
Choosing Models and Evaluating the Whole Pipeline

ECE 57000 — September 28, 2026

David I. Inouye

| Friday connected **model flexibility and trustworthy evaluation**

- At a fixed input, squared prediction error separates into squared bias, training-sample variance, and unresolved outcome noise.
- Changing k , available data, or representation changes which nearby evidence the model uses.
- Validation chooses a pipeline; its winning score is optimistic on average. Final test data must stay outside the entire selection process.

Today: finish the development/evaluation boundary, then examine how finite and imperfectly matched evidence limits our claims.

| Separate validation for choosing from test data for reporting

Reserve all three roles **before development**:

Data	Role	What may it influence?
Training	Fit each candidate.	Its learned parameters or stored examples.
Validation	Compare candidates and choose k .	The selected pipeline.
Test	Score the settled pipeline once.	The reported performance estimate.

Freeze the pipeline before using the test outcomes. An independent, representative test set gives an **unbiased error estimate for that fixed pipeline**, with finite-sample uncertainty.

| Refit the chosen pipeline **before opening the test set**

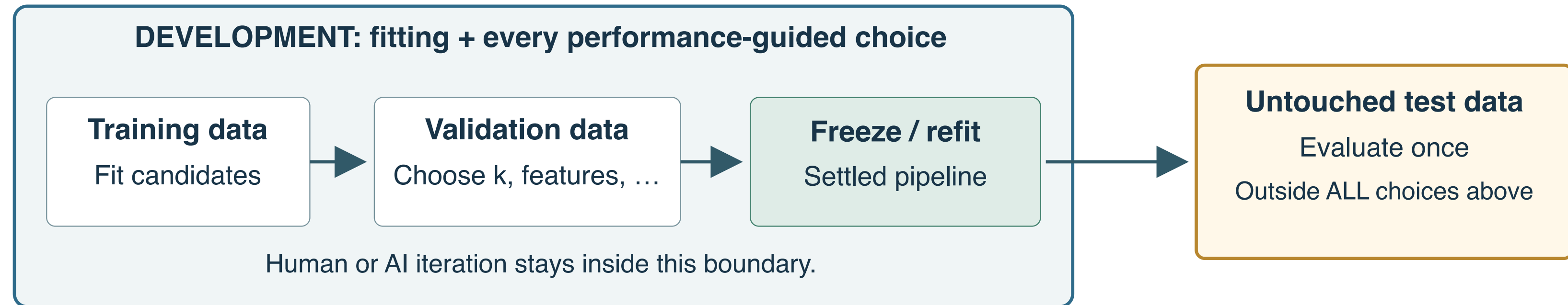
After choosing k and all other pipeline settings, we may fit once more on **training + validation data** to use all development examples.

$$\underbrace{\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{validation}}}_{\text{refit with settled choices}} \longrightarrow f_{\text{final}} \xrightarrow{\mathcal{D}_{\text{test}}} \widehat{R}_{\text{test}}(f_{\text{final}}).$$

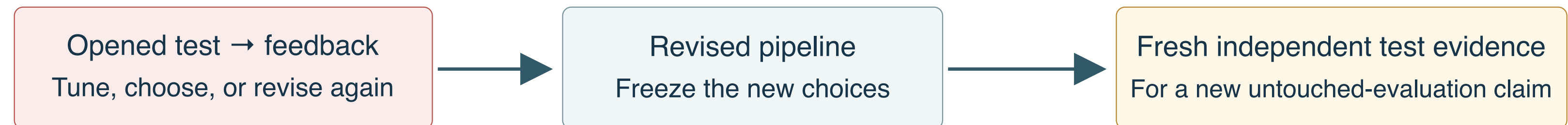
For KNN, this means storing all development examples with the chosen k and refitting any learned preprocessing on those development examples.

The test result describes this final predictor. Changing it after seeing the test result turns that test into more development evidence.

| Keep final testing outside **every development feedback loop**



If test results guide another change, the old test joins development.



One final test can cover all development choices. A new split is needed when we reuse final-test feedback to make further choices—not for every fitting step.

5. When and why should a learned solution hold?

A test score becomes useful when we understand its uncertainty and the scope of its evidence.

| Interpret the score through **three connected questions**

Our reasoning process	Question to resolve
1. What did the test measure? ← now	How much evidence is behind this score?
2. Does it represent deployment?	Could dependence, population, or available information differ?
3. What can we claim?	State the supported result and the conditions for extending it.

We protected the test from selection. Now interpret what it tells us.

| A rough 95% interval is **the score $\pm$ two standard errors**

Fix the model before testing on m independent examples from the target population. Let $\hat{a}$ be the fraction of test predictions that are correct.

Approximate 95% interval: $\hat{a} \pm 2\sqrt{\frac{\hat{a}(1 - \hat{a})}{m}}.$

900 correct out of 1,000: 90% accuracy, with a margin of about **1.9 percentage points**.

“With approximately 95% confidence, this fixed model’s accuracy in this population is between **88% and 92%.**”

| More test examples support more precise claims

Suppose measured accuracy is **90%** in each case:

Independent test examples	Correct predictions	Rough 95% margin (pp)
10	9/10 = 90%	±19†
30	27/30 = 90%	±11†
100	90/100 = 90%	±6
1,000	900/1,000 = 90%	±1.9
2,000	1,800/2,000 = 90%	±1.3
5,000	4,500/5,000 = 90%	±0.8
10,000	9,000/10,000 = 90%	±0.6

10–30: wide uncertainty. **1,000–10,000:** roughly 2-point to sub-1-point precision here.

pp = percentage points. Margins use the previous slide’s rough formula. †Small-sample scale only: use Wilson for an actual interval. Fixed model;

| Next ask whether the evidence matches deployment

Our reasoning process	What we have established
1. What did the test measure?	A count of errors from a finite sample.
2. Does it represent deployment? ← now	Check independence, population, and available information.
3. What can we claim?	Combine the measured result with the scope of its evidence.

A very large test set can measure the wrong deployment population very precisely.

| Repeated rows need not provide **independent evidence**

Imagine 100 customers, each represented by 10 related campaign records.

- A random row split may put the **same customer** in both training and test data.
- A model may exploit person-specific information that will not help on a new customer.
- The 1,000 rows also need not provide the precision of 1,000 independent customers.

If the claim concerns new customers, keep each customer in one split.

| Independent examples can still come from **a different population**

$$R_p(f) = \Pr_{(\mathbf{x}, y) \sim p} \{f(\mathbf{x}) \neq y\}, \quad R_q(f) = \Pr_{(\mathbf{x}, y) \sim q} \{f(\mathbf{x}) \neq y\}.$$

- The test set estimates performance under p .
- Deployment may draw customers and outcomes from $q \neq p$: distribution shift.
- **Out-of-distribution (OOD) performance** depends on q ; even perfect knowledge of $R_p(f)$ does not determine $R_q(f)$.

What changes operationally—and could it change who responds to our offer?

| Would this new contact list preserve accuracy?

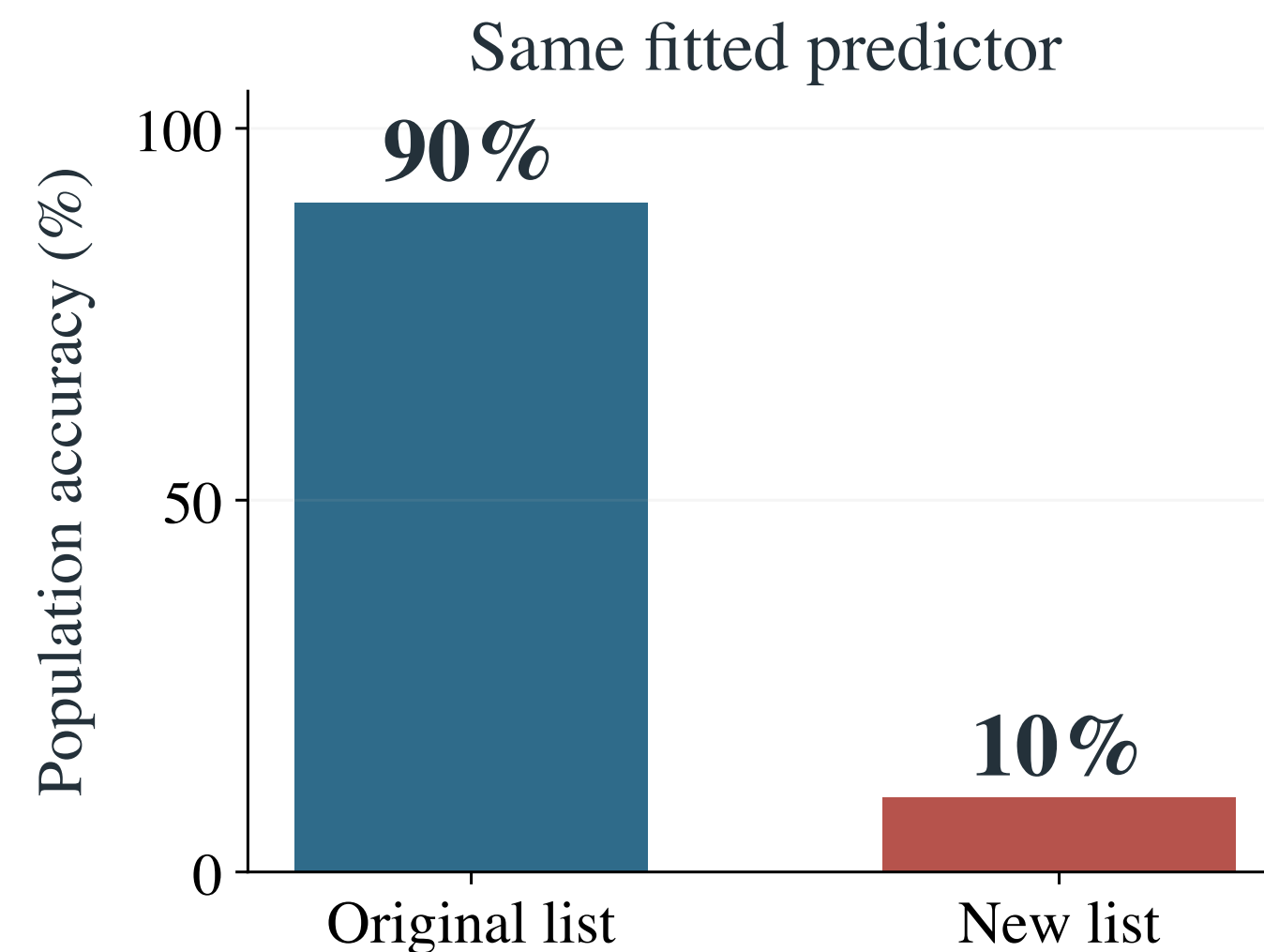
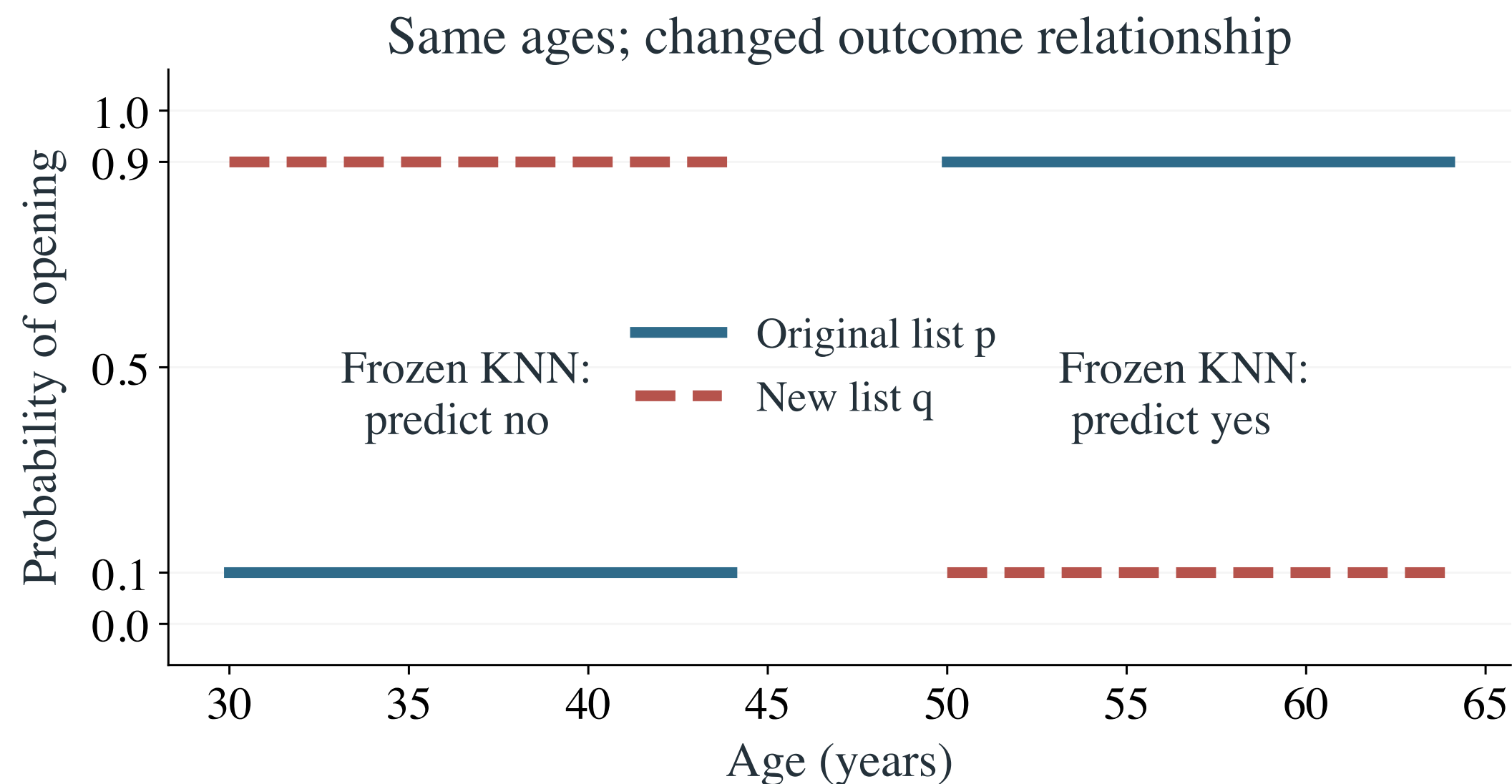
Synthetic bank campaign: a frozen age-only KNN model is 90% accurate under the original list. A new supplier provides the **same age distribution** and **same overall account-opening rate**.

Do those checks establish similar accuracy on the new list?

- A. Yes: the input distribution and overall outcome rate are unchanged.
- B. No: the relationship between age and opening may have changed.
- C. Yes, if we make the original-list test set much larger.

B. Matching the separate summaries does not establish a matching joint relationship.

| The unchanged predictor falls from 90% to 10% accuracy



Synthetic bank example. Both lists: half ages 30–44, half ages 50–64; uniform ages within each group; **50% open overall.** Only the relationship between age and outcome changes.

The old 90% remains valid for the old population. More old-list data cannot establish new-list accuracy.

| Which test matches the claim about new people?

A walking classifier uses many sensor windows from each volunteer. Deployment will involve **people absent from development data**.

Which split most directly tests that claim?

- A. Randomly assign windows to training and test sets.
- B. Put each volunteer entirely in development or entirely in test.
- C. Test later windows from volunteers already used for training.

B. It tests unseen people. It still does not establish transfer to different sensors, activities, or collection settings.

| Choose the evaluation design **from the deployment claim**

Intended claim	More relevant evidence	What still needs reasoning?
More cases from the same population	Random held-out examples	Does the sample represent deployment?
Future campaigns	Train earlier, test on a later campaign	Will later changes resemble this one?
New customers	Keep each customer entirely in one split	Are these customers representative?

All designs require **prediction-time inputs**: current-call duration remains unavailable before the call. A split cannot repair that information mismatch.

| Finally state **what the evidence supports**

Our reasoning process	What we have established
1. What did the test measure?	Report the errors and number of test examples.
2. Does it represent deployment?	Examine dependence, changed populations, and available inputs.
3. What can we claim? ← now	State the supported result and the conditions for extending it.

A useful number comes with a reasoned account of where it applies.

| Report a result, its scope, and its uncertainty

Illustrative report: “The frozen before-call predictor made 100 errors on 1,000 independent held-out customers from supplier A’s campaign: **10% error.**”

- **Scope:** one prediction per customer; account opening as the outcome; test data excluded from all development choices.
- **Uncertainty:** roughly **10% $\pm$ 1.9 percentage points** at 95% confidence.
- **Extension:** a different supplier may change response patterns within age groups. This result does not establish its error without evidence connecting the populations.

Say what may change, why it matters, and what evidence would resolve it.

| From interpreting evidence to **rethinking the predictor**

So far: What does a settled model's test score tell us about deployment?

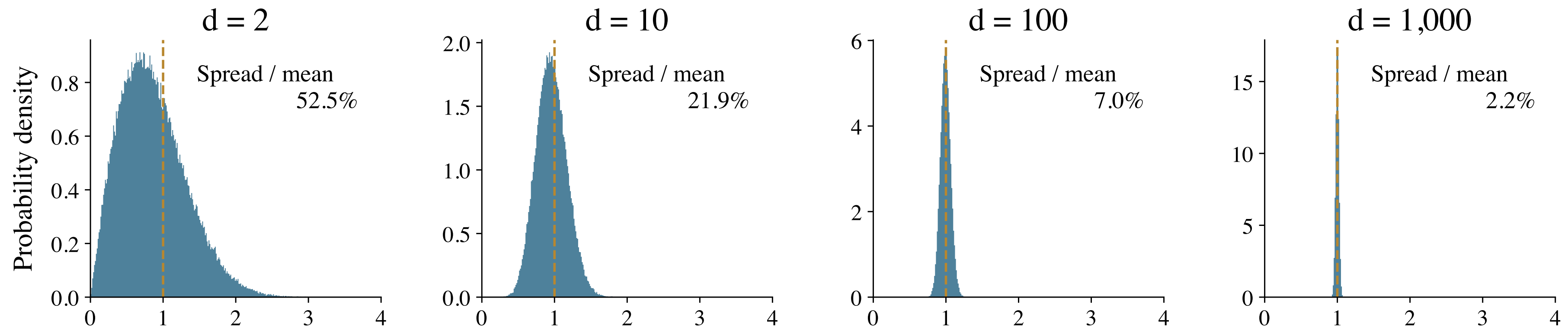
Now: When does KNN's notion of “nearby” stop being useful—and what should we learn instead?

Our next steps:

1. See how near and far distances become less distinct in high dimensions.
2. Test what happens when irrelevant features distort distance.
3. Motivate a learned score that emphasizes predictive information.

| In high dimensions, **near and far become less distinct**

Draw **500 points** with independent standard-normal coordinates: $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. Plot every pairwise distance.



Euclidean distance / $\sqrt{2d}$ (same horizontal scale in every panel)

Typical raw distance grows like $\sqrt{2d}$; its relative spread shrinks. Dividing by $\sqrt{2d}$ lets us compare dimensions on the same scale.

A “near” point is only slightly closer than a “far” point: **distance loses contrast.**

| Add measurements that contain **no new information**

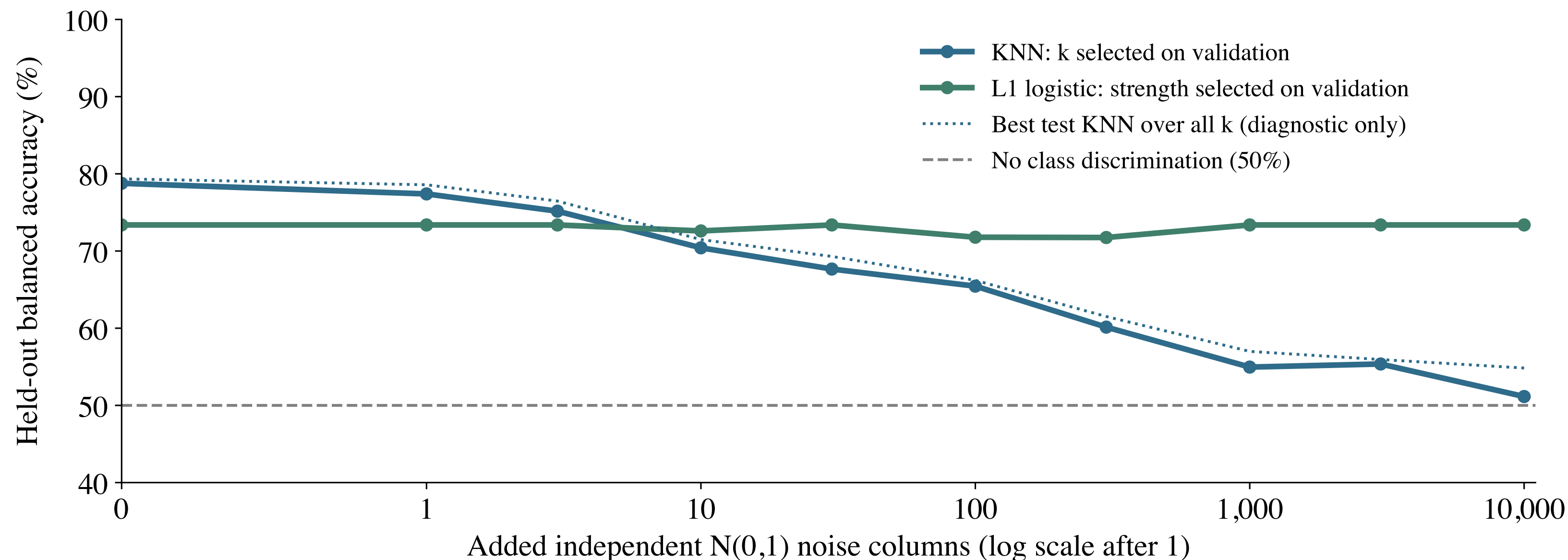
Return to our **simulated bank campaign**: age and prior contacts predict opening. Keep every customer and outcome fixed; append independent Gaussian columns.

$$\mathbf{x}_{\text{aug}} = [\text{age}, \text{prior contacts}, z_1, \dots, z_d], \quad z_j \stackrel{\text{iid}}{\sim} \mathcal{N}(0, 1).$$

- Noise is independent of both the original features and the outcome.
- Standardize using training data; select k or regularization on validation.
- **Same split**: 500 training, 500 validation, 2,000 test customers.

Available predictive information is unchanged. Is distance still useful?

| Independent noise can make neighbors **nearly uninformative**



Balanced accuracy: average of the two class recall rates; **50% means no discrimination**. At 10,000 noise columns: **KNN 51.1%; learned-score model 73.4%** (preview of Arc 4).

| Noise dominates distance; **which coordinates should matter?**

$$\|\mathbf{x}_{\text{aug}} - \mathbf{x}'_{\text{aug}}\|^2 = \underbrace{\|\mathbf{x}_{\text{signal}} - \mathbf{x}'_{\text{signal}}\|^2}_{\text{two relevant coordinates}} + \underbrace{\sum_{j=1}^d (z_j - z'_j)^2}_{\text{many irrelevant coordinates}} .$$

- Independent noise differences accumulate and scramble which customers are nearest.
- Increasing k averages more labels; it does not identify the relevant coordinates.
- Could we **learn which coordinates matter**, giving irrelevant ones less influence?

High dimension alone is not the verdict: correlated features can carry redundant, useful structure. Real data need not fill the whole ambient space.

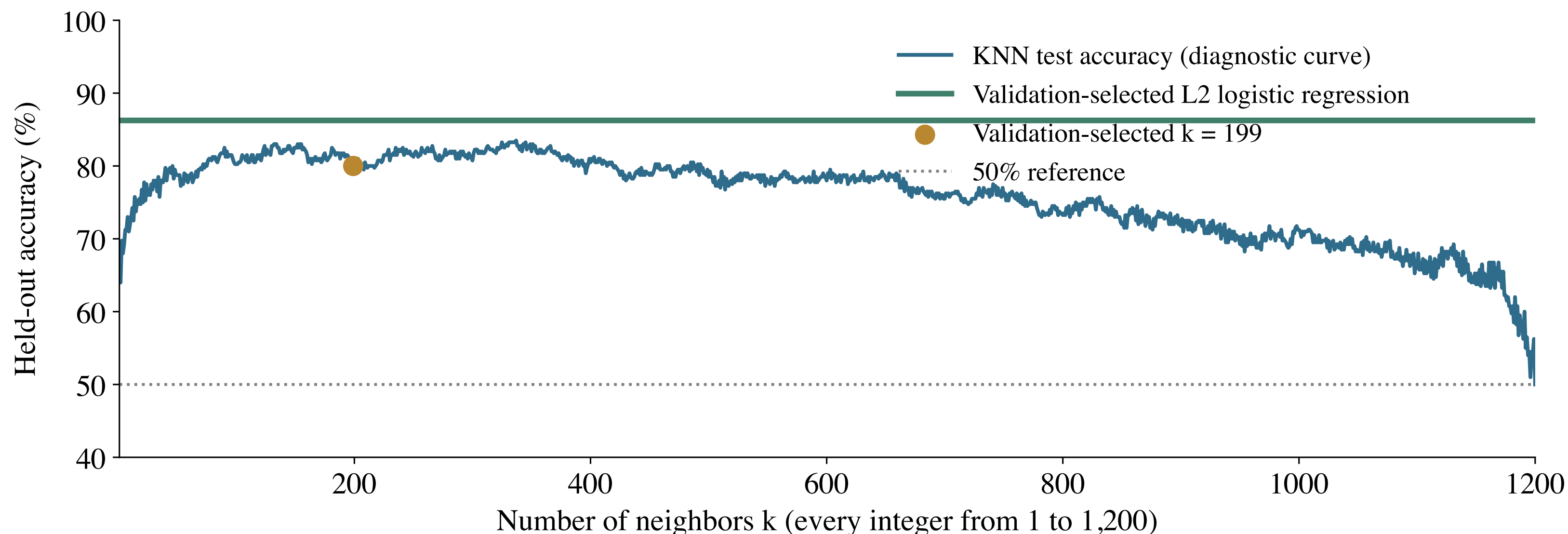
| Real-world example: **movie-review sentiment**

- **2,000 real reviews:** predict positive or negative sentiment.
- **18,746 vocabulary words:** represent each review with one numerical feature per word.
- Shared words need not mean shared sentiment: **“good” versus “not good.”**

A new idea: learn a positive or negative score for each word and combine the contributions. This leads to a **linear model**, which we will study next.

| On real reviews, a learned score **outperforms every tested k**

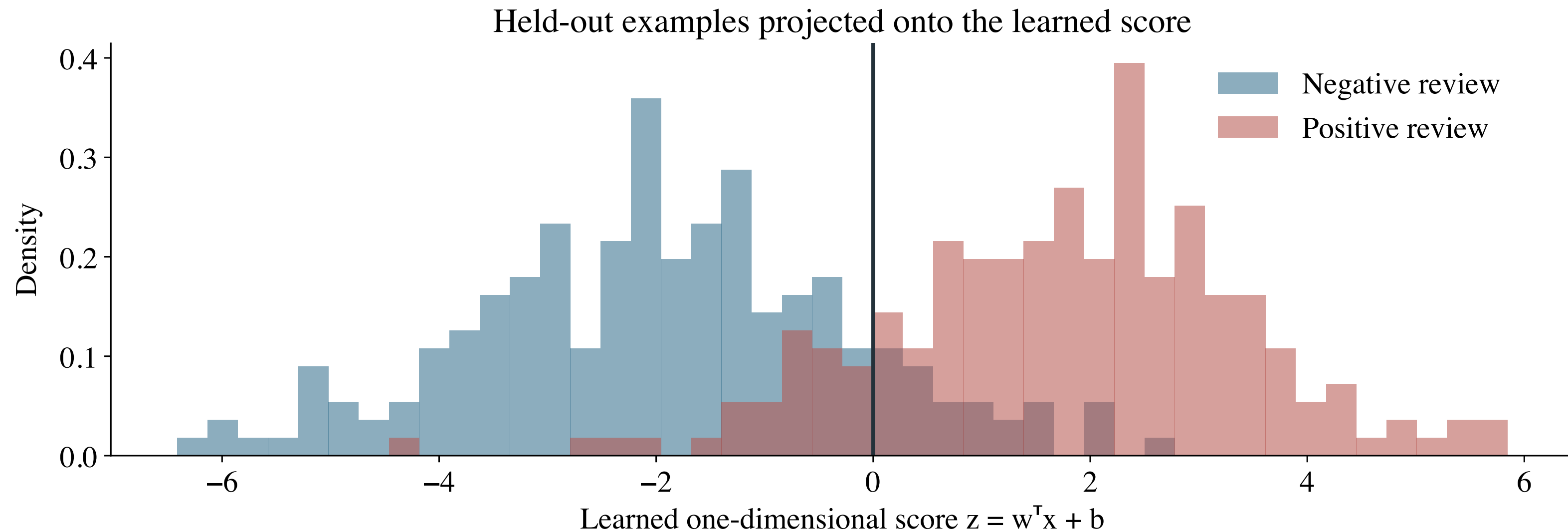
Cornell movie reviews: positive versus negative sentiment; **18,746 word features**. Train: 1,200 reviews; validate: 400; test: 400. No synthetic examples or added noise.



Validation-selected test accuracy: **KNN 80.0%; learned word-score model 86.25%**. Meaningful word features are not independent noise: **KNN still has useful predictive signal.**

| A useful one-dimensional projection **separates the outcomes**

$$x \in \mathbb{R}^{18746} \longrightarrow z = w^T x + b \in \mathbb{R} \longrightarrow \hat{y} = \mathbf{1}\{z \geq 0\}.$$



The weights are **learned from training reviews**; these distributions show held-out reviews.

| Next: learn the predictive direction from labels

KNN asks which stored examples are nearby. A **linear model** learns how much each input contributes to one predictive score.

PCA preserves **input variation**; our goal is to separate **outcomes**.

Arc 4: How do labels tell us which weights to learn—and what can a linear score miss?