
Interpreting PCA and Formalizing Prediction

ECE 57000 — September 21, 2026

David I. Inouye

| Friday connected PCA's equations to working NumPy code

- PCA's leading directions minimize reconstruction error and maximize retained variance; usefulness depends on the data and purpose.
- Array shapes, reduction axes, and broadcasting determine which computation the code performs.
- Center the observations, compute the reduced SVD, then encode with $X @ W$ and reconstruct with $Z @ W.T + \mu$.
- Reuse the fitted mean and directions for new observations; changing matrix-product grouping can greatly change memory and computation costs.

Today: can mathematically equivalent formulas give different numerical answers? Then return to real gene-expression data and ask what PCA reveals.

| New observations use **the fitted mean and directions**

```
1 # Apply the representation fitted on Xraw
2 Xraw_new = np.array([[5., 16.], [7., 20.]])
3 Z_new = (Xraw_new - mu) @ W
4 Xraw_new_hat = Z_new @ W.T + mu
5 print("Z_new.round(3):\n", Z_new.round(3))
6 print("Xraw_new_hat.round(3):\n",
7       Xraw_new_hat.round(3))
```

```
Z_new.round(3):
[[0.   ]
 [4.472]]
Xraw_new_hat.round(3):
[[ 5.    16.   ]
 [ 7.051 19.974]]
```

- **Fitting** estimates μ and W from a chosen dataset.
- **Applying** uses those fixed values for other observations with the same feature definitions.
- Re-estimating the mean or directions creates a different fitted representation.

PCA relies on centering and **then** measuring variance. **What would happen if we computed variance directly, without centering first?**

| Computing spread for PCA starts with **an exact identity**

For one feature's values $x_1, \dots, x_n$, let $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$. The mean $\bar{x}$ is constant with respect to i .

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 &= \frac{1}{n} \sum_{i=1}^n (x_i^2 - 2x_i\bar{x} + \bar{x}^2) && \text{(Expand the square inside the sum)} \\ &= \frac{1}{n} \sum_{i=1}^n x_i^2 - \frac{2\bar{x}}{n} \sum_{i=1}^n x_i + \frac{1}{n} \sum_{i=1}^n \bar{x}^2 && \text{(Distribute the sum; pull out constants)} \\ &= \frac{1}{n} \sum_{i=1}^n x_i^2 - \frac{2\bar{x}}{n} (n\bar{x}) + \frac{1}{n} (n\bar{x}^2) && (\sum_{i=1}^n x_i = n\bar{x}; n \text{ copies of } \bar{x}^2) \\ &= \frac{1}{n} \sum_{i=1}^n x_i^2 - 2\bar{x}^2 + \bar{x}^2 && \text{(Cancel the factors of } n) \\ &= \frac{1}{n} \sum_{i=1}^n x_i^2 - \bar{x}^2 && \text{(Combine terms)} \end{aligned}$$

| Will mathematically equivalent computations give the same answer?

Take the five numbers $10^9 + 1, \dots, 10^9 + 5$. Their mean is $10^9 + 3$. **What variances will the two computations report?** Predict before revealing.

```
1 v = 1e9 + np.arange(1., 6.)
2 centered = v - v.mean()
3 center_first = np.mean(centered**2)
4 subtract_totals = np.mean(v**2) - v.mean()**2
5 print("Centered:", center_first)
6 print("Raw squares:", subtract_totals)
```

Centered: 2.0
Raw squares: 0.0

Choice	Centered	Raw squares
A	2	2 (up to a tiny rounding error)
B	2	0
C	2	Overflow: the squared values are too large

| Numerical errors can erase the entire answer

```
1 print("Centered values:", centered)
2 print("Mean of squares:", np.mean(v**2))
3 print("Square of mean:", v.mean()**2)
4 print("Centered:", center_first)
5 print("Raw squares:", subtract_totals)
```

```
Centered values: [-2. -1.  0.  1.  2.]
Mean of squares: 1.000000006e+18
Square of mean: 1.000000006e+18
Centered: 2.0
Raw squares: 0.0
```

- Centering first gives $(-2, -1, 0, 1, 2)$: the mean square is $(4 + 1 + 0 + 1 + 4)/5 = 2$.
- The other formula subtracts rounded values near 10^{18} . The small difference is lost: **catastrophic cancellation**.
- This is **100% relative error**: zero instead of two, not just a changed last decimal.

Numerical stability: choose computations that control the effect of rounding errors. A correct mathematical formula alone does not ensure a reliable computation.

Later: log probabilities help avoid tiny probabilities rounding to zero; softmax needs careful implementation to avoid overflow.

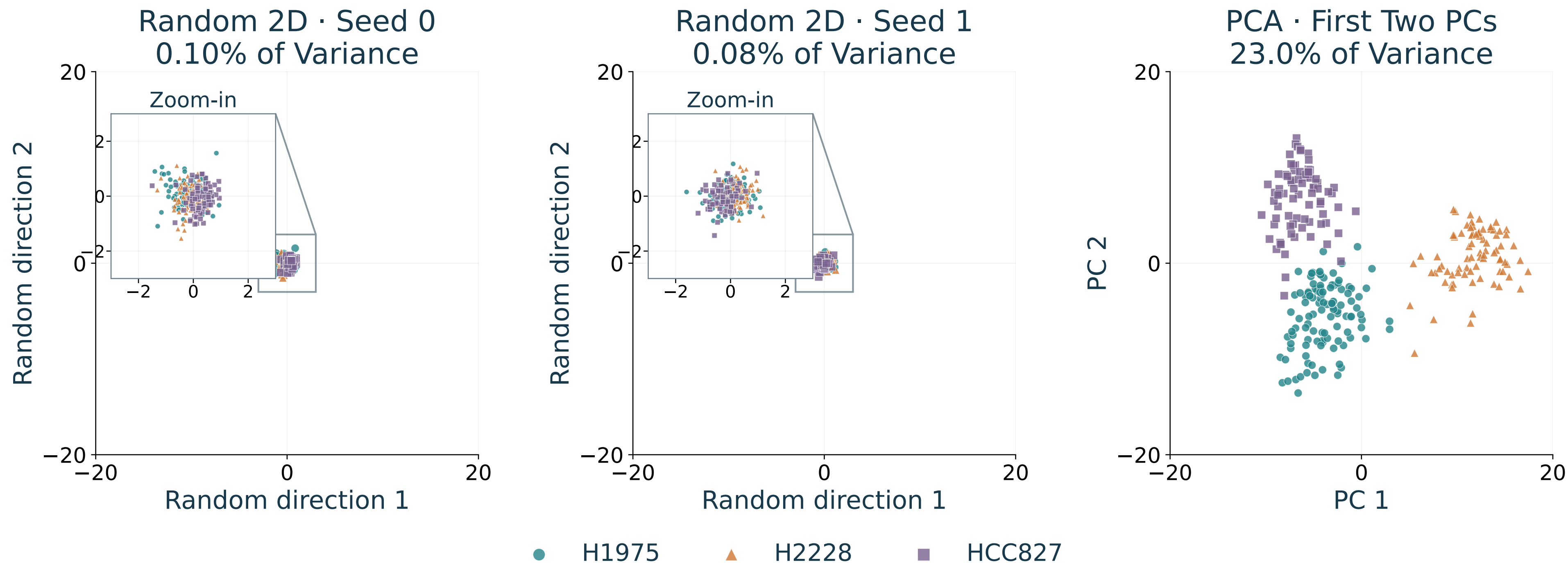
From computing PCA to interpreting real data

Back to our original question: what should a useful representation preserve?

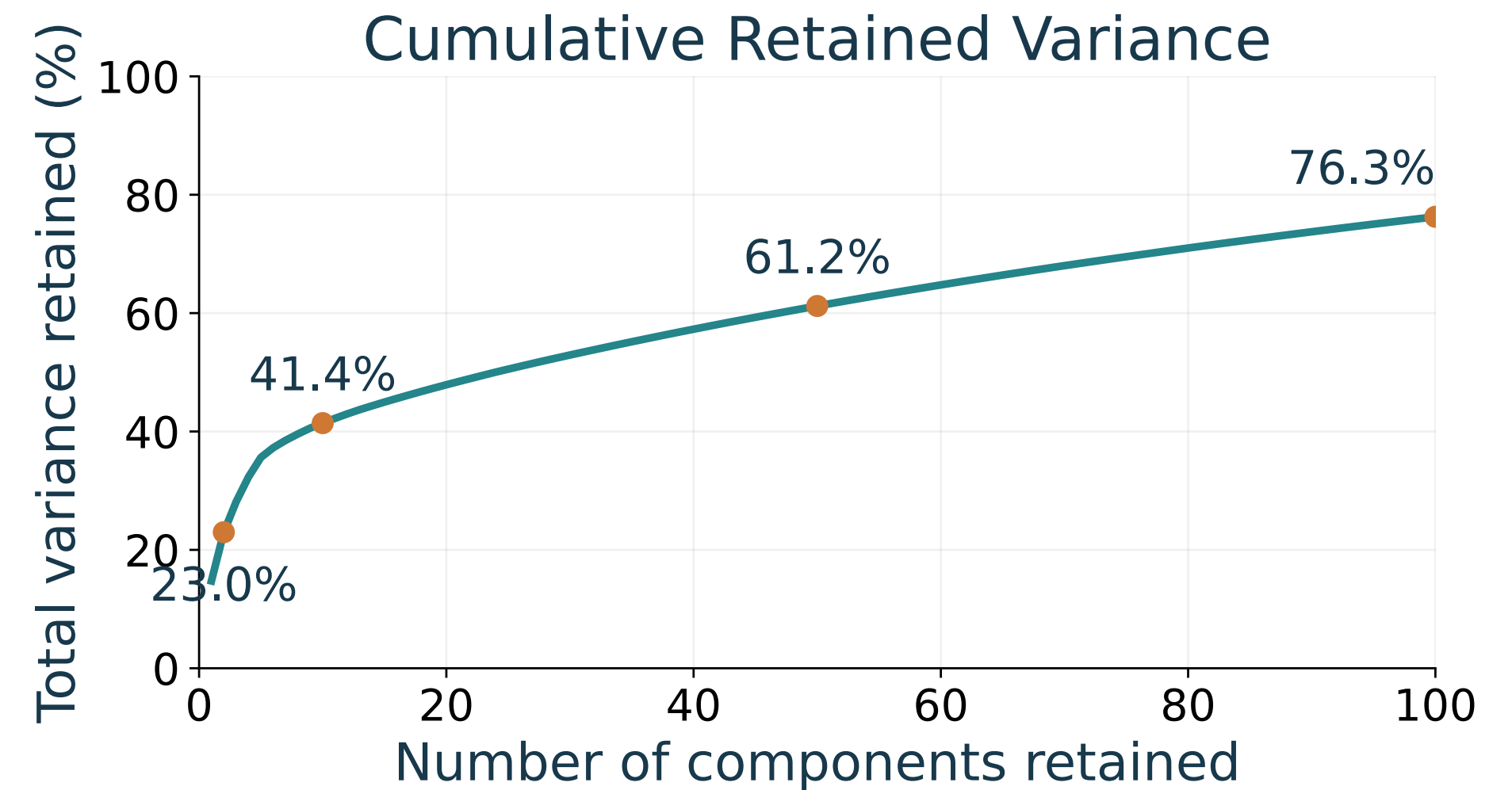
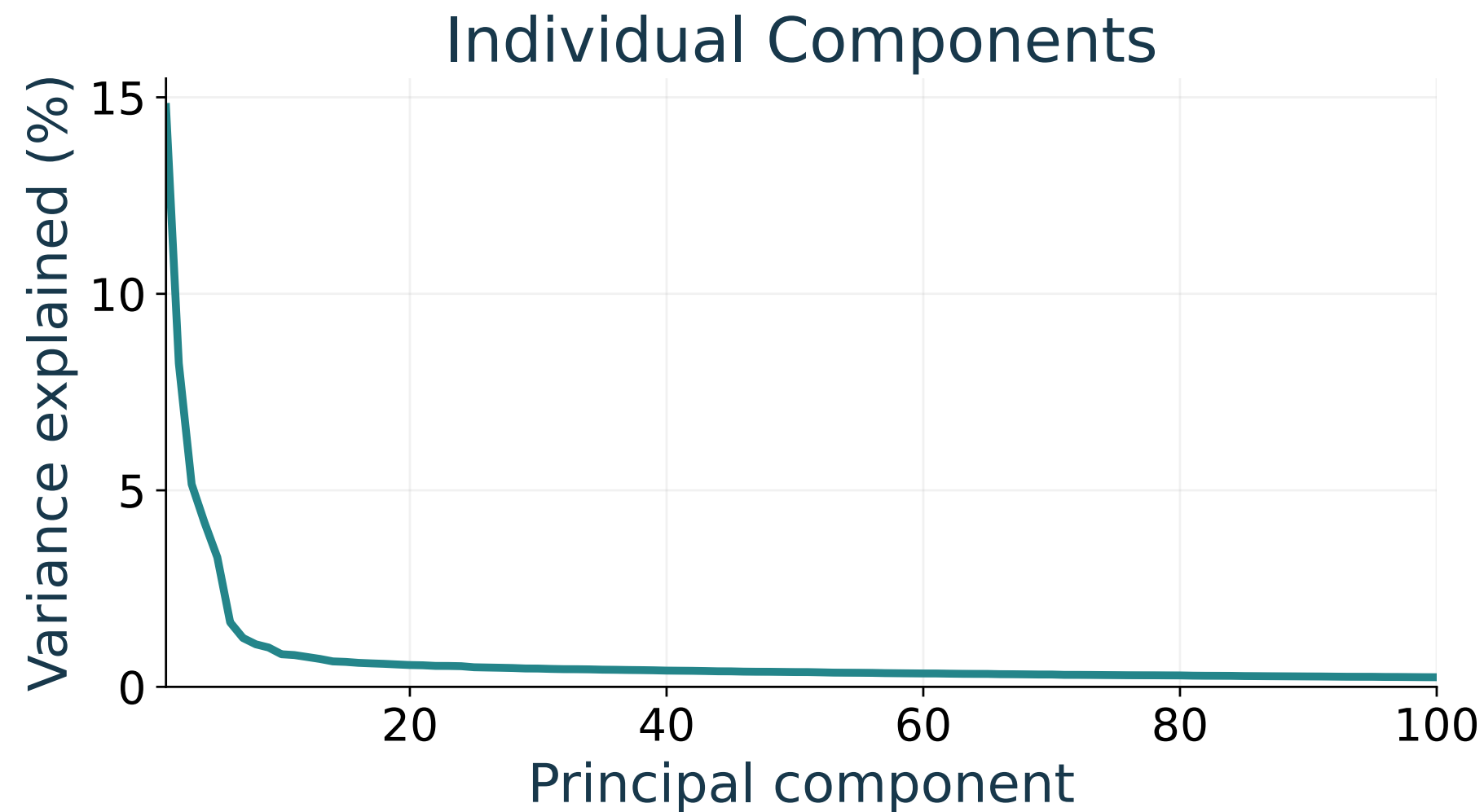
We can now compute PCA. Does it reveal useful structure in real gene-expression data?

| PCA reveals known cell-line structure **without seeing the labels**

Real gene expression: 274 cells from three lung-cancer cell lines; 2,000 selected genes per cell.



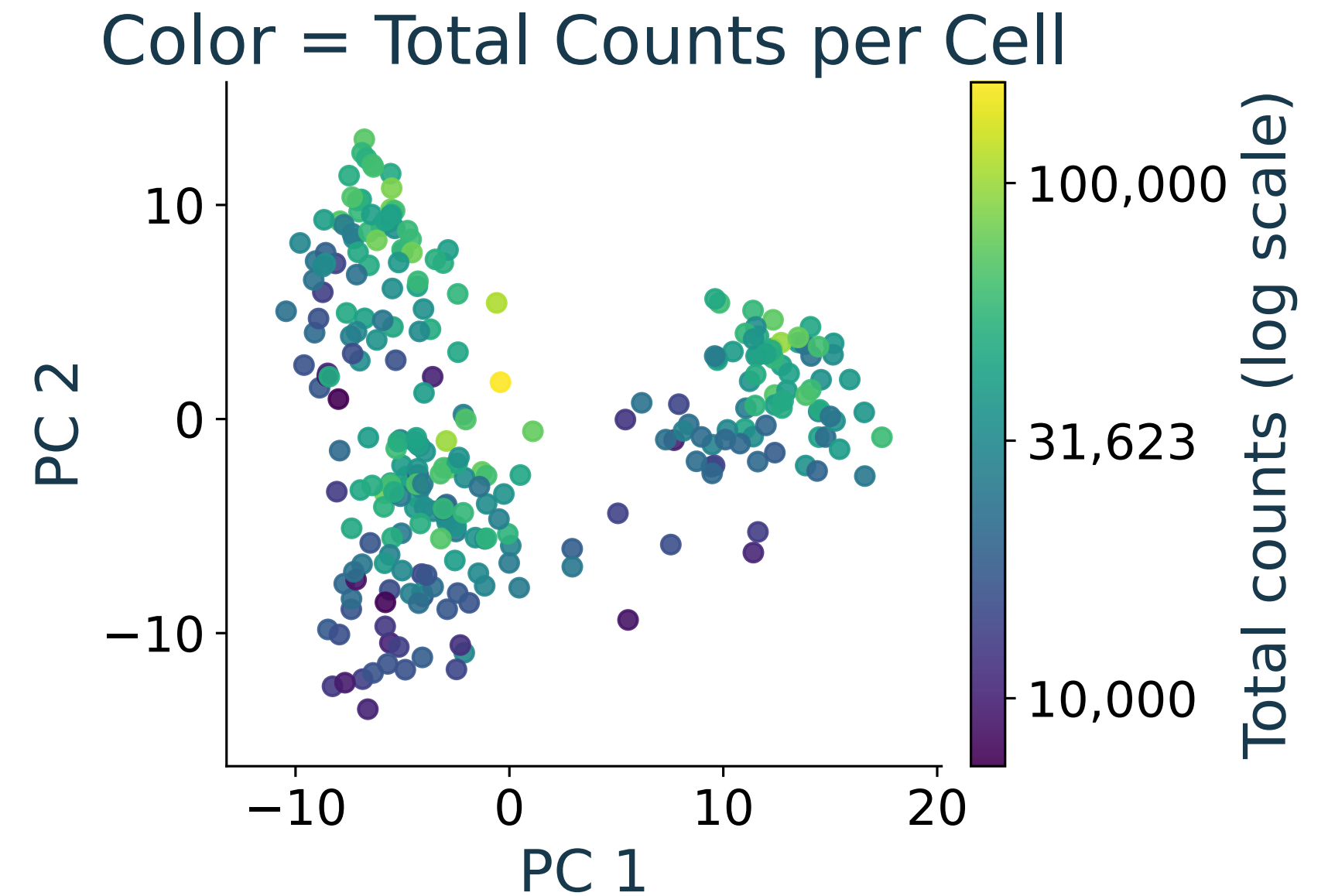
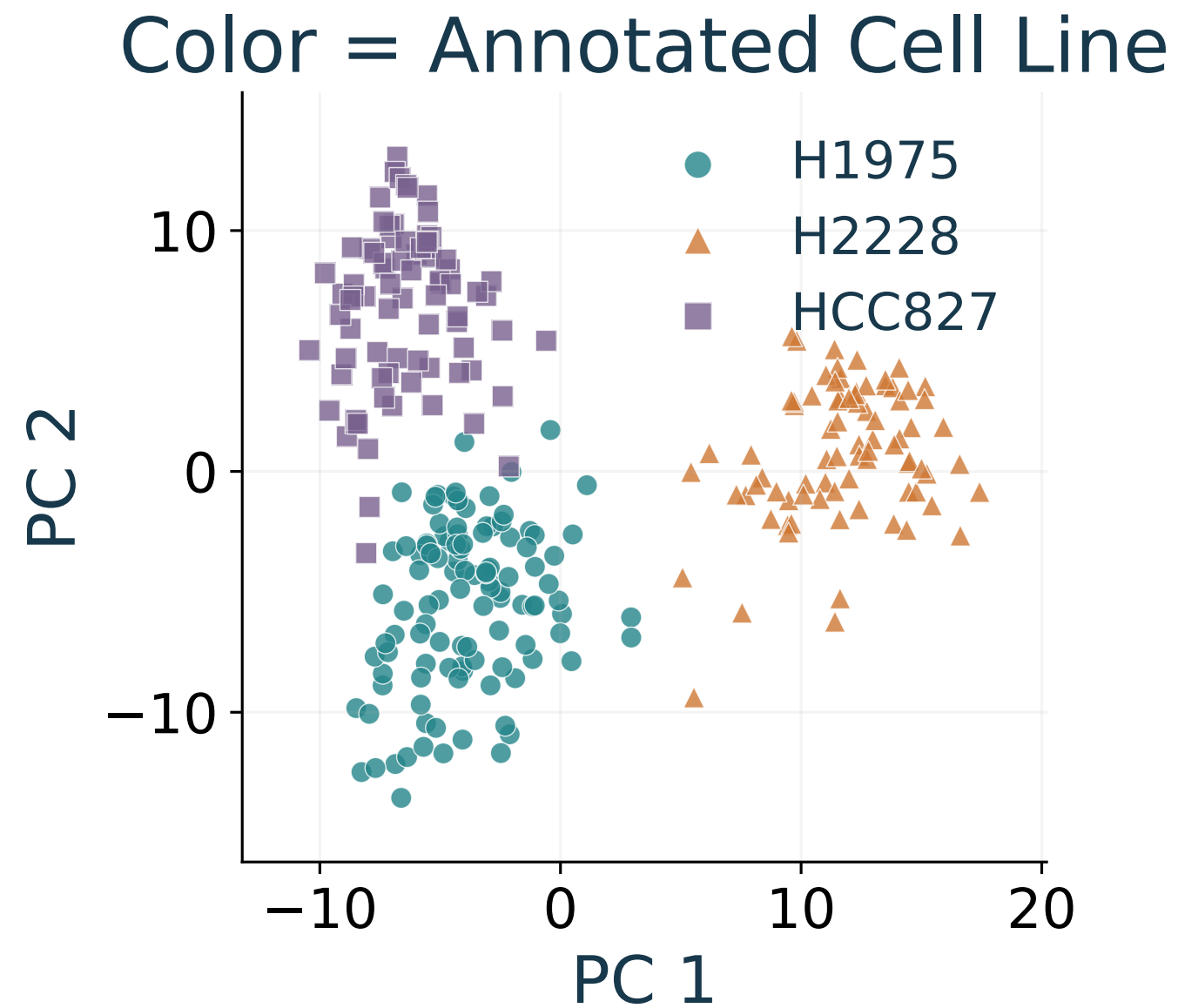
| Two PCs reveal groups but retain **only 23% of the variance**



$$\text{Retained fraction}(k) = \frac{\sum_{j=1}^k \sigma_j^2}{\sum_{j=1}^r \sigma_j^2}, \quad r = \text{rank}(X).$$

- **Visualization:** two coordinates already reveal cell-line structure.
- **Reconstruction:** 100 PCs retain 76.3%; reaching 90% requires 173 PCs.

| The same PCA plot supports different questions about usefulness



Discuss: Do these plots justify saying that PCA preserved the biology we care about?

- What does agreement with known cell-line labels support?
- What might total counts per cell explain—and what evidence would you want next?

| We fitted these observations; **new cells need new evidence**

From the original question...	...to what we established
Make thousands of measurements easier to inspect	Choose what the representation should preserve
Preserve observations under a squared-error criterion	PCA gives the best rank- k linear reconstruction
Compute that representation	Center, use SVD, and attend to finite precision
Inspect this cell dataset	Two PCs reveal known groups, but retain only 23% of variance

For new cells: keep the fitted preprocessing, gene set, mean, and directions fixed.

Before moving on: what general lessons does this case teach us about building useful representations?

| A useful solution begins with **deciding what to preserve**

Gene-expression measurements → desired representation → objective → constraints → solution

Choice we made	What it allowed us to specify
Which measurements, preprocessing, and units?	The observations and geometry
What should the representation preserve?	Squared reconstruction error
Which transformations are allowed?	A linear representation with k coordinates
What is the best solution under those choices?	Leading right singular vectors of centered data

Formalization makes a vague goal solvable. Choosing the goal remains our responsibility.

| Representations preserve **some structure and discard other structure**

In our cell-expression example	What it teaches us
Two PCs reveal known cell-line groups	A small representation can expose useful structure
Those two PCs retain only 23% of variance	Visible groups do not imply faithful reconstruction
100 PCs retain 76.3% of variance	The number of coordinates depends on the purpose

- Rank and discarded singular values describe **linear information loss**.
- Low-dimensional structure need not be linear: recall the curved-data example.
- Large variance is not automatically the variation that matters for our task.

Ask what was preserved, what was lost, and whether that matches the intended use.

A proof, a working computation, and useful evidence are **different achievements**

Question	What this arc established
Did we solve the stated problem?	SVD gives the optimal linear PCA reconstruction
Did we compute it correctly and efficiently?	Shapes, broadcasting, operation order, and finite precision matter
Does the result accomplish our purpose?	Real-data evidence supports particular claims, with limits

A mathematical guarantee does not establish every practical claim we might want to make.

Next question: What evidence would justify trusting a learned rule on new observations?

Binary Prediction: Goals and Historical Data

David I. Inouye

September 21, 2026

What would justify trusting a prediction for someone we have never seen?

| Different settings ask **the same kind of prediction question**

Setting	Information available	Question with two possible answers
Bank phone campaign	Customer attributes and prior contact history	Will this customer open a term-deposit account?
Wine screening	Chemical measurements such as acidity and alcohol	Will this wine receive a high sensory rating?
Activity recognition	Measurements from motion sensors	Is this person walking?

In each case, use available information to predict **an outcome we do not yet know**.

Our running case: will a customer open a **term-deposit account**—depositing money for a fixed period to earn interest?

| A useful prediction begins with a precise question

A bank calls customers to offer a **term-deposit account**: money deposited for a fixed period to earn interest.

Ask first	Our initial specification
Who is the prediction about?	A new customer in the population we intend to contact
When do we need the answer?	Before calling the customer
What may we use?	Information available at that moment
What outcome are we predicting?	Whether the customer opens the offered account during the campaign
What does “best” mean for now?	Make as few prediction mistakes as possible on new customers

How can we express that goal precisely enough to reason about it?

Binary classification turns the question into **inputs, outcomes, and a predictor**

Object	Meaning in the bank case
$x \in \mathcal{X}$	Customer information available before calling
$y \in \{0, 1\}$	Actual outcome: did not open (0) or opened (1) the offered account
$f : \mathcal{X} \rightarrow \{0, 1\}$	A rule that predicts an outcome from the available information
$\hat{y} = f(x)$	Our prediction for this customer

$$x \xrightarrow{f} \hat{y} \in \{0, 1\}.$$

A prediction is correct when $f(x) = y$. The same abstraction describes all three opening cases.

Which predictor would we want if we could choose the best one?

| The goal is minimum error **on new customers**

A **distribution** p describes how likely different customer–outcome pairs are. For a randomly selected customer, $\mathbf{X}$ is their information and Y is what actually happens: these are **random variables** drawn jointly from p .

Intuition: how likely is our prediction $f(\mathbf{X})$ to match Y ? We formalize the opposite event—the **population error**:

$$R_p(f) = \Pr_{(\mathbf{X}, Y) \sim p} (f(\mathbf{X}) \neq Y).$$

Goal: choose $f^* \in \arg \min_{f: \mathcal{X} \rightarrow \{0,1\}} R_p(f)$.

- Minimizing error is equivalent to maximizing accuracy, $1 - R_p(f)$.
- This specifies our goal; it does not yet tell us how to find or evaluate f .

We know what we want. What information do we actually have?

| We have **historical examples**, not future answers

A real bank dataset records customer information and whether the customer opened the offered account.

Record	Age	Job	Housing loan?	Opened account?
1	56	Housemaid	No	No
3	37	Services	Yes	No
76	41	Blue-collar	Yes	Yes

Selected actual records from a term-deposit phone campaign; not a representative sample.

We call the labeled examples available for constructing a predictor the **training data**:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n.$$

When can outcomes for past customers tell us something about new customers?

| Using the past requires a connection to the future

Working assumption: new customers resemble past customers in both their available information and how that information relates to opening the offered account.

Our clean mathematical model is **i.i.d. sampling**:

- **Identically distributed:** past and new input–outcome pairs come from the same p .
- **Independent:** one sampled example does not supply information about another draw.
- The new case is independent of the historical sample used to construct f .

Same distribution does not mean the same people or exactly repeated feature values.

Before using the historical table, will its inputs even be available at prediction time?