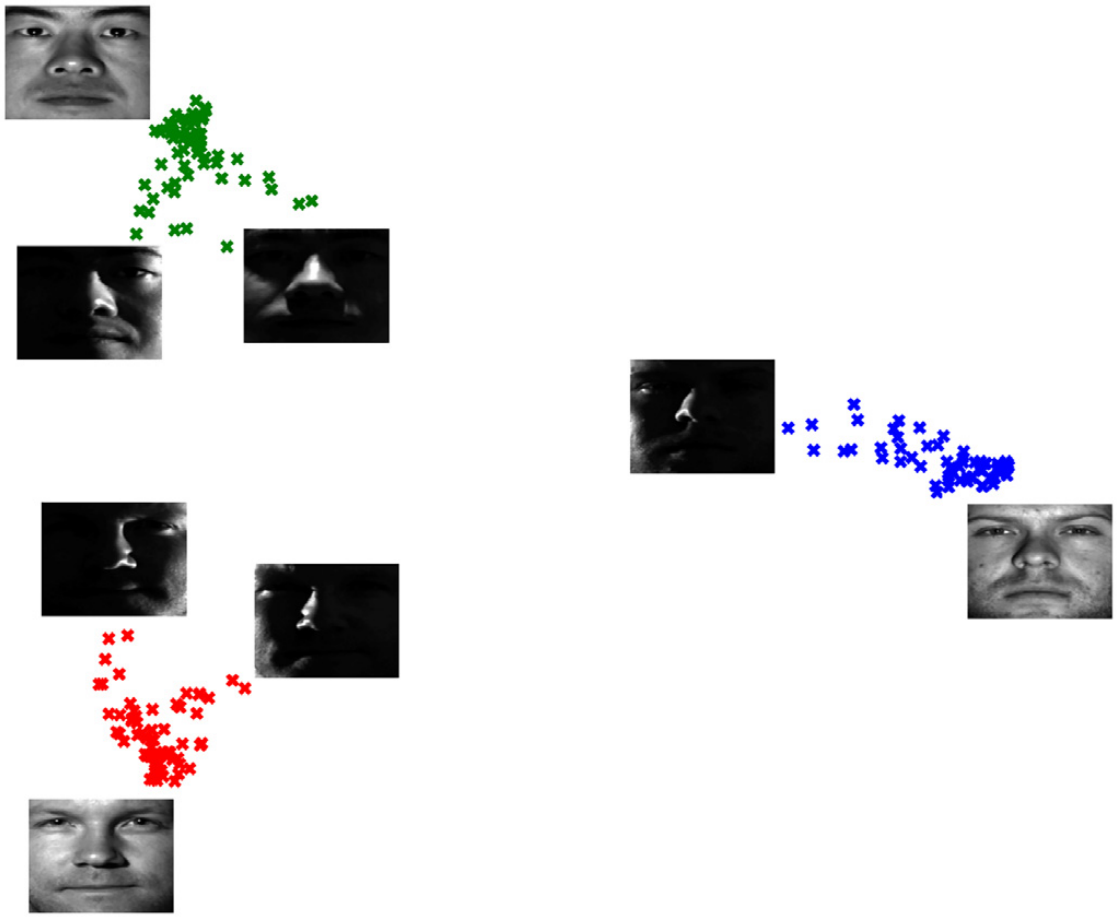


**Fig. 2.** Two-dimensional embedding of points in Extended Yale Face Database B, subjects 13, 26, and 38 using multidimensional scaling on the proposed point-to-point distance. The proposed distance provides an indication of which points lie near the estimated subspace and groups outliers with similar forms of shadow.

Figure courtesy of [9].

Estimate $p(\boldsymbol{x})$ from samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$.

Axis labels:

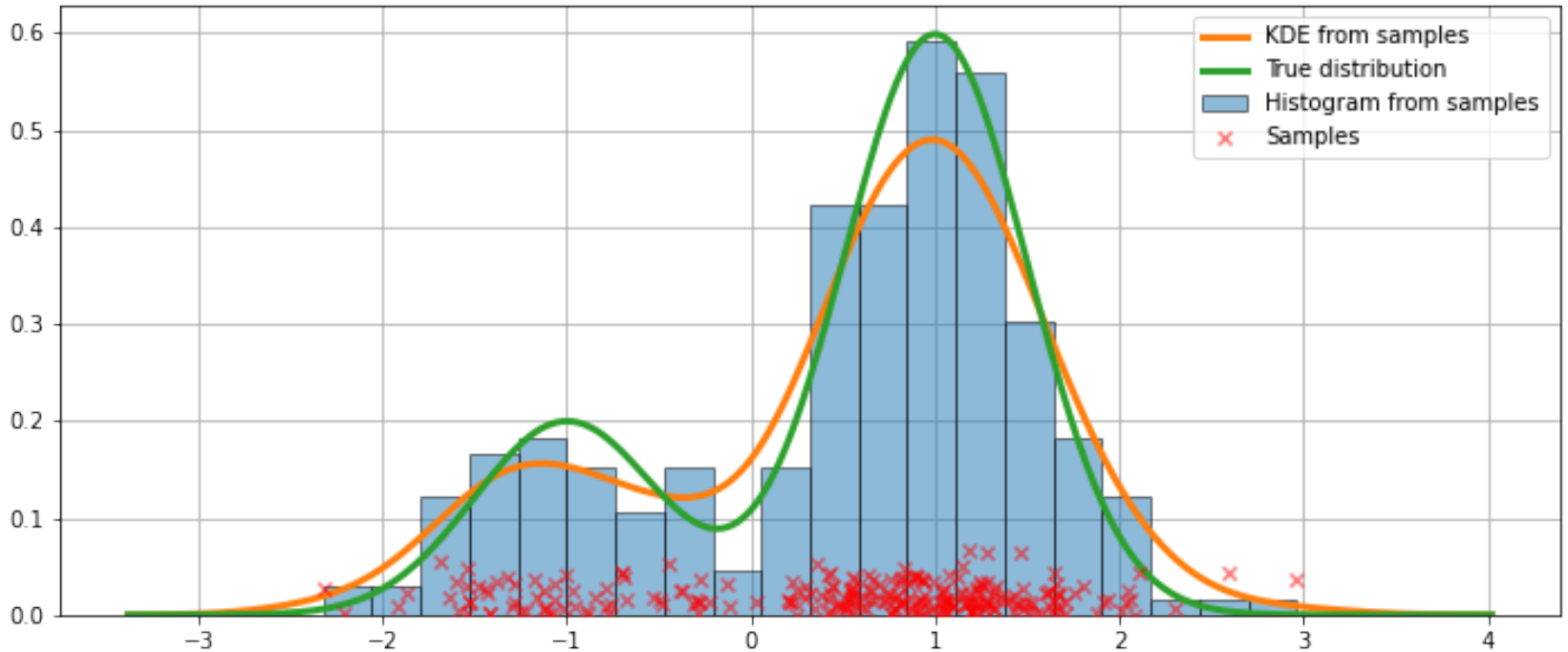
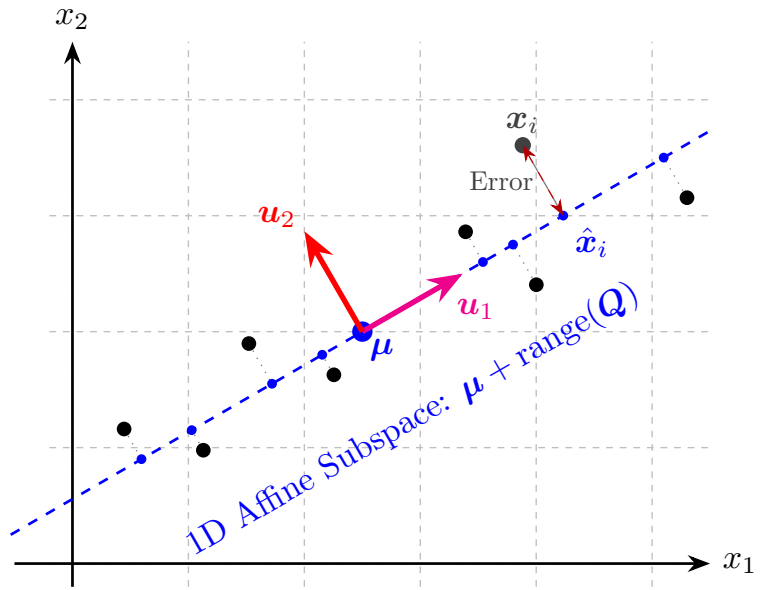



Figure courtesy [10].

The idea behind the **unsupervised learning** method called **principal component analysis** (**PCA**), is to approximate




where

- $\boldsymbol{\mu} \in \mathbb{R}^d$
- $\boldsymbol{Q} \in \{\boldsymbol{Q} \in \mathbb{R}^{d \times k} : \boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I}_{k \times k}\}, \; 1 \le k \ll d$
- $\boldsymbol{\theta}_i \in \mathbb{R}^k, \; i = 1, \ldots, n$
- We must learn all the PCA model parameters $\boldsymbol{\mu}$, $\boldsymbol{Q}$, $\{\boldsymbol{\theta}_i\}$ from the (unlabeled) training data $\{\boldsymbol{x}_i\}$.

Example of applying PCA with

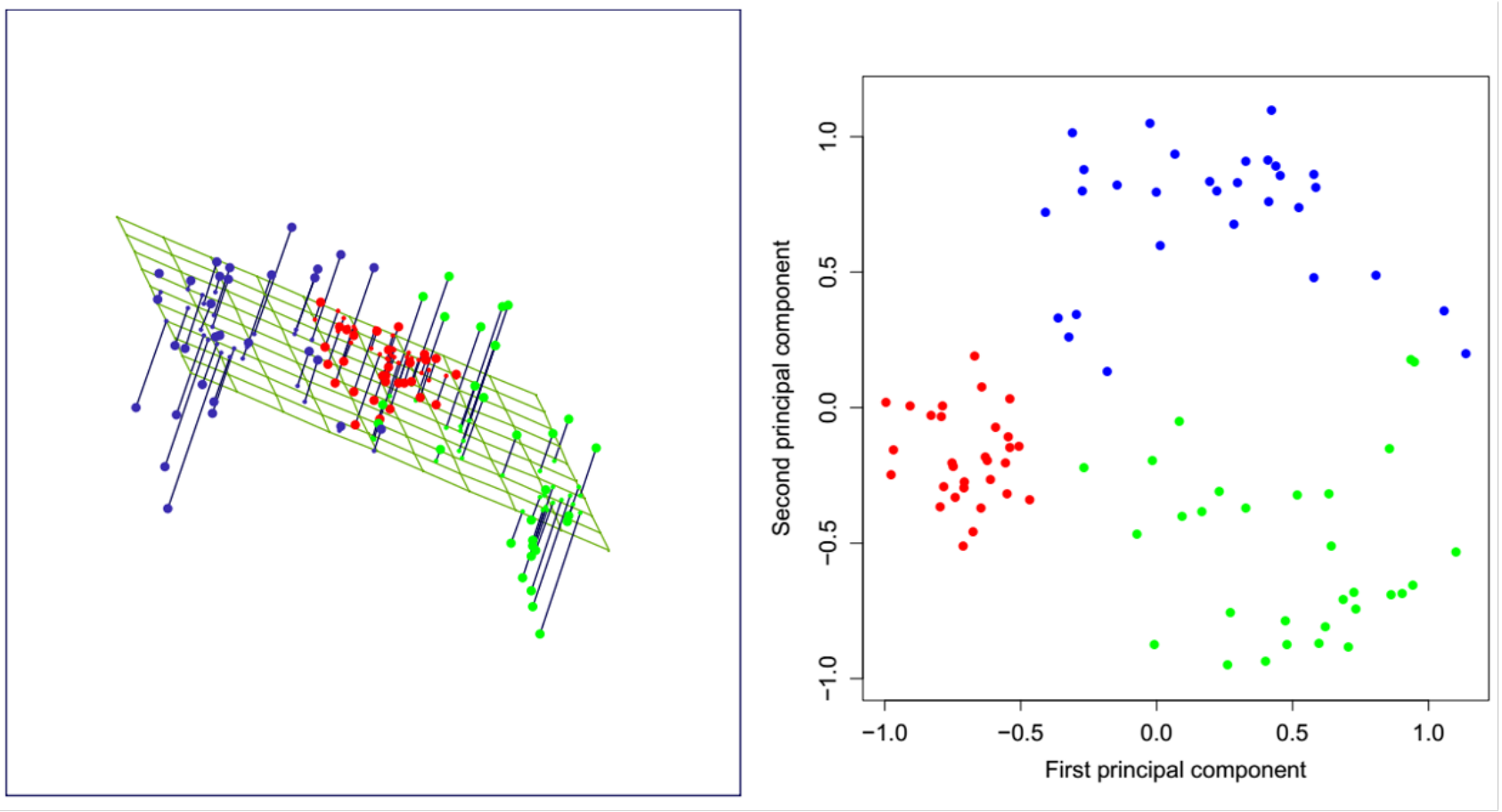



Figure courtesy [11].

# PCA optimization problem

© 2026 Fessler

- Mathematically, we define $\boldsymbol{\mu}, \boldsymbol{Q}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n$ to be the solution of

$$\qquad$$

- PCA gives the least squares approximation of the data by an **affine** subspace of dimension $k$. Often called "rank $k$" imprecisely. Matrices have rank; subspaces have dimension.
- We must solve for $n+1$ vectors and 1 matrix jointly.
- The solution for $\boldsymbol{\theta}_i$ (in terms of $\boldsymbol{Q}$ and $\boldsymbol{\mu}$) is simply the least-squares solution:

$$\boldsymbol{\theta}_i = \qquad .$$

Substitute this in to solve for $\boldsymbol{\mu}$ and $\boldsymbol{Q}$.

$$\underset{\boldsymbol{\mu}, \boldsymbol{Q} : \boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I}}{\arg\min} \sum_{i=1}^{n} \left\| \boldsymbol{\mu} + \boldsymbol{Q}\boldsymbol{Q}^\top (\boldsymbol{x}_i - \boldsymbol{\mu}) - \boldsymbol{x}_i \right\|_2^2$$

- With this new form, the solution for $\boldsymbol{\mu}$ (in terms of $\boldsymbol{Q}$) satisfies

  Therefore for any $\boldsymbol{Q}$ we can take $\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \triangleq \bar{\boldsymbol{x}}$.
- The solution for $\boldsymbol{Q}$ is given in terms of the spectral (or eigenvalue) decomposition of the **sample covariance** matrix

# CQ: PCA sample covariance

© 2026 Fessler

Q18.2 What do we know about the matrix $\boldsymbol{S} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top$?

A: It is a diagonal matrix.

B: It is positive semi-definite.

C: Its eigenvalues are strictly positive.

D: All of the above

E: B and C

- Letting $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \dots & \boldsymbol{x}_n \end{bmatrix}$, we can write $\sum_{i=1}^{n} \left\| \boldsymbol{\mu} + \boldsymbol{Q}\boldsymbol{Q}^\top (\boldsymbol{x}_i - \boldsymbol{\mu}) - \boldsymbol{x}_i \right\|_2^2 = \left\| \boldsymbol{\mu}\mathbf{1}^\top + \boldsymbol{Q}\boldsymbol{Q}^\top (\boldsymbol{X} - \boldsymbol{\mu}\mathbf{1}^\top) - \boldsymbol{X} \right\|_{\mathsf{F}}^2$ and then use that $\|\boldsymbol{B}\|_{\mathsf{F}}^2 = \mathsf{Tr}(\boldsymbol{B}^\top \boldsymbol{B})$ along with trace manipulations to see that this is equivalent to:

- Note that $\boldsymbol{S}$ is PSD, so its eigenvalues are nonnegative.
- This problem therefore has a solution that uses the eigenvalue decomposition:

where $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ is an eigenvalue decomposition of $\boldsymbol{S}$

- In summary, a solution to the PCA optimization problem is

1. $\boldsymbol{\mu} = \bar{\boldsymbol{x}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i$
2. 
3. $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{u}_1 & \dots & \boldsymbol{u}_k \end{bmatrix}$ $k$ principal eigenvectors
4. $\boldsymbol{\theta}_i = \boldsymbol{Q}^\top (\boldsymbol{x}_i - \bar{\boldsymbol{x}}), \; i = 1, \dots, n$

where $\boldsymbol{S} = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^\top$ is an **eigendecomposition** of $\boldsymbol{S}$ with (real!) eigenvalues in descending order.

Q18.3 The above solution is unique when $1 < k \ll d$ and the vectors $\{\boldsymbol{x}_i\}$ are linearly independent ?

A: True B: False **??**

- **Principal components** or **scores**:

- Principal eigenvectors/directions

- Reconstruction (or approximation or representation) of $\boldsymbol{x}_i$:

© 2026 Fessler

Q18.4 The principal components and principal eigenvectors of $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top$ are the same as those of

$$\boldsymbol{S} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^\top.$$

A: Yes B: No C: Insufficient information

- Suppose $\bar{\boldsymbol{x}} = \mathbf{0}$ for simplicity.
- The sample variance of the components $\{\theta_i^{(1)} = \boldsymbol{u}_1^\top \boldsymbol{x}_i\}$ is:

$$\operatorname{Var}(\theta^{(1)}) = \frac{1}{n} \sum_{i=1}^{n} \left(\theta_i^{(1)}\right)^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\boldsymbol{u}_1^\top \boldsymbol{x}_i\right)^2$$

$$= \quad$$

- The solution of the "maximize variance" problem

$$\underset{\boldsymbol{u}_1 \,:\, \|\boldsymbol{u}_1\|_2 = 1}{\arg\max} \quad \boldsymbol{u}_1^\top \boldsymbol{S} \boldsymbol{u}_1$$

is [12, p. 80]

- Can remove $\boldsymbol{u}_1$ component and continue to find $\boldsymbol{u}_2$ etc. [12, p. 272].

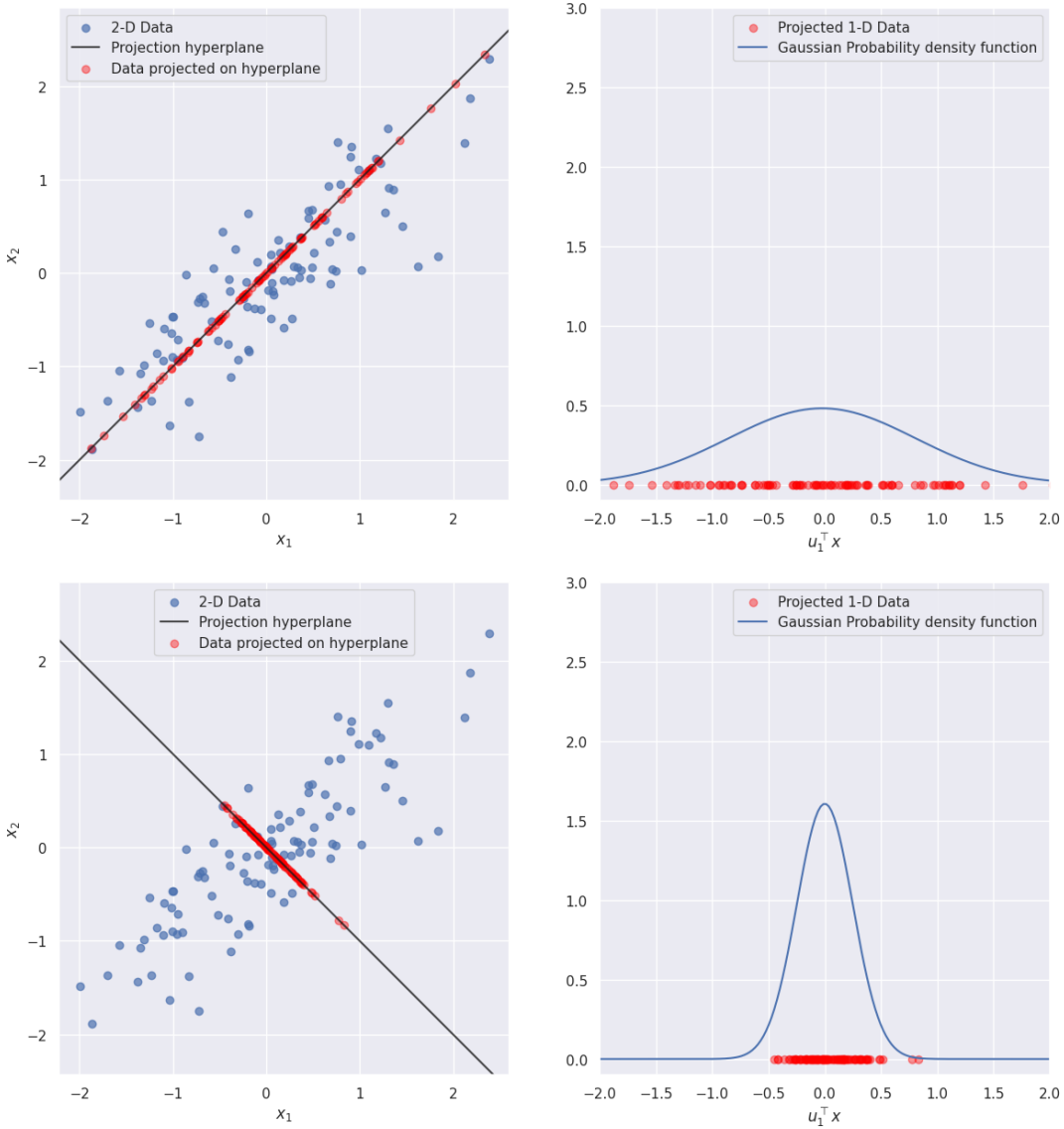
- Suppose $\bar{\boldsymbol{x}} = \boldsymbol{0}$.
- By the **Eckart-Young-Mirsky theorem**, the best rank-at-most-$k$ approximation to the matrix $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 & \dots & \boldsymbol{x}_n \end{bmatrix}$ is $\widehat{\boldsymbol{X}} = \boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{X}$, when "best" is measured by $\|\boldsymbol{X} - \widehat{\boldsymbol{X}}\|$ for any **unitarily invariant** matrix norm [12, p. 239].
- In terms of the columns of $\boldsymbol{X}$, this best approximation leads to orthogonal projections:

$$\hat{\boldsymbol{x}}_i = \underbrace{\boldsymbol{Q}\boldsymbol{Q}^\top}\boldsymbol{x}_i = \boldsymbol{Q}\boldsymbol{\theta}_i = \sum_{j=1}^{k} \theta_i^{(j)} \boldsymbol{u}_j.$$

- Intuition:
  - Columns of $\boldsymbol{Q}$ define a $k$-dimensional coordinate system for range($\boldsymbol{Q}$).
  - $\boldsymbol{\theta}_i = \boldsymbol{Q}^\top \boldsymbol{x}_i$ are the coordinates of $\widehat{\boldsymbol{x}}_i$ in the subspace

Interactive colab demo

Choosing $\boldsymbol{u}_1$ to be "best aligned with the data" maximizes the variance of the first principle component.



# CQ: Hyperparameters

© 2026 Fessler

Q18.5 Which of the following are hyperparameters?

A: rank $k$ in a PCA embedding

B: $\lambda$ in ridge regression

C: $\boldsymbol{\mu}$ in PCA

D: all of the above

E: A and B

# Embedding and dimensionality reduction: Goal © 2026 Fessler

The goal of **embedding** aka **dimensionality reduction** methods is to map a (potentially) high-dimensional dataset into a low-dimensional space in a way that preserves

The term as used in ML may be somewhat counter-intuitive compared to the English meaning of embed.

# Embedding vs dimensionality reduction

© 2026 Fessler

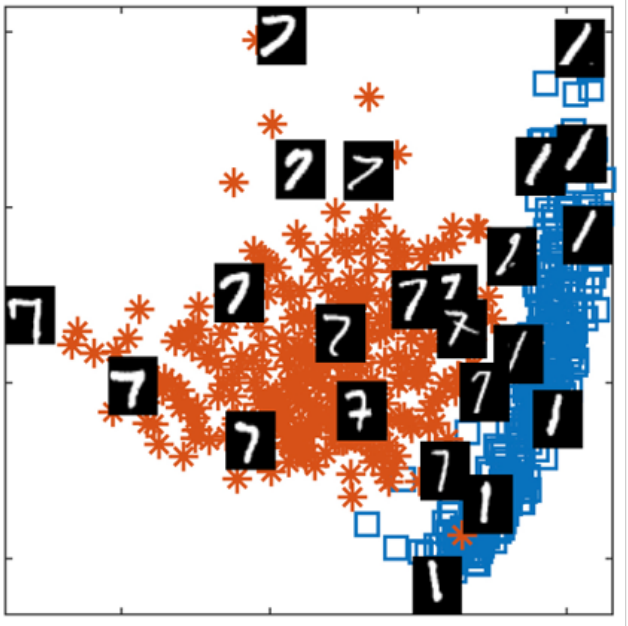
- **Embedding** in ML generally means transforming data into a lower-dimensional Euclidean-space representation.
  - The initial data may be real-valued, but often is not.
  - The goal is typically to find a useful task-oriented representation.
- **Dimensionality reduction** also means to transform to a lower-dimensional representation.
  - It typically assumes the initial data are at least numerical and sometimes real-valued.
  - It can provide data compression and can reduce computation.

# Embedding and dim. reduction: Examples

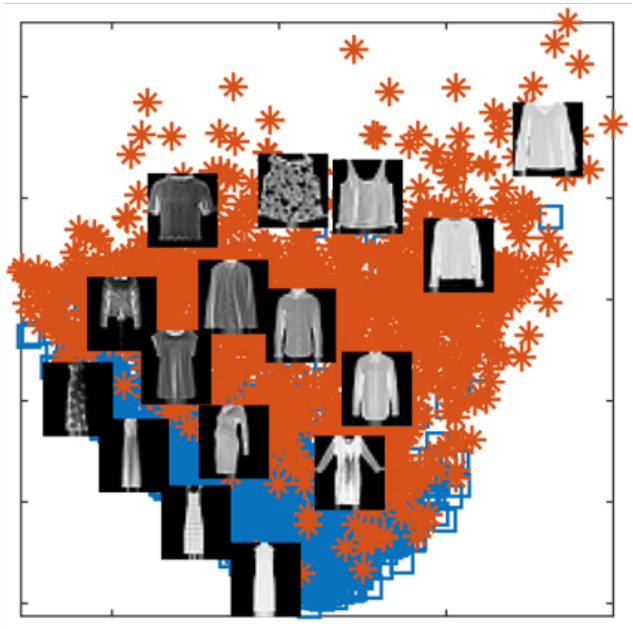
© 2026 Fessler

- Embed words into $\mathbb{R}^d$ for $d \approx 100 - 1000$ in natural language processing (*e.g.*, **Word2vec**).
- Embed commercial products (categorical data) in $\mathbb{R}^d$ for $d \approx 20 - 100$ in recommender systems.
- Embed songs into $\mathbb{R}^d$ for $d \approx 20 - 50$ for playlist prediction [13]. Songs have different lengths so raw audio waveforms are unsuitable for finding similarities.
- Embed images in $\mathbb{R}^d$ where $d \ll$ number of pixels. Could also be called dimension reduction, especially if new features are learned by a generic process.
- Reduce the dimension of data by subsampling it.
- Reduce the dimension of data for visualization purposes.
- Reduce the dimension of a large state-space representation of a dynamical system to facilitate computation.

# Applications of embedding: image classification © 2026 Fessler



First 2 PCs for MNIST 1s and 7s



First 2 PCs for FMNIST shirts and dresses

Figure courtesy [14].

Every digit classification example all semester long has used PCA to embed MNIST images into 2D ($k = 2$) to facilitate visualization.
Right: Fashion-MNIST: [15]

# Applications of embedding: image denoising

© 2026 Fessler

- Image denoising is a key component of diffusion models for image generation.
- Here $16 \times 16 \times 3$ image patches ($d = 768$) are "bottleneck" embedded in $\approx$ 64-dimensional space
- Impressive image generation results

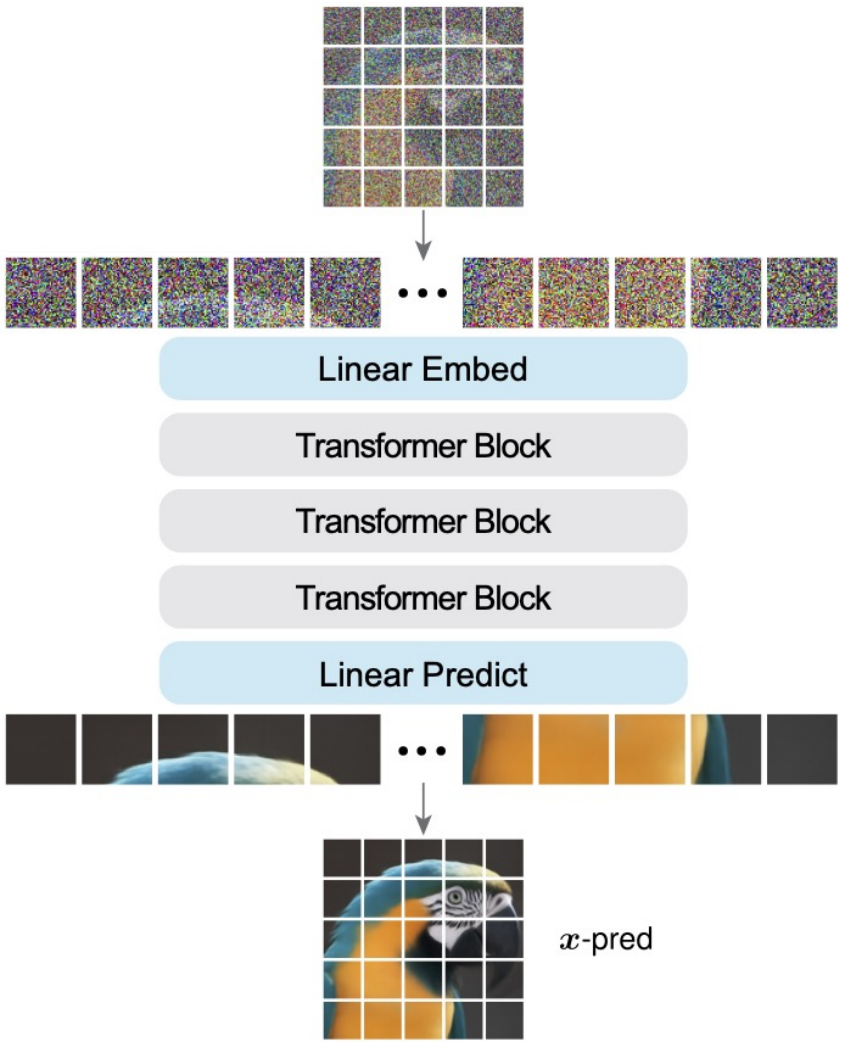



Figure from [16]

# Applications: genetics

© 2026 Fessler

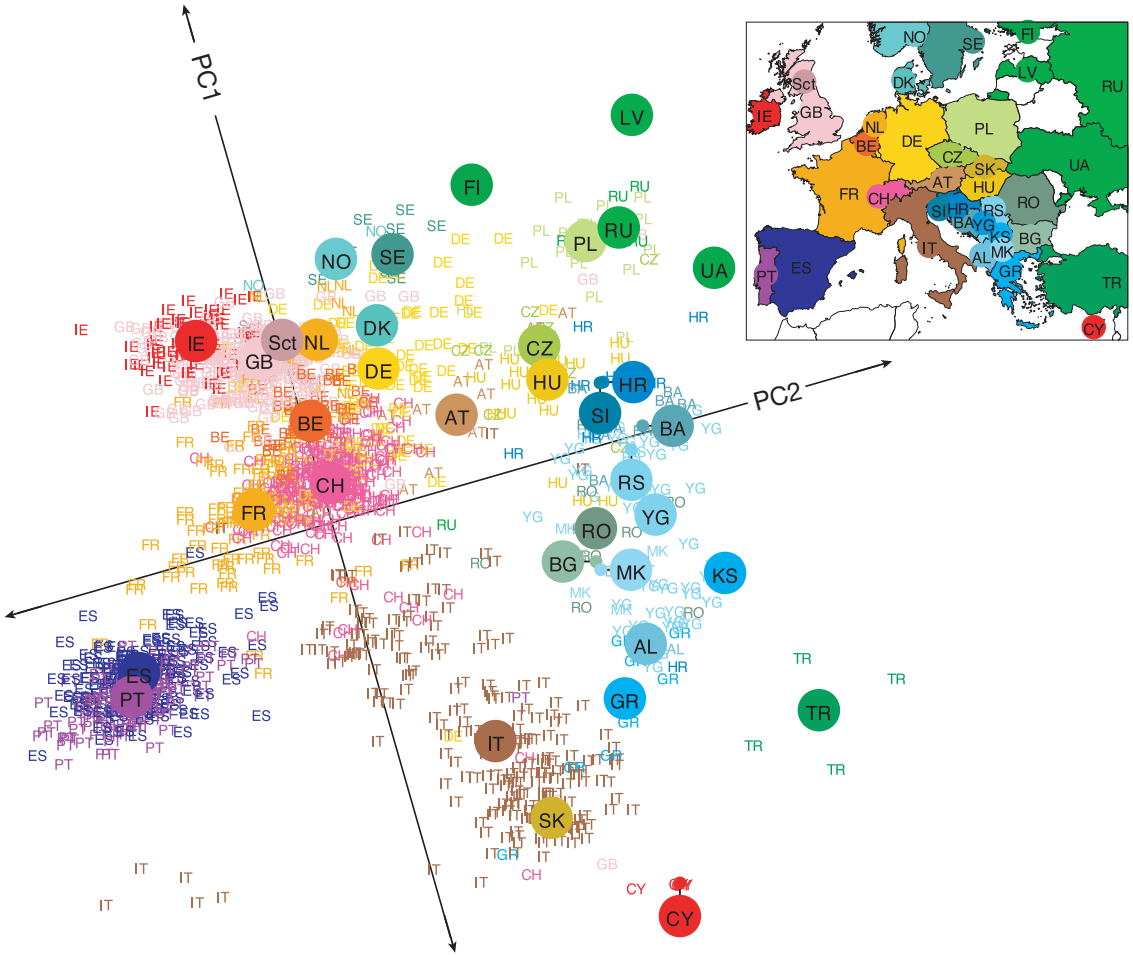



Figure courtesy [17].

PCA with on human genotype data from $d = 197,146$ loci/SNPs in $n = 1,387$ European individuals.

# Applications: cell biology

© 2026 Fessler

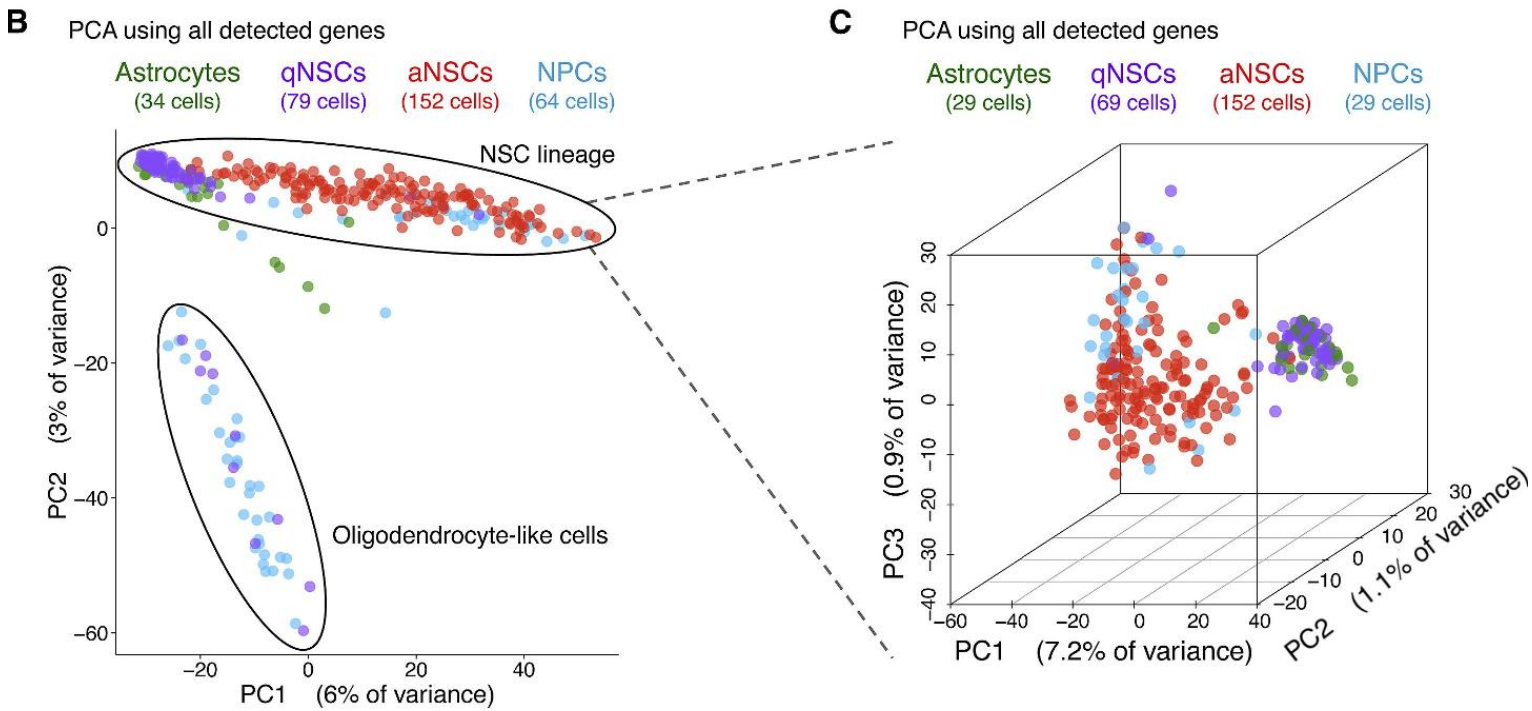



Figure courtesy [18]. DOI

Exploratory analysis via PCA with $k = 2$ and $k = 3$ of single-cell gene data:

- Quiescent neural stem cells (qNSC)
- Activated neural stem cells (aNSC)
- Neural progenitor cells (NPC)

Gemini said $d = 14,551$ genes, but when pressed it admitted that it had hallucinated and claimed that a typical number is 12,000 to 18,000 genes.

# References I


[1] K. P. Murphy.
*Probabilistic machine learning: An introduction*.
MIT Press, 2022.

[2] R. O. Duda, P. E. Hart, and D. G. Stork.
*Pattern classification*.
Wiley, New York, 2001.

[3] Dave Gershgorn.
The data that transformed ai research—and possibly the world.
*Quartz*, 2017.
https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world.

[4] J. Deng, W. Dong, R. Socher, L-J. Li, K. Li, and F-F. Li.
ImageNet: A large-scale hierarchical image database.
In *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 248–55, 2009.

[5] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie.
Learning visual clothing style with heterogeneous dyadic co-occurrences.
In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4642–4650, 2015.

[6] paperswithcode.
Image matching, 2012.
https://paperswithcode.com/task/image-matching.

[7] Aditri Srivastava.
Image segmentation using k-means, 2021.
https://medium.com/analytics-vidhya/image-segmentation-using-k-means-4bd79e801785.

# References II


[8] Dongnan Liu, Donghao Zhang, Yang Song, Fan Zhang, Lauren O'Donnell, Heng Huang, Mei Chen, and Weidong Cai.
Unsupervised instance segmentation in microscopy images via panoptic domain adaptation and task re-weighting.
In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4243–4252, 2020.

[9] John Lipor and Laura Balzano.
Clustering quality metrics for subspace clustering.
*Pattern Recognition*, 104:107328, 2020.

[10] Josef Perktold, Skipper Seabold, and Jonathan Taylor.
Kernel density estimation, 2009.
https://www.statsmodels.org/v0.12.2/examples/notebooks/generated/kernel_density.html.

[11] Trevor Hastie, Robert Tibshirani, and Jerome H Friedman.
*The elements of statistical learning: data mining, inference, and prediction*, volume 2.
Springer, 2009.

[12] J. A. Fessler and R. R. Nadakuditi.
*Linear algebra for data science, machine learning, and signal processing*.
Cambridge, 2024.

[13] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims.
Playlist prediction via metric embedding.
In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 714–722, 2012.

[14] Alexander Ritchie, Laura Balzano, Daniel Kessler, Chandra S Sripada, and Clayton Scott.
Supervised pca: A multiobjective approach.
*arXiv preprint arXiv:2011.05309*, 2020.

# References III


[15] H. Xiao, K. Rasul, and R. Vollgraf.
Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, 2017.

[16] T. Li and K. He.
Back to basics: Let denoising generative models denoise.
In *Proc. IEEE Conf. on Comp. Vision and Pattern Recognition*, 2026.

[17] John Novembre, Toby Johnson, Katarzyna Bryc, Zoltán Kutalik, Adam R Boyko, Adam Auton, Amit Indap, Karen S King, Sven Bergmann, Matthew R Nelson, et al.
Genes mirror geography within europe.
*Nature*, 456(7218):98–101, 2008.

[18] Ben W Dulken, Dena S Leeman, Stéphane C Boutet, Katja Hebestreit, and Anne Brunet.
Single-cell transcriptomic analysis defines heterogeneity and transcriptional dynamics in the adult neural stem cell lineage.
*Cell reports*, 18(3):777–790, 2017.