UNIVERSIDAD DE CHILE

Minería de Datos

Welcome to the Machine Learning class

Valentin Barriere

Universidad de Chile – DCC

CC5205, Fall 2025

Dimensionality Reduction

Motivations for Lower Dimensionality

Outline : Motivations for Lower Dimensionality

Feature Selection and Dimensionality Reduction

General Motivation

- In many learning problems, we may have irrelevant or redundant features.
- These features can negatively affect our classifiers' performance or make training more expensive.
- To address this, there are two main approaches:
  - Feature selection (supervised): choose a subset of features that is most useful for the task (classification, regression, etc.).
  - Dimensionality reduction (unsupervised): find a lower-dimensional projection that preserves the information in the data.

Reasons to Select Features

- Decision Trees can be affected by irrelevant features: although the tree tries to pick relevant ones, in practice it may "learn" noise when depth is high or data become fragmented.
- KNN is very sensitive to irrelevant features, since all dimensions count equally in distance calculations. A noisy feature can distort neighbor proximity.
- Naïve Bayes tends to be robust to irrelevant features (it ignores them because they don't significantly change the posterior), but suffers from **redundant** highly correlated features.

Reasons to Select Features

- Decision Trees can be affected by irrelevant features: although the tree tries to pick relevant ones, in practice it may "learn" noise when depth is high or data become fragmented.
- KNN is very sensitive to irrelevant features, since all dimensions count equally in distance calculations. A noisy feature can distort neighbor proximity.
- Naïve Bayes tends to be robust to irrelevant features (it ignores them because they don't significantly change the posterior), but suffers from **redundant** highly correlated features.

In general, this aims to mitigate the harmful effects of the *curse of dimensionality*

Feature Selection

Outline : Feature Selection

Outline : Feature- or Model-based

Difference Between Filter (Feature-based) and Wrapper (Model-based)

Feature-based or Scheme-independent (Filters)

- Evaluate features based on **general data properties** (e.g., variance, correlation, mutual information).
- Do not use a specific classifier to measure each feature's relevance.
- They are **fast** and less prone to overfitting, but they **do not consider** complex feature–task interactions.

Model-based or Scheme-dependent (Wrappers)

- Select features by **training a classifier** (or regressor) and evaluating its performance on each subset.
- Their goal is to maximize that model's predictive performance.
- They are **computationally more expensive** and can overfit easily.
- They find feature subsets that are **more tailored to the task**.

Outline : Feature-based (Univariate)

Feature-based Methods (Scheme-independent)

Key Characteristics

- Do not depend on a specific model to assess feature relevance $\Rightarrow$ Univariate Feature Selection
- Use **metrics** such as entropy, Information Gain, correlation, chi-square (χ^2), etc.
- Some popular examples include:
  - *Information Gain* (entropy-based)
  - *Mutual Information*
  - *Correlation-based Feature Selection (CFS)*
  - *Low Variance*
- They are **fast** and easy to implement, though they **do not consider** interaction with the final classifier model.

Mutual Information and χ^2 are especially useful for sparse data.

Correlation-based Feature Selection (CFS)

- In the scheme-independent (or **filter**) approach, a feature subset is evaluated using a general data-driven metric.
- **CFS** seeks features that:
  - Have high correlation with the class.
  - Have low inter-correlation (to avoid redundancy).
- Correlation between categorical features is measured using *symmetric uncertainty*:

$$\mathrm{SymmUnc}(A, B) = \frac{2 \times \big(H(A) - H(A \mid B)\big)}{H(A) + H(B)} = \frac{2 \times IG(A, B)}{H(A) + H(B)},$$

where $H(\cdot)$ is entropy and $IG(A, B)$ is the information gain.

$$H(A \mid B) = H(A, B) - H(B);\ H(x, y) = -\sum_{x,y} p(x, y) \log p(x, y).$$

Subset Search Strategies

- The total number of feature subsets is exponential in the number of features (2^n).
- **Greedy heuristics** are used to avoid searching the whole space:
  - **Forward Selection**: start with no features and add them one by one if they improve the criterion.
  - **Backward Elimination**: start with all features and remove those that do not contribute.
- More sophisticated methods include:
  - **Best-first search**, **Beam search**, **Genetic algorithms**, etc.

Outline : Model-based (Scheme-specific)

Feature Selection Based on Importance

You can select, post-training, the top *n* most important features of a trained model by using its feature importance scores.

- Assigns an **importance score** to each feature.
- Relies on a model attribute (e.g. `coef_` or `feature_importances_`) or on a callable that returns feature importances after fitting.
- Features with importance below a certain **threshold** (`threshold`) are removed.
- The `max_features` parameter can set an upper limit on the number of selected features.

More information at scikit-learn

ℓ_1 Regularization for Feature Selection

- ℓ_1 **(Lasso)** induces sparsity: some coefficients become zero.
- Useful for feature selection in linear models (linear regression or logistic regression).
- The ℓ_1 penalty term is:

$$\lambda \sum_j |w_j|$$

- Features whose coefficients are driven to zero can be discarded without affecting the model.
- Can be used with `SelectFromModel` in scikit-learn, e.g. `Lasso` or `LogisticRegression` with ℓ_1 penalty.

Wrapper-based Feature Selection

- The wrapper approach evaluates each feature subset by training a model and measuring its **performance** (e.g. accuracy, F1, AUC).
- It is **computationally expensive**, since the model must be retrained for every subset (e.g. within a 10-fold CV).
- Under a greedy strategy, each candidate subset can lead to $O(m^2)$ or even $O(2^m)$ complexity in the worst case (exhaustive search).
- Works well with lightweight models like Naïve Bayes, but becomes prohibitive with many features.

Recursive Feature Elimination (RFE)

General Concept

- RFE uses an external estimator that assigns **weights** to features (e.g. coefficients of a linear model or feature importances of a tree).
- The goal is to **select features** by recursively working on smaller subsets.
- Procedure:
  1. Train the estimator on the initial feature set.
  2. Obtain the **importance** of each feature (e.g. `coef_` or `feature_importances_`).
  3. **Eliminate** the least important features.
  4. Repeat **recursively** on the new reducted feature set, until the desired number of features is reached.

Dimensionality Reduction

Outline : Dimensionality Reduction

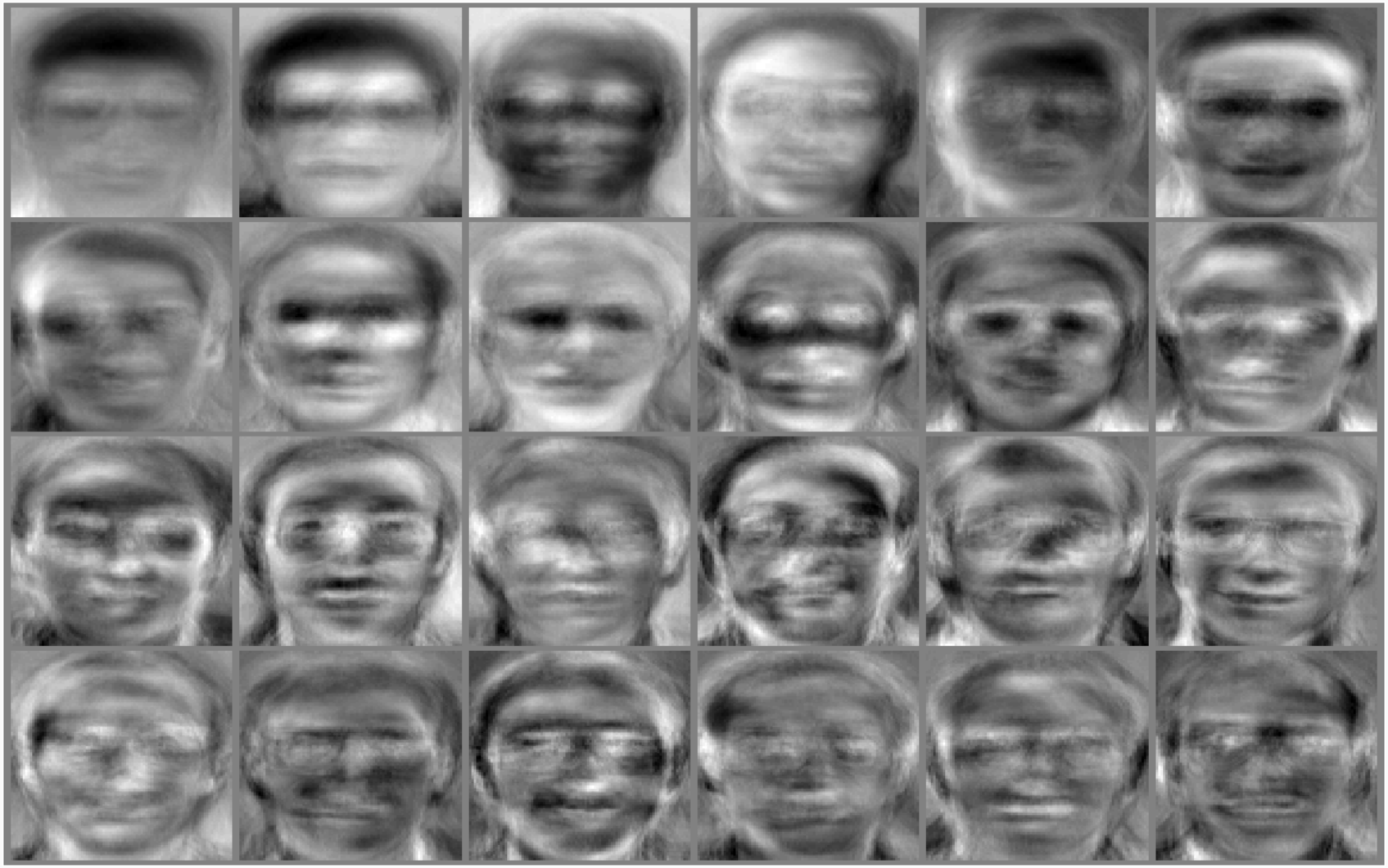
Benefits of Dimensionality Reduction

- **Simplify data**: Fewer features mean less noise and better generalization.
- **Reduce computational cost**: Models with many features can be very expensive to train and use.
- **Visualization**: Projecting high-dimensional data to 2D or 3D enables intuitive graphical interpretation.
- **Avoid the curse of dimensionality**: Distance-based methods degrade in very high dimensions.
- **Data compression**: Reduces the memory footprint of a dataset.

Outline : Principal Component Analysis

PCA: Principal Component Analysis

Uses the eigenvectors of the data covariance matrix to find the most representative directions (principal components).

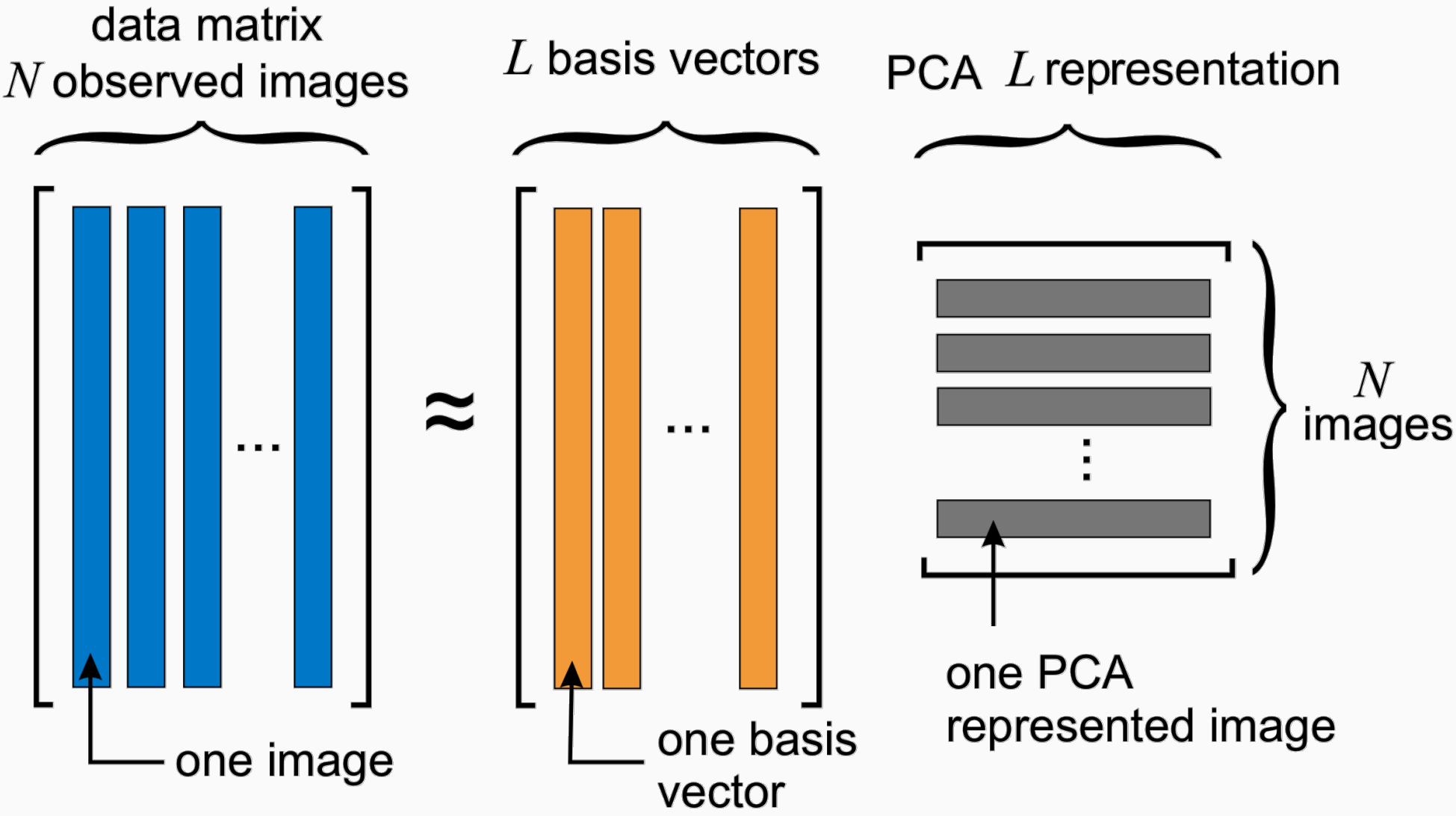


PCA: Motivation

Motivation

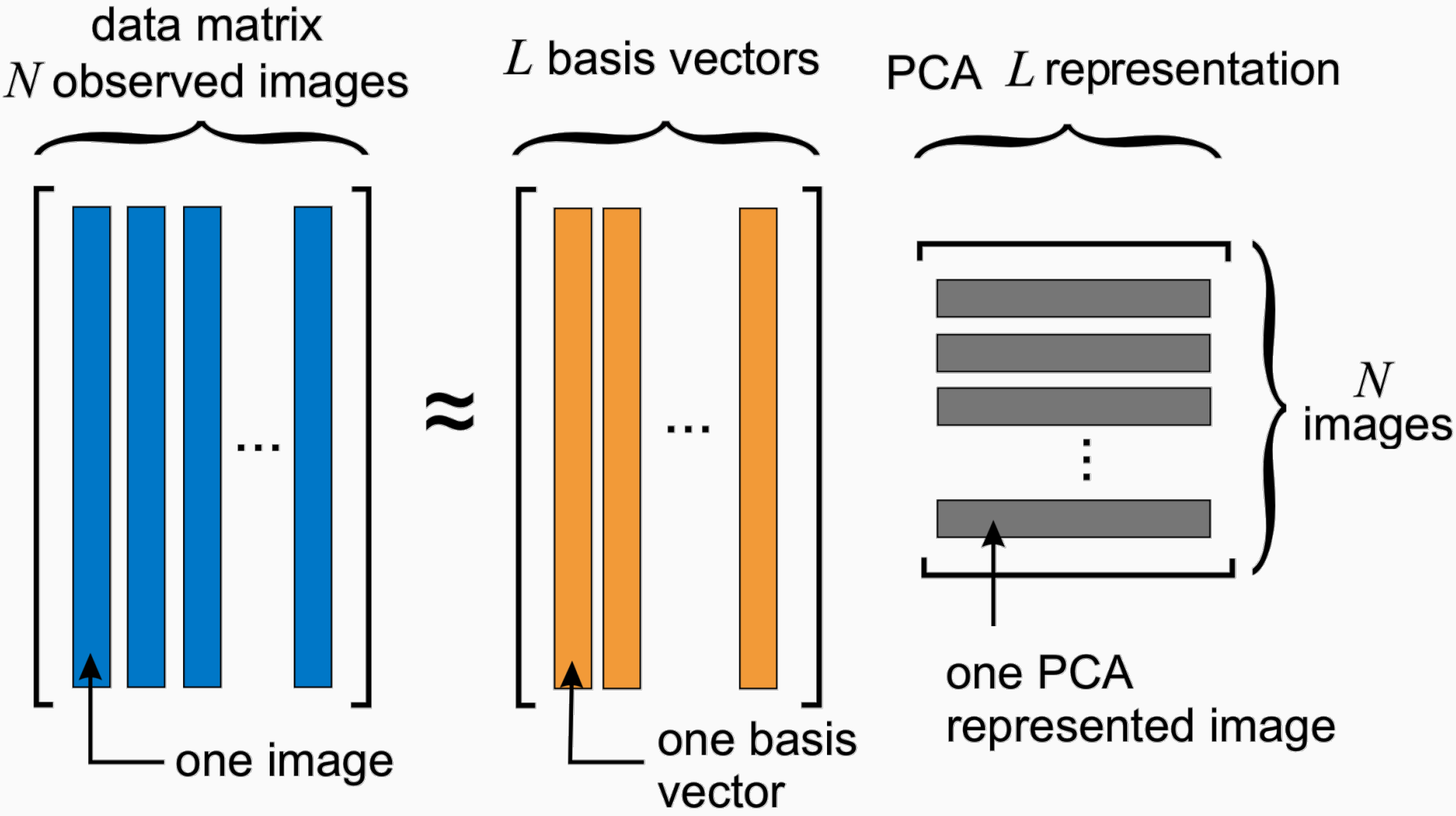
- In statistics, PCA simplifies a multidimensional dataset to lower dimensions by reducing the feature space. In other words, it reduces the dimension of the feature space.
- This dimension reduction is useful for analysis, visualization, and data compression.
- PCA represents data in a new coordinate system whose basis vectors follow the modes of greatest variance (the covariance matrix's eigenvectors).
- Thus, we compute a new basis tailored to the specific dataset.

PCA: Illustration



Each data point (here, an image) is represented in this smaller basis.

PCA: Illustration

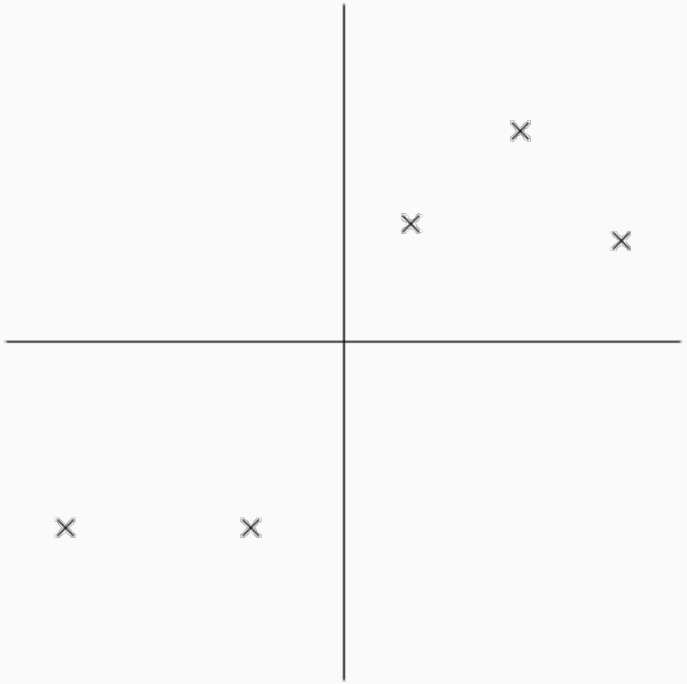


Each data point (here, an image) is represented in this smaller basis.

$\Rightarrow$ represented with fewer coordinates.

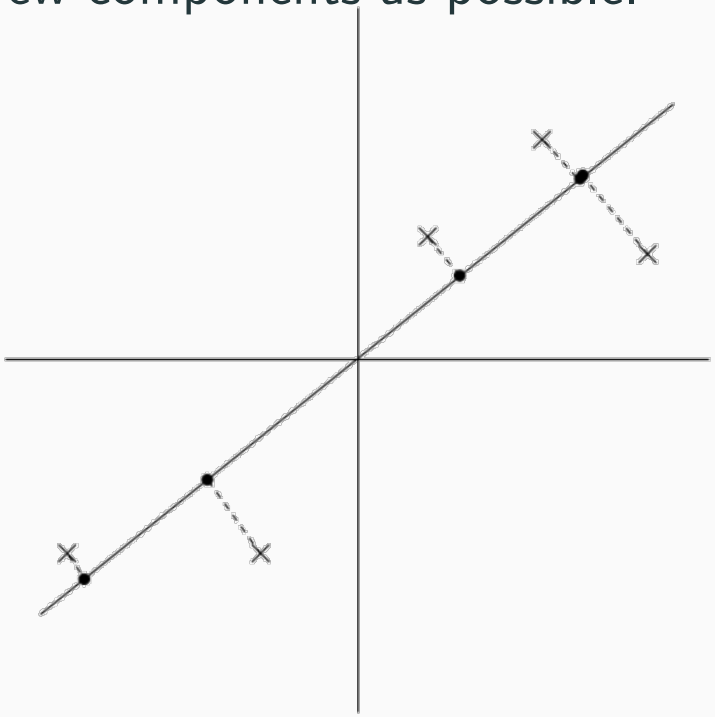
PCA: Motivation

- **Feature redundancy**: Let's suppose we have features casi linearly dependents, such as speed in km/h and mph.
- Can be considered as "repeated information"; we often want to remove redundancy and keep a single representative feature.
- **PCA goal**: find a projection of the data where most variance is captured by as few components as possible.



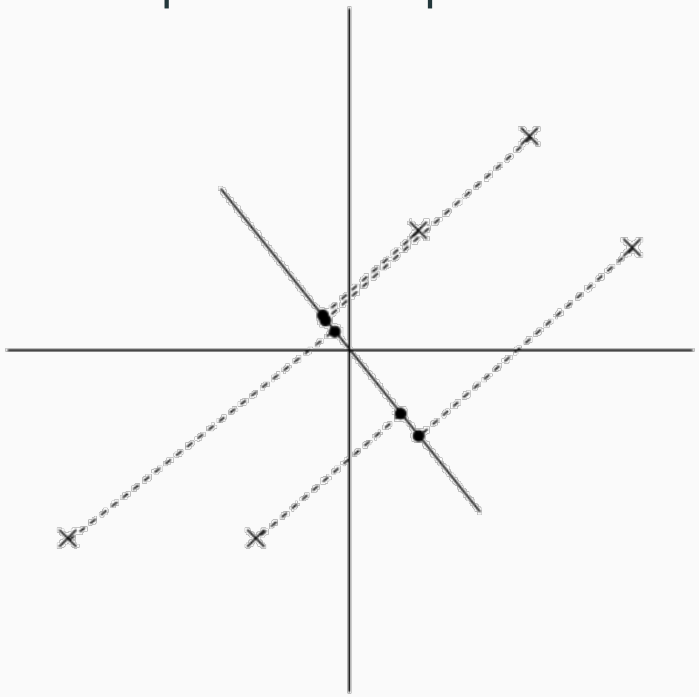
PCA: Motivation

- **Feature redundancy**: Let's suppose we have features casi linearly dependents, such as speed in km/h and mph.
- Can be considered as "repeated information"; we often want to remove redundancy and keep a single representative feature.
- **PCA goal**: find a projection of the data where most variance is captured by as few components as possible.



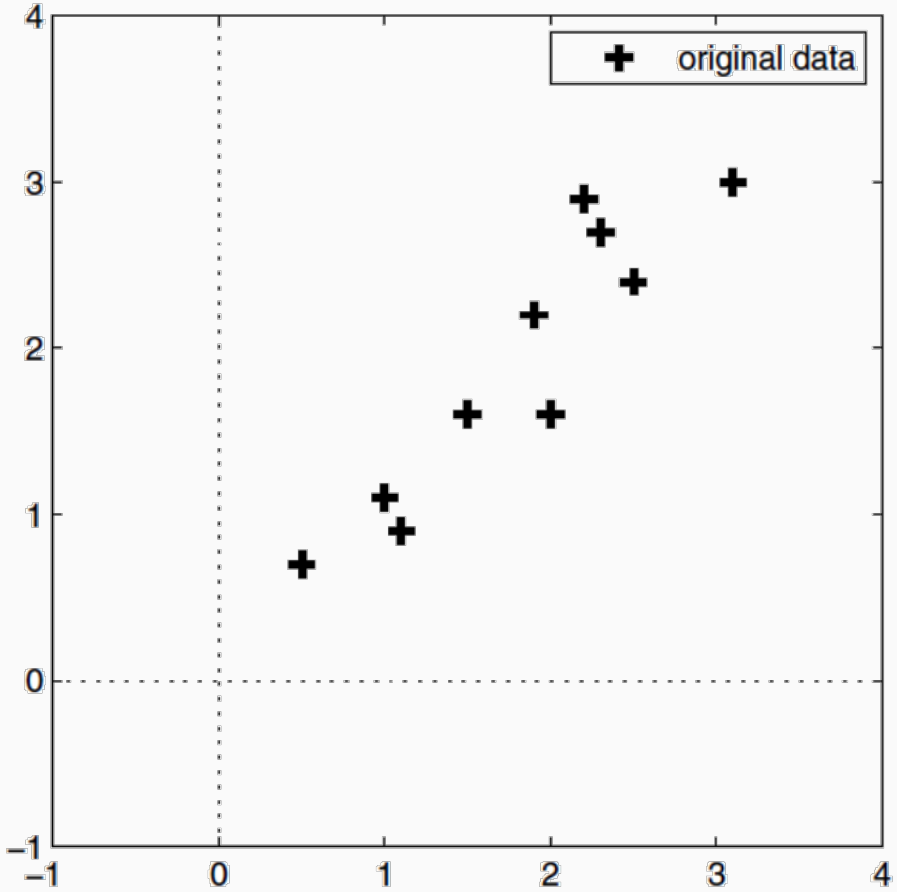
PCA: Motivation

- **Feature redundancy**: Let's suppose we have features casi linearly dependents, such as speed in km/h and mph.
- Can be considered as "repeated information"; we often want to remove redundancy and keep a single representative feature.
- **PCA goal**: find a projection of the data where most variance is captured by as few components as possible.



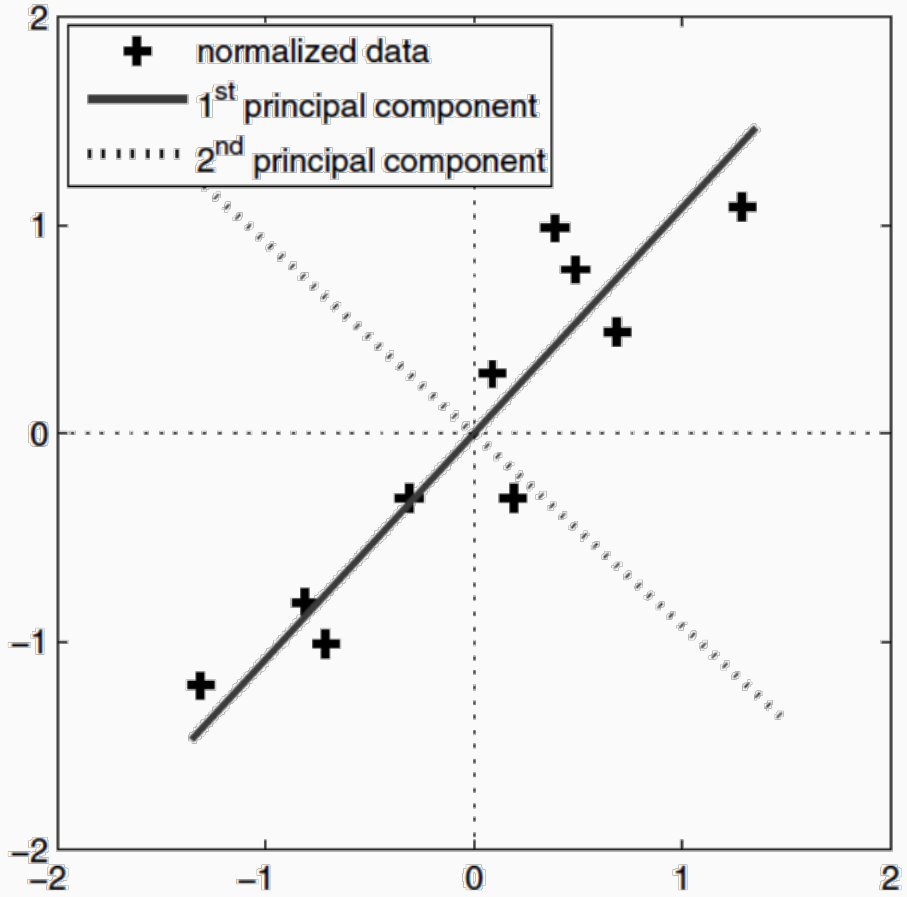
PCA: Visualization of the Transformations

- The original dataset



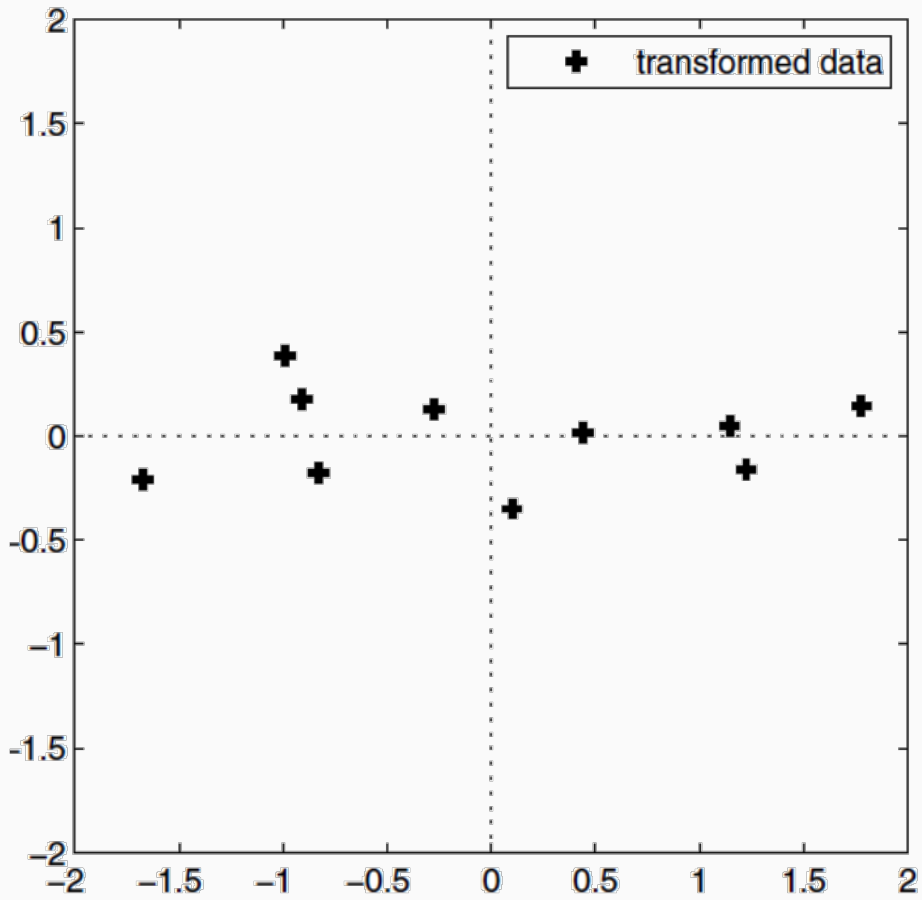
PCA: Visualization of the Transformations

- The original dataset
- The data normalized along the PCA axes



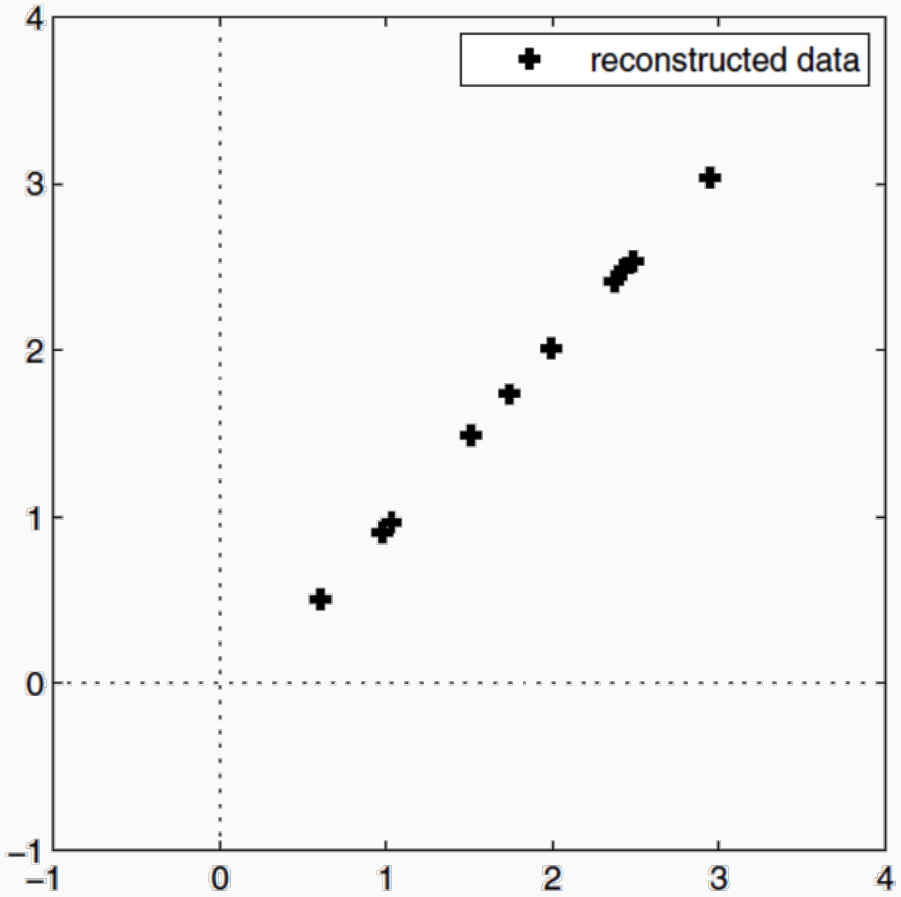
PCA: Visualization of the Transformations

- The original dataset
- The data normalized along the PCA axes
- The data transformed and normalized (in the new axes)



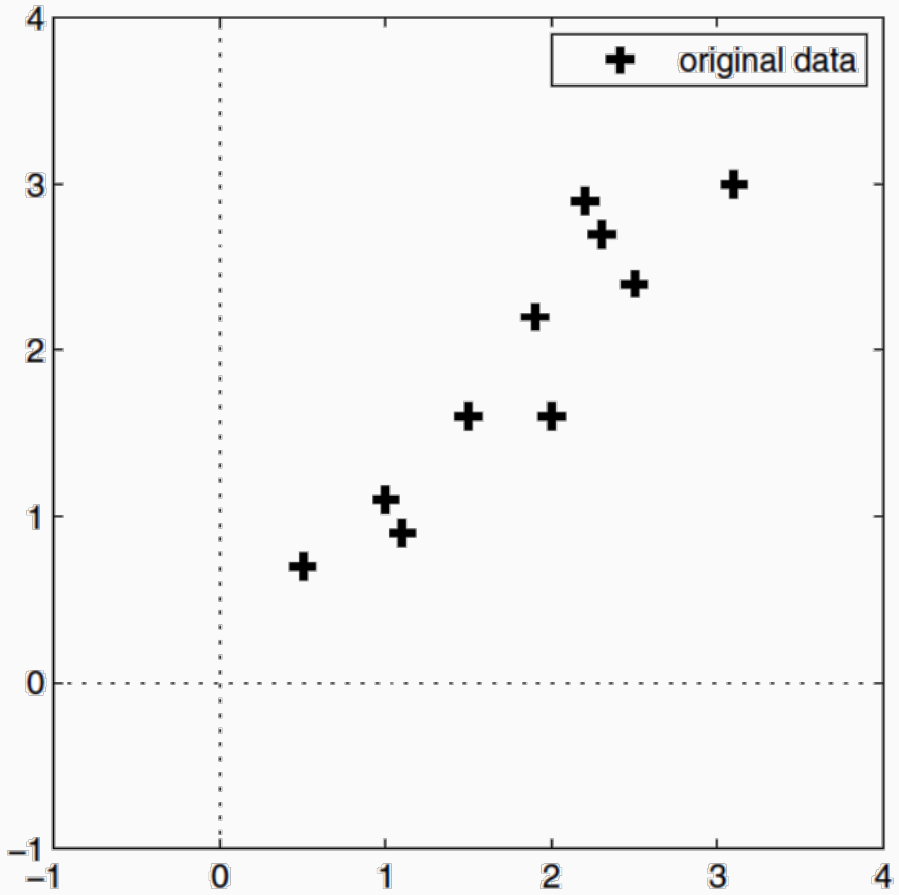
PCA: Visualization of the Transformations

- The original dataset
- The data normalized along the PCA axes
- The data transformed and normalized (in the new axes)
- The data reconstructed by PCA (with information loss)



PCA: Visualization of the Transformations

- The original dataset
- The data normalized along the PCA axes
- The data transformed and normalized (in the new axes)
- The data reconstructed by PCA (with information loss)



Reminder on Eigenvectors and Eigenvalues

- Let A be an $n \times n$ square matrix.
- The eigenvectors are solutions to:

$$A\mathbf{u} = \lambda\mathbf{u},$$

  where λ is the eigenvalue.
- The magnitude of λ reflects its importance in representing the transformation by A.

If $\det(A) \neq 0$, then: $\exists U = \begin{pmatrix} | & | & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_n \\ | & | & | \end{pmatrix}$ and $D = \begin{pmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_d \end{pmatrix}$

such that $A = U\,D\,U^{-1}$, with eigenvectors $\mathbf{u}_i$ and eigenvalues λ_i.

Reminder on Covariance

- **Definition:** The covariance of random variables X and Y is

$$\mathrm{cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])].$$

- **Interpretation:**
  - $\mathrm{cov}(X, Y) > 0$: X and Y tend to increase together (direct relation).
  - $\mathrm{cov}(X, Y) < 0$: one increases as the other decreases (inverse relation).
  - $\mathrm{cov}(X, Y) = 0$: no apparent linear relationship.
- **Covariance Matrix:** For a set of variables, this is the square matrix where each entry is the covariance between a pair.

PCA: Data Approximation

If we approximate data $\mathbf{X} = (\mathbf{X}_i)_i$ by a lower-dimensional subspace of dimension L, the approximation error is minimized by choosing basis vectors $(\mathbf{b}_i)_{i=1..L}$ that solve

$$(\mathbf{b}_i)_{i=1..L} = \arg\max_{\|\mathbf{b}_i\|=1} \sum_{i=1}^{L} \mathbf{b}_i^T \operatorname{cov}(\mathbf{X})\, \mathbf{b}_i.$$

Solution

These are the eigenvectors of $\operatorname{cov}(\mathbf{X})$[1] with the largest eigenvalues.

Intuition: Maximize projected variance

$$\max_{\|u\|=1} \sum_{i=1}^{d} (X_{0,i}^T u)^2 = \max_{\|u\|=1} u^T \Sigma u,$$

which is trivially solved via a Lagrangian!

[1] $\operatorname{cov}(\mathbf{X}) = (\mathbf{X} - \mu_{\mathbf{X}})^T (\mathbf{X} - \mu_{\mathbf{X}})$

PCA: Centering the Data

Given the data matrix $\mathbf{X} = \begin{pmatrix} | & | & | \\ \mathbf{X}_1 & \dots & \mathbf{X}_n \\ | & | & | \end{pmatrix}$:

- To remove biases in the calculus, apply PCA to the centered data $\mathbf{X}_0$.
- Subtract the sample mean from each column of $\mathbf{X}$:

$$\mathbf{X}_0 = \begin{pmatrix} | & | & | \\ \mathbf{X}_1 - \mu_1 & \dots & \mathbf{X}_n - \mu_n \\ | & | & | \end{pmatrix}, \text{ donde } \mu_i = \frac{1}{d}\sum_{k=1}^{d} \mathbf{X}_i^{(k)}$$

Optional variance normalization

$$\mathbf{X} = \begin{pmatrix} | & | & | \\ \frac{\mathbf{X}_1 - \mu_1}{\sigma(\mathbf{X}_1)} & \dots & \frac{\mathbf{X}_n - \mu_n}{\sigma(\mathbf{X}_n)} \\ | & | & | \end{pmatrix}, \text{ donde } \sigma_i^2 = \frac{1}{d}\sum_{k=1}^{d} (\mathbf{X}_i^{(k)} - \mu_i)^2$$

PCA: Principle

- Each data point $\mathbf{X}_i$ of the dataset is a vector in feature space (can be a sound, an image).
- We seek an orthogonal basis whose vectors best represent the data.
  - The data space has a natural orthogonal basis that can be used for this.
  - Each of the data can be expressed in terms of the vectors of this orthogonal basis.
- That basis is the one formed by the eigenvectors of the centered data's covariance matrix $\Sigma = \mathbf{X}_0^T \mathbf{X}_0$.

Eigenvalue

Each of the vector of this new base can be ranked with its propension to represent the majority of the $\mathbf{X}_{0,i}$. This corresponds to its eigenvalue, reflecting how much variance it captures.

PCA: Method

1. Form the data matrix $\mathbf{X} = \begin{pmatrix} | & | & | \\ \mathbf{X}_1 & \dots & \mathbf{X}_n \\ | & | & | \end{pmatrix}$
2. Center the data:

$$\mathbf{X}_0 = \mathbf{X} - \mu, \text{ donde } \mu_i = \frac{1}{d} \sum_{k=1}^{d} \mathbf{X}_i^{(k)}$$

3. Compute covariance $\Sigma = \mathbf{X}_0^T \mathbf{X}_0$.
4. Find eigenvectors u_k and eigenvalues λ_k of Σ.
5. Select the top L eigenvectors such that $\frac{\sum_{k=1}^{L} \lambda_k}{\sum_{k=1}^{n} \lambda_k} > 0.99$, to reconstruct 99% of the data.
6. Get the data representation of the images into $\mathbb{R}^L$.

PCA: Example with 32 Images

PCA: Example

Visualization and Reconstruction

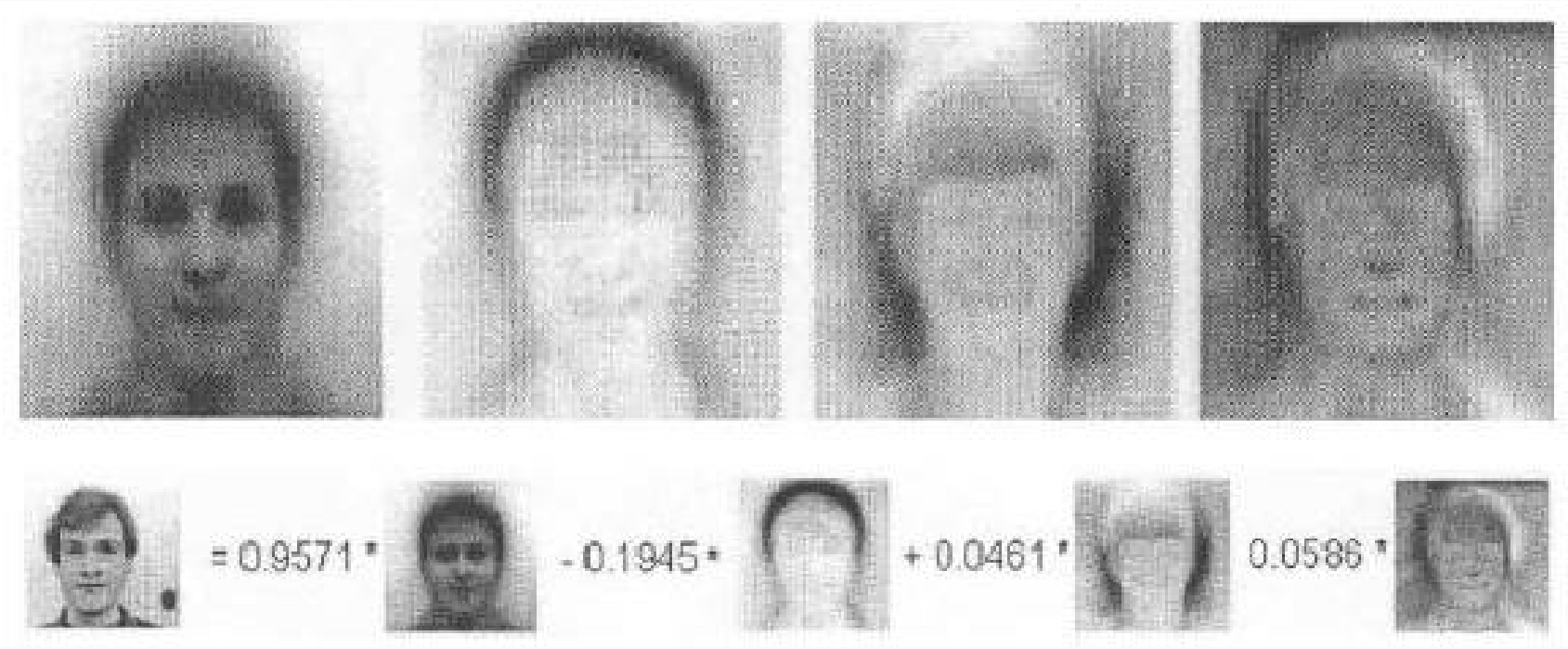
- Reconstruct the image from our 4 basis vectors $\mathbf{u}_1, \ldots, \mathbf{u}_4$.
- The linear combination is:

$$0.078\,\mathbf{u}_1 + 0.062\,\mathbf{u}_2 - 0.182\,\mathbf{u}_3 + 0.179\,\mathbf{u}_4.$$

Reconstruction: 4 Components

Reconstruction: Raw Data

PCA: Another Example



Outline : Other Algorithms

Other Dimensionality Reduction Algorithms

- **Multidimensional Scaling (MDS)**: maps objects to a lower-dimensional space while preserving relative distances.
- **t-SNE (t-Distributed Stochastic Neighbor Embedding)**: maps high-dimensional points to 2D or 3D so that similar points stay close and dissimilar ones stay apart. More explanation here.
- **Independent Component Analysis (ICA)**: separates a signal into statistically independent components.
- **UMAP (Uniform Manifold Approximation and Projection)**: based on Riemannian geometry and algebraic topology. More explanation here.
- **Auto-encoder**: an unsupervised neural network that compresses data (encoder) and reconstructs it (decoder), using the bottleneck layer for dimensionality reduction.

Conclusions

- **Feature selection** (filter or wrapper) reduces model complexity and can improve performance, especially for classifiers sensitive to irrelevant features.
- **Dimensionality reduction** (PCA, auto-encoders, MDS, t-SNE, etc.) consolidates information into fewer dimensions, enhancing visualization and robustness of distance-based methods.
- Always balance the gain in simplicity against the loss of information when removing or projecting features.

Questions?

References i