Please scan this code to register your presence on Ilias
for contact-tracing purposes. Your Ilias identity will be used.
– Only do this if you are *physically present*! –

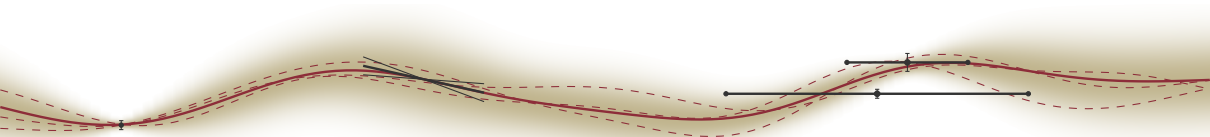
Data Literacy
Lecture 08
Dimensionality Reduction

Philipp Hennig

13 December 2020

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning



Challenges arising from Practical Data

Lecture Outlook

Data Collection How do we *get* data? — Lectures 2 & 3

- avoiding *bias* in the collection of data — Experimental Design
- making sure data is as *informative* as possible — Information

Estimating from Data What does the data (not) tell us? — Lectures 4–9

- Deriving *estimates* of related quantities from observations — Statistical Concepts
- Can we *trust* the estimate? — Variance of Estimators
- Do data/estimates tell us anything noteworthy? — Tests of Significance
- Did we cheat on ourselves? — Properties and Problems of Testing
- Making Predictions from Data — Regression
- Finding Structure in Data — Dimensionality Reduction and Clustering

Fairness and other societal aspects — Lectures 10 & 11

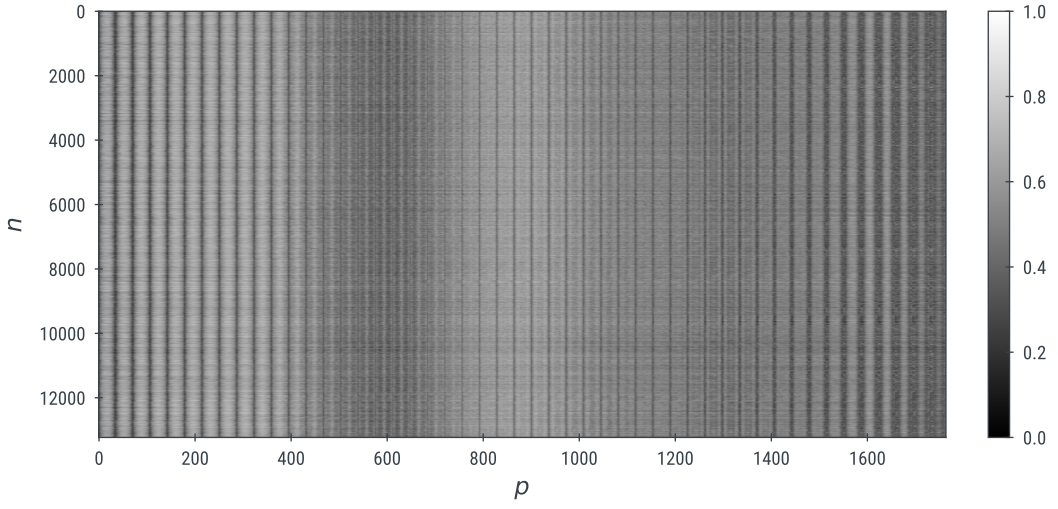
- Does our analysis put certain people at an disadvantage?
- Can/should we do anything about this?

Style and Techniques — Lectures 12-14

- Collecting, documenting and storing data big and small
- Designing good code in the presence of data
- Visualizing and presenting data, analysis and results

Data is Messy

and usually high-dimensional



Consider a dataset $X \in \mathbb{R}^{N \times D}$. **Generative Dimensionality Reduction** aims to find an **encoding** $\phi : \mathbb{R}^D \to \mathbb{R}^M$ and a **decoding** $\psi : \mathbb{R}^M \to \mathbb{R}^D$ with $M \ll D$ such that the encoded representation

$$Z := \phi(X) \in \mathbb{R}^{N \times M}$$

is a *good approximation* of X in the sense that some **reconstruction loss** of $\tilde{X} = \psi(Z)$,

$$\mathcal{L}(X, \psi(Z)) = \mathcal{L}(X, \psi \circ \phi(X))$$

is minimized or small. This may be done, e.g., to

- save memory
- construct a low-dimensional visualization
- "find structure"

Dimensionality reduction does *not* normally use known class labels $Y \in \mathbb{R}^{N \times C}$. This is what supervised methods (remember logistic regression!) is for.

Principal Component Analysis (PCA)

Linear dimensionality reduction — Beltrami, 1873, Jordan, 1874, Pearson, 1901, Schmidt, 1907, Hotelling, 1933, Lanczos, 1950

Consider Data $X \in \mathbb{R}^{N \times D} = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_N]$. We are looking for "the best" basis $U = \{\boldsymbol{u}_i\}_{i=1,\ldots,D}$, $\boldsymbol{u}_i^\intercal \boldsymbol{u}_j = \delta_{ij}$ to construct an M-dimensional approximation to

$$\boldsymbol{x}_n = \sum_{i=1}^{D} (\boldsymbol{x}_n^\intercal \boldsymbol{u}_i)\boldsymbol{u}_i \qquad \text{i.e. } X = (XU)U^\intercal \qquad \text{as} \qquad \tilde{\boldsymbol{x}}_n := \sum_{i=1}^{M} a_{ni}\boldsymbol{u}_i + \sum_{i=M+1}^{D} b_i \boldsymbol{u}_i$$

To define "best", decide to minimize the *square empirical risk*

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N}\sum_{j=1}^{D}\left[\boldsymbol{x}_n - \sum_{i=1}^{M} a_{ni}\boldsymbol{u}_i - \sum_{i=M+1}^{D} b_i \boldsymbol{u}_i\right]_j^2$$

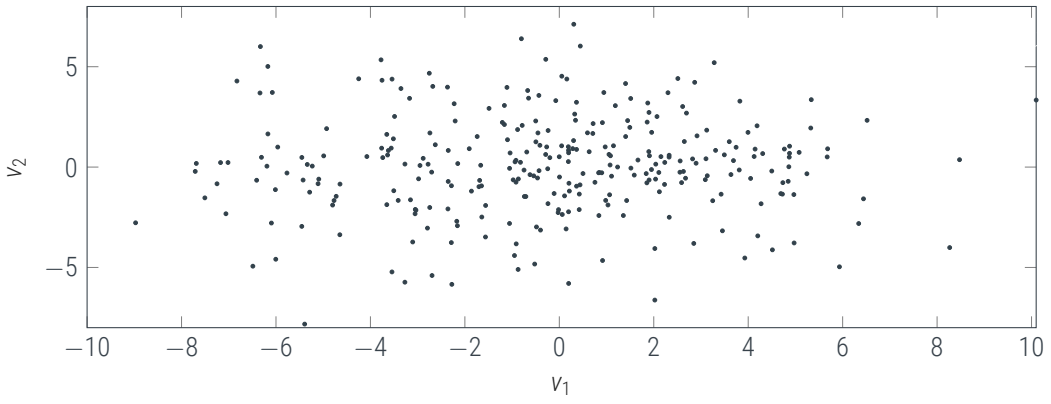
This yields (homework) $a_{ni} = \boldsymbol{x}_n^\intercal \boldsymbol{u}_i$, and $b_i = \bar{\boldsymbol{x}}^\intercal \boldsymbol{u}_i$. (So $Z = \phi(X) = XU_{:M} + \frac{1}{N}\mathbf{1}\mathbf{1}^\intercal X U_{M+1:}$) and

$$J = \frac{1}{N}\sum_{i=M+1}^{D}\sum_{n=1}^{N} u_i^\intercal (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^\intercal \boldsymbol{u}_i = \sum_{i=M+1}^{D} u_i^\intercal S u_i \text{ with } S := \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^\intercal$$

S is the *sample covariance matrix*. Hence, choose U as the eigenvectors of U. If we first center the data `Xhat = X - X.mean()`, so $\boldsymbol{b} = 0$, the U are the right **singular vectors** of $\hat{X} = Q\Sigma U^\intercal$.

Low-Dimensional Representation

PCA on the mystery dataset

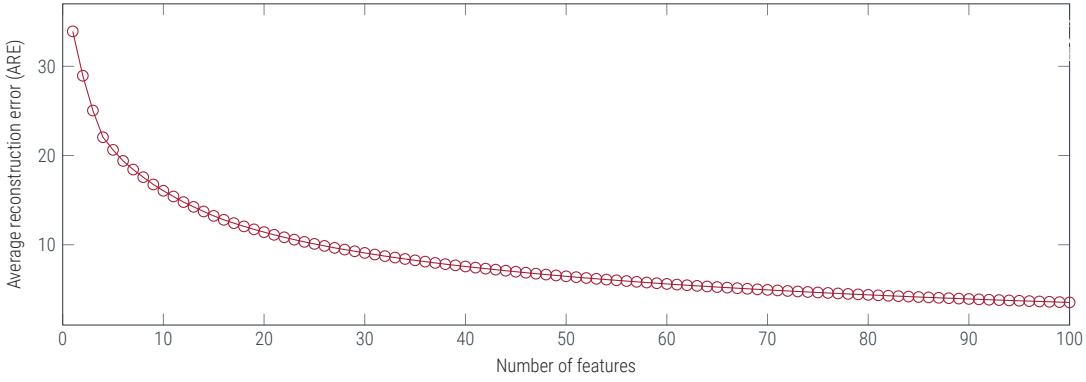


PCA does *not* necessarily reveal “structure” in the data.

But it *does* compress the data

i.e. what it is designed to do

$$\text{ARE} = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2 \qquad \text{(data has } D = 1764\text{)}$$



Some Datasets are easy on our eyes

Labeled Faces in the Wild (LFW)

Huang et al., 2009. Pre-processed by Samira Samadi (more in next lecture)

$N = 5924$ faces at $49 \times 36 = 1764$ pixels, gray scale (pre-processed from full dataset)

Compressing Faces

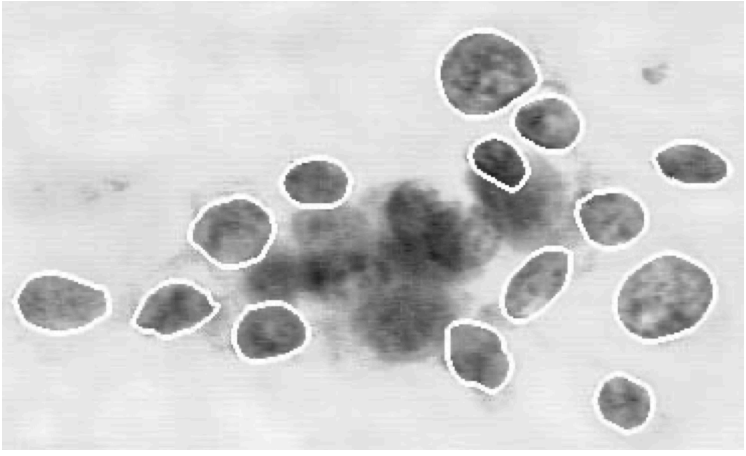
PCA on *Labeled Faces in the Wild*

Principal Component Analysis (PCA) constructs the low-dimensional *linear projection* of the data that *minimizes* the ℓ_2 reconstruction error.

Another Dataset

The Wisconsin Breastcancer Dataset

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

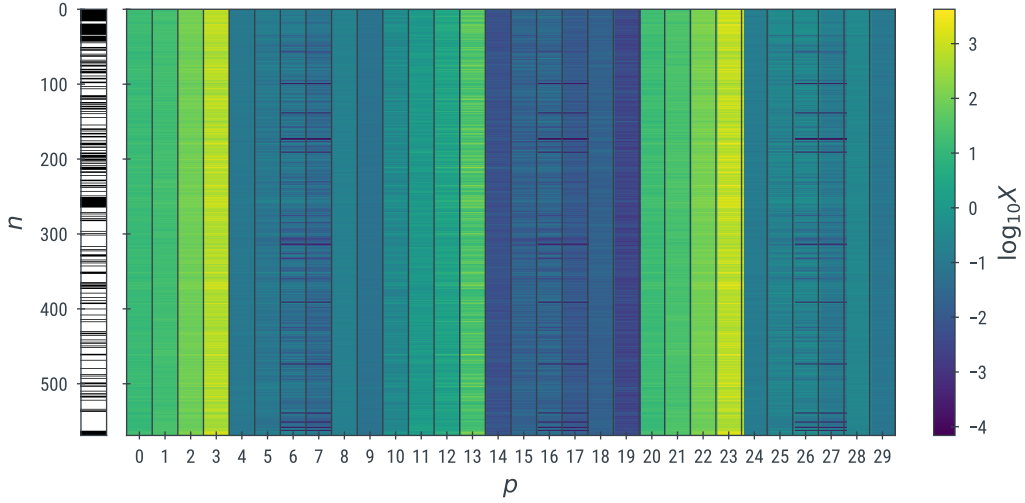


Street, Wolberg, Mangasarian. *Nuclear Feature Extraction For Breast Tumor Diagnosis*.

Proc. SPIE Int. Symp. on Electronic Imaging, vol. **1905**, pp. 861–870. 1992

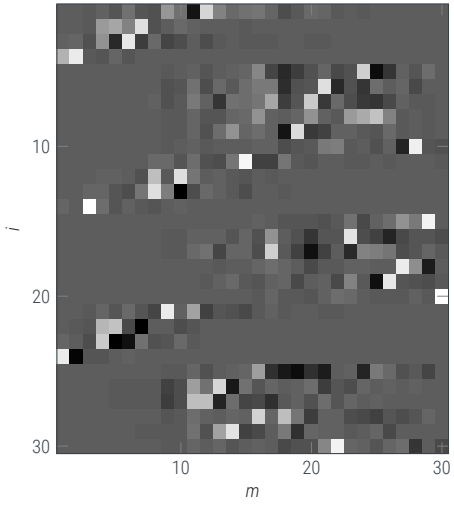
The Data

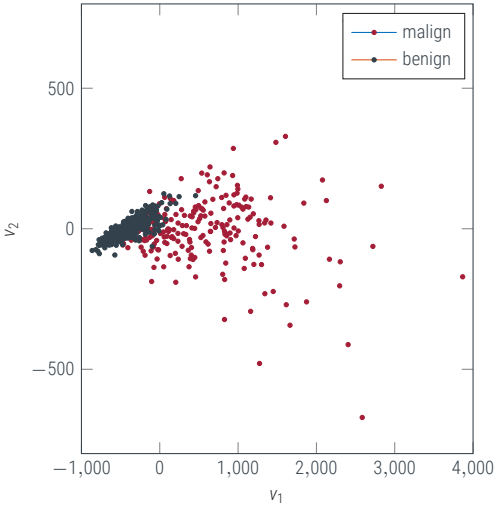
not much to see here



Visualized

First two principal components





The Meta Data

for the Wisconsin Breast Cancer Dataset

```
Attribute Information:
1) ID number 2) Diagnosis (M = malignant, B = benign)
3-32) Ten real-valued features are computed for each cell nucleus:
    1) radius (mean of distances from center to points on the perimeter)
    2) texture (standard deviation of gray-scale values)
    3) perimeter
    4) area
    5) smoothness (local variation in radius lengths)
    6) compactness (perimeter^2 / area - 1.0)
    7) concavity (severity of concave portions of the contour)
    8) concave points (number of concave portions of the contour)
    9) symmetry
    10) fractal dimension ("coastline approximation" - 1)
The mean, standard error and "worst" or largest (mean of the three largest values) of these
features were computed for each image, resulting in 30 features.
For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.
```

All Data has Units!

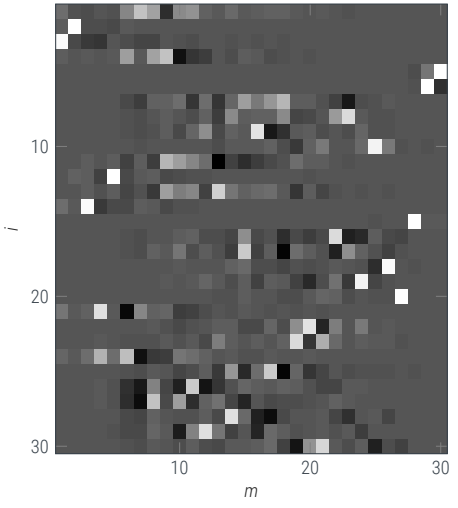
even if they are not recorded in the metadata!

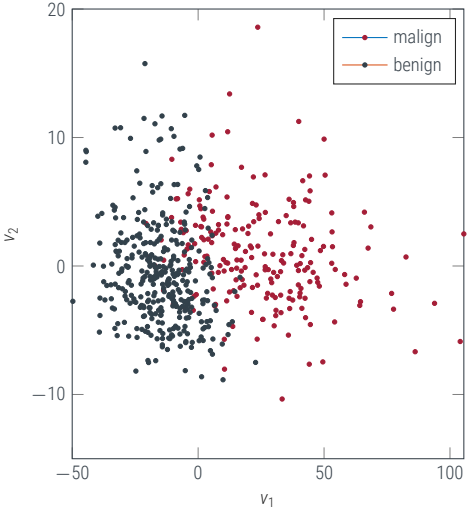
1.	radius (mean of distances from center to points on the perimeter)	[cm]
2.	texture (standard deviation of gray-scale values)	$\in [0, 255]$
3.	perimeter	[cm?]
4.	area	[cm^2]
5.	smoothness (local variation in radius lengths)	[cm]
6.	compactness (perimeter^2 / area - 1.0)	$\in [1, \infty]$
7.	concavity (severity of concave portions of the contour)	[???]
8.	concave points (number of concave portions of the contour)	[???]
9.	symmetry	[???]
10.	fractal dimension (“coastline approximation” - 1)	[???]

Note that $X \twoheadleftarrow X \cdot \text{diag}(\boldsymbol{s})$ changes the singular vectors of X!

PCA, in other units

rescaled $d_2/255$, $d_5/10$, $d_8/200$





Principal Component Analysis

- amounts to *projecting* the (centered) data X onto its first M right **singular vectors**

$$\hat{X} = X - \mathbf{1}\bar{x}^\intercal, \qquad \hat{X} = Q\Sigma U^\intercal = \mathrm{svd}(\hat{X}), \qquad \tilde{X} = (\hat{X}U_{1:M})U_{1:M}^\intercal$$

- PCA is **meaningless** if the individual data dimensions do not all have the same units.

Before reducing the dimensionality of your data, think about *why* you want to do so?

- Are you hoping to see structure?
- Do you want to save memory?
- Are there classes in the data, and you want to know how they differ?

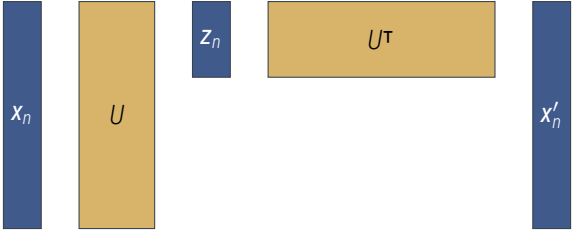
Nonlinear Dimensionality Reduction

beyond PCA

- PCA minimizes the **square (ℓ_2) reconstruction error** of a **linear** projection $\boldsymbol{z}_n = \boldsymbol{x}_n^\intercal U$

$$J(U) = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \boldsymbol{x}_n UU^\intercal\|^2 = \frac{1}{N}\sum_{n=1}^{N} \ell_2(\boldsymbol{x}_n, \psi(\phi(\boldsymbol{x}_n)))$$

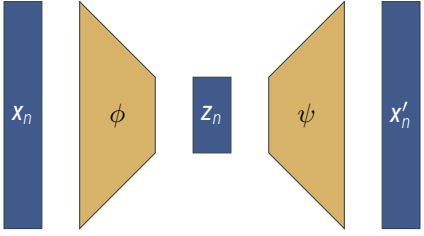
with *encoder* $\phi(\boldsymbol{x}) = U^\intercal(\boldsymbol{x} - \bar{\boldsymbol{x}})^\intercal$ and *decoder* $\psi(\boldsymbol{z}) = U\boldsymbol{z} + \bar{\boldsymbol{x}} = (\boldsymbol{x}')^\intercal$



- PCA minimizes the **square (ℓ_2) reconstruction error** of a **linear** projection $\boldsymbol{z}_n = \boldsymbol{x}_n^\intercal U$

$$J(U) = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \tilde{\boldsymbol{x}}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \boldsymbol{x}_n UU^\intercal\|^2 = \frac{1}{N}\sum_{n=1}^{N} \ell_2(\boldsymbol{x}_n, \psi(\phi(\boldsymbol{x}_n)))$$

with *encoder* $\phi(\boldsymbol{x}) = U^\intercal(\boldsymbol{x} - \bar{\boldsymbol{x}})^\intercal$ and *decoder* $\psi(\boldsymbol{z}) = U\boldsymbol{z} + \bar{\boldsymbol{x}} = (\boldsymbol{x}')^\intercal$



- Exchange the **linear** projections by **nonlinear** ones to get an **autoencoder**.

Nonlinear Dimensionality Reduction

beyond PCA

- PCA minimizes the **square (ℓ_2) reconstruction error** of a **linear** projection $z_n = x_n^\intercal U$

$$J(U) = \frac{1}{N}\sum_{n=1}^{N} \|x_n - \tilde{x}_n\|^2 = \frac{1}{N}\sum_{n=1}^{N} \|x_n - x_n UU^\intercal\|^2 = \frac{1}{N}\sum_{n=1}^{N} \ell_2(x_n, \psi(\phi(x_n)))$$

  with *encoder* $\phi(x) = U^\intercal x^\intercal$ and *decoder* $\psi(z) = Uz = (x')^\intercal$

- Exchange the **linear** projections by **nonlinear** ones to get an **autoencoder**.
- replace the ℓ_2 with the log loss (KL-divergence) to get a *variational autoencoder* (VAE)

$$J(\phi, \psi) = D_{\mathrm{KL}}(q_\phi(z \mid x) \| p_\psi(z \| x)) = \frac{1}{N}\sum_{n=1}^{N} \log q_\phi(z \mid x) - \log p_\psi(z \mid x)$$

- this requires parametrization choices for q_ϕ, p_ψ. Typically Gaussian.

This *can* help with the problems above, but only if you know what you're doing!

Generative Dimensionality Reduction

- amounts to *projecting* the data X onto a low-dimensional *latent representation* $Z = \phi(X)$ such that the *reconstruction*

$$X' = \psi(Z) = (\psi \circ \phi)X$$

  "captures structure" of X, as described by some loss function J
- The simplest case is PCA, where ϕ, ψ are linear and J is quadratic.
- Note: "Capturing structure" does not automatically do what you may want
  - the latent representation may not be a good visualization
  - if there is structure the loss does not capture, it will not generally be preserved by the encoding

But there are also *non-generative* ways to reduce dimensionality. For example, we may search for a low-dimensional encoding that preserves "structure" of the high-dimensional data, but without requiring an invertible map.

Embedding without Encoding

neighborhood embedding

G. Hinton & S. Roweis. *Stochastic neighbor embedding*. NeurIPS 2002

- Instead of explicitly constructing the encoding/decoding ϕ, ψ, define *affinity* graph / neighborhood probabilities p_{ij} and q_{ij} on input $X = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^{n \times d}$ and visualization space $Y = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n] \in \mathbb{R}^{n \times 2}$
- tune $\boldsymbol{y}$ (thus q_{ij}) to minimize some divergence, e.g. $D_{\mathrm{KL}}(p \| q)$

Some immediate observations

- this means the embedded representation $\boldsymbol{y}$ can not be interpreted as an encoding. There are no explicit maps $\phi : \boldsymbol{x} \mapsto \boldsymbol{y}$ or $\psi : \boldsymbol{y} \mapsto \boldsymbol{x}$.
- building blocks: affinity p, affinity q, divergence, optimizer. All of which come with assumptions and caveats.

Sam Roweis

1972–2010

- affinity p as 'probability for i to pick j as neighbor if picked by isotropic Gaussian, symmetrised'

$$p_{j|i} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik})} \quad \text{with} \quad d_{ij}^2 = \frac{\|x_i - x_j\|^2}{2\sigma_i^2} \quad \text{and set} \quad p_{ij} = \frac{1}{2n}(p_{j|i} + p_{i|j}), p_{ii} = 0$$

- set hyperparameters σ_i to achieve ad-hoc *perplexity* $2^{\mathbb{H}(p_i)} = 2^{-\sum_j p_{ij} \log_2 p_{ij}}$
- affinity q as Cauchy distribution ('Student-t distribution with one degree of freedom')

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_k \sum_{\ell \neq k} (1 + \|y_k - y_\ell\|^2)^{-1}} \quad \text{and} \quad q_{ii} = 0$$

- divergence: Kullback-Leibler $D_{\mathrm{KL}}(p\|q) = \sum_{ij, i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$
- optimization: random initialization, momentum stochastic gradient descent

$$\frac{\partial D_{KL}(p\|q)}{\partial y_i} = 4 \sum_{j \neq i} \frac{(p_{ij} - q_{ij})(y_i - y_j)}{1 + \|y_i - y_j\|^2}$$

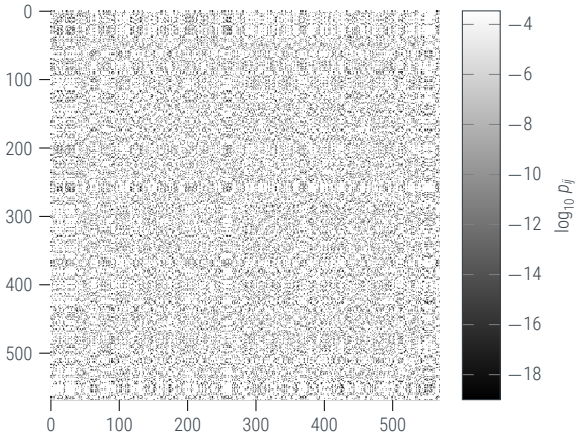
Affinity p

WBC dataset

$$p_{j|i} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik})}$$

where $$d_{ij}^2 = \frac{\|x_i - x_j\|^2}{2\sigma_i^2}$$

and set $$p_{ij} = \frac{1}{2n}(p_{j|i} + p_{i|j}),$$
$$p_{ii} = 0$$



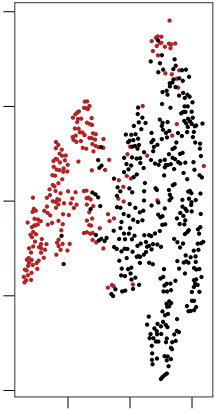
What do your dimensions mean?

re-scaled inputs

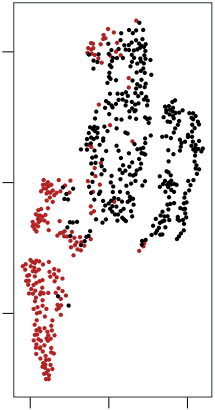
- Remember that our (WBC) dataset has 30 dimensions of differing meaning
- in particular, $X[:, :10]$ contains means, $X[:, 10:20]$ contains variances, $X[:, 20:30]$ maxima

```
X2 = np.hstack([X[:, :10],np.sqrt(X[:, 10:20]),X[:, 20:30]])
```

perplexity 25



perplexity 25, transformed



Summary *t*-stochastic neighborhood embedding (*t*-SNE)

- an unsupervised way to make low-dimensional representations of high-dimensional data that retain as much *local* similarity as possible
- lots of free parameters, assumptions and algorithmic tricks needed. Not a *black box* like PCA
- SNE is not a formal way to find scientific "insights", but a way to explore a dataset and find leads to interesting properties

Powerful tools can be dangerous or useless if you do not understand them.

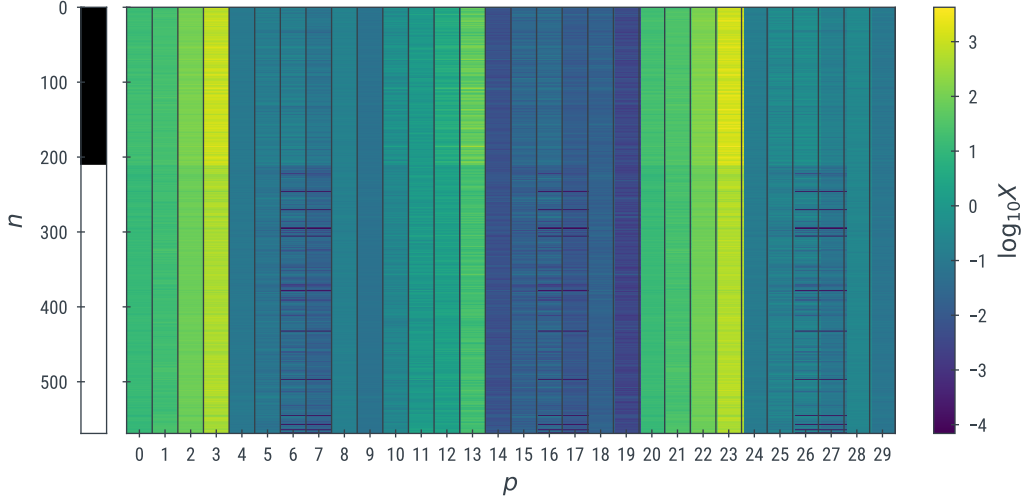
So far, we have ignored that our dataset is labeled.
Can we use the labels to learn a *supervised* embedding?

Can we separate the two classes?

What can we do to *separate* the classes from each other? To **discriminate** among them?

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

WBC dataset, sorted by class label

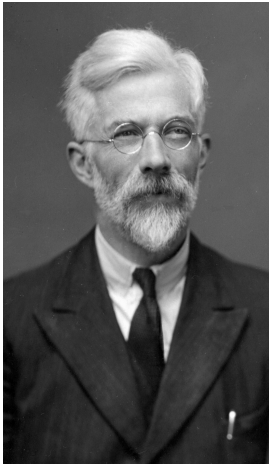


Can we separate the two classes?

Latent Discriminant Analysis

R.A. Fisher, *The Use of Multiple Measurements in Taxonomic Problems.* Annals of Eugenics, 1936

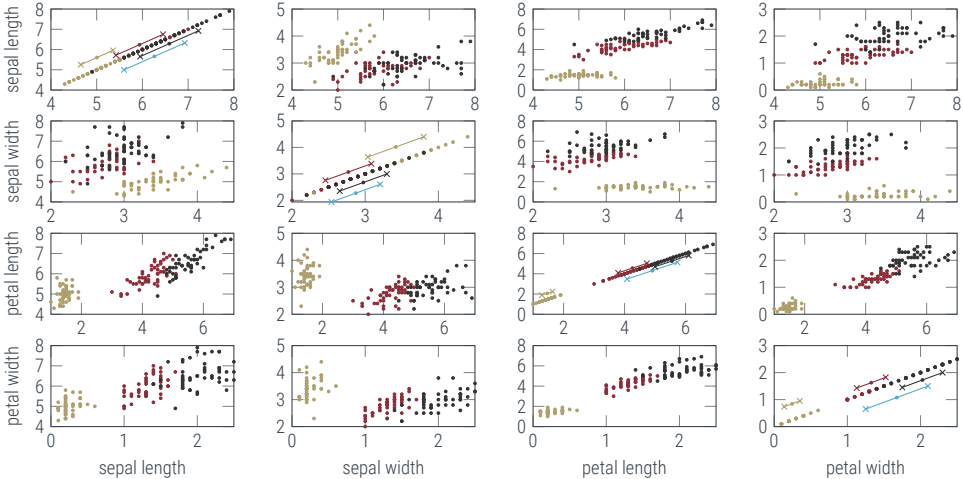
Iris setosa				*Iris versicolor*				*Iris virginica*			
Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
5·1	3·5	1·4	0·2	7·0	3·2	4·7	1·4	6·3	3·3	6·0	2·5
4·9	3·0	1·4	0·2	6·4	3·2	4·5	1·5	5·8	2·7	5·1	1·9
4·7	3·2	1·3	0·2	6·9	3·1	4·9	1·5	7·1	3·0	5·9	2·1
4·6	3·1	1·5	0·2	5·5	2·3	4·0	1·3	6·3	2·9	5·6	1·8
5·0	3·6	1·4	0·2	6·5	2·8	4·6	1·5	6·5	3·0	5·8	2·2
5·4	3·9	1·7	0·4	5·7	2·8	4·5	1·3	7·6	3·0	6·6	2·1
4·6	3·4	1·4	0·3	6·3	3·3	4·7	1·6	4·9	2·5	4·5	1·7
5·0	3·4	1·5	0·2	4·9	2·4	3·3	1·0	7·3	2·9	6·3	1·8



Ronald A. Fisher
1890–1962

Which projection discriminates best?

The Iris dataset



● *versicolor* ● *setosa* ● *virginica* ● virginica ∪ versicolor

Consider Data $\boldsymbol{x}_n \in \mathbb{R}^D$, $n = 1, \ldots, N$ from two classes $\mathcal{C}_1, \mathcal{C}_2$ (i.e. index sets with $\mathcal{C}_1 \cup \mathcal{C}_2 = [1, \ldots, N]$ and $\mathcal{C}_1 \cap \mathcal{C}_2 = \varnothing$) with class means and un-normalised covariances

$$\boldsymbol{m}_i := \frac{1}{N_i} \sum_{n \in \mathcal{C}_i} \boldsymbol{x}_n \qquad S_i := \sum_{n \in \mathcal{C}_i} (\boldsymbol{x}_n - \boldsymbol{m}_i)(\boldsymbol{x}_n - \boldsymbol{m}_i)^\intercal \qquad \text{where } N_i = |\mathcal{C}_i|.$$

Try to find a projection $\boldsymbol{w}$ such that the scalar variables $y_i = \boldsymbol{w}^\intercal \boldsymbol{x}_i$ maximize the ratio between inter-class variance $(m_2 - m_1)^2 = (\boldsymbol{w}^\intercal(\boldsymbol{m}_2 - \boldsymbol{m}_1))^2$ and in-class square distances

$$s_i^2 = \sum_{n \in \mathcal{C}_i} (y_i - m_i)^2 = \sum_{n \in \mathcal{C}_i} \boldsymbol{w}^\intercal (\boldsymbol{x}_n - \boldsymbol{m}_i)(\boldsymbol{x}_n - \boldsymbol{m}_i)^\intercal \boldsymbol{w} = \boldsymbol{w}^\intercal S_i \boldsymbol{w}.$$

That is, maximise

$$J(\boldsymbol{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

Maximise (note we can scale $\boldsymbol{w}$ arbitrarily, so only its direction matters)

$$J(\boldsymbol{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\boldsymbol{w}^\intercal S_B \boldsymbol{w}}{\boldsymbol{w}^\intercal S_W \boldsymbol{w}} = \frac{\boldsymbol{w}^\intercal \overbrace{(\boldsymbol{m}_2 - \boldsymbol{m}_1)(\boldsymbol{m}_2 - \boldsymbol{m}_1)^\intercal}^{=:S_B} \boldsymbol{w}}{\boldsymbol{w}^\intercal \underbrace{\left(\sum_{n \in \mathcal{C}_1} (\boldsymbol{x}_n - \boldsymbol{m}_1)(\boldsymbol{x}_n - \boldsymbol{m}_1)^\intercal + \sum_{n \in \mathcal{C}_2} (\boldsymbol{x}_n - \boldsymbol{m}_2)(\boldsymbol{x}_n - \boldsymbol{m}_2)^\intercal \right)}_{S_W} \boldsymbol{w}}$$

$$\frac{\partial J}{\partial \boldsymbol{w}} = 2\frac{S_B \boldsymbol{w}}{\boldsymbol{w}^\intercal S_W \boldsymbol{w}} - 2\frac{\boldsymbol{w}^\intercal S_B \boldsymbol{w}}{(\boldsymbol{w}^\intercal S_W \boldsymbol{w})^2} S_w \boldsymbol{w} = C_1 \cdot (\boldsymbol{m}_2 - \boldsymbol{m}_1) - C_2 \cdot S_w \boldsymbol{w}$$

$$\boldsymbol{w} \propto S_W^{-1}(\boldsymbol{m}_2 - \boldsymbol{m}_1)$$

When should we predict class 1 over 2? The (m_i, S_i) define *generative* Gaussian distributions $p(y_n \mid c = i) = \mathcal{N}(y_n; m_i, s_i)$, and the marginal frequencies give a "prior" class probability $p(c_i) = N_i/N$.

Exercise: Find the value of the *best discriminator* y_0 set by considering *posterior log odds*

$$r(y) = \log \frac{p(c_1 \mid y)}{p(c_2 \mid y)} = \log \frac{p(y \mid c_1)p(c_1)}{p(y \mid c_2)p(c_2)} = \log \frac{\mathcal{N}(y; m_1, s_1^2)N_1}{\mathcal{N}(y; m_2, s_2^2)N_2}$$

and solving for $r(y_0) = 0$.

LDA for the Iris dataset

Directly from Fisher's paper

R.A. Fisher, *The Use of Multiple Measurements in Taxonomic Problems.* Annals of Eugenics, 1936

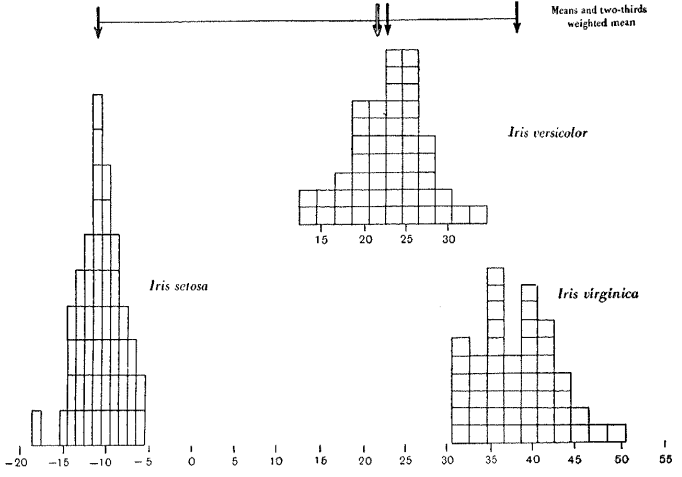


Fig. 1. Frequency histograms of the discriminating linear function, for three species of *Iris*.

Summary: Dimensionality Reducition

- There are different motivations for dimensionality reduction
  - encoding/decoding paradigm (save memory)
  - neighborhood embedding (preserve similarity)
  - discriminiation (separate classes from each other)
- which one to choose depends on your application. But you should know *why* you are using one technique over the other, because each dimensionality reduction creates some artifacts

Please provide feedback:

A date for your calendar?

Methods of Machine Learning Research Seminar

Wednesday, 24 November, 15:00 (st)
Sarah Bechtle (MPI IS / NYU)

Lifelong learning in the real world

I'm going to present work centred around the fundamental research question of how a robot can learn in the real world. The robot gathers the data it needs to learn through interactions with - and perception of - the environment. In pursuing answering this research question, I work on two interconnected directions: 1) Model Based Learning in the Action Perception Loop, where a model of the environment and a policy are learned iteratively from data collected on the robot and 2) Learning to Learn in the Action Perception Loop, where the robot learns how to learn a task. These directions fundamentally are concerned with learning representations that generalize quickly to new tasks and scenarios, which is a key capability that allows us humans to continuously learn over a lifetime.

details at https://talks.tue.ai/talks/talk/id=50