WARWICK
THE UNIVERSITY OF WARWICK

Multivariate Analysis
PCA and other dimensionality reduction approaches

Dr. Fayyaz Minhas

09 Feb 2023

Department of Computer Science

University of Warwick

(slides are text-heavy to provide self-contained notes for students)

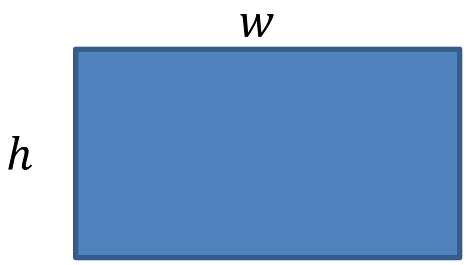
Preliminaries

- Lagrange Multipliers

- What is the relationship between the following?
  - Vectors
  - Matrices
  - Distance
  - Dot Product
  - Projections
  - Transformations
  - Eigen Vectors
  - Variance
  - Variance and Information
  - Correlation
  - Covariance
  - Covariance Matrix
  - Covariance and Information
- Principal Component Analysis (PCA)

Old MacDonald Had a Farm …

Old MacDonald had a plan, E-I-E-I-O,
With a length L of wire in his hand, E-I-E-I-O,
He dreamt of barns with spacious land,
Largest area, that was his grand stand.
"Maximize A equals h times w," he cried, E-I-E-I-O,
For the largest barn, he'd not be denied, E-I-E-I-O,
With a width w and length h, he made it so,
The grandest barn began to show, E-I-E-I-O!

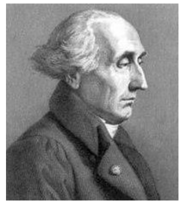


Constrained Optimization Problem (COP)

$$\max_{h,w} A = hw$$

Such that:

$$2(h+w) = L$$



Given a COP
Express each constraint as g(x)=0
And add an additional variable α called a LaGrange multiplier
Change the objective function to include the term αg(x)
Solve the resulting UCOP

https://en.wikipedia.org/wiki/Lagrange_multiplier

Unconstrained Optimization Problem (UCOP)

$$\max_{h,w,\alpha} D = hw - \alpha(2(h+w) - L)$$

$$\frac{\delta D}{\delta h} = w - 2\alpha = 0$$

$$\frac{\delta D}{\delta w} = h - 2\alpha = 0$$

$$\frac{\delta D}{\delta \alpha} = 2(h+w) - L = 0$$

$$h = w = L/4$$

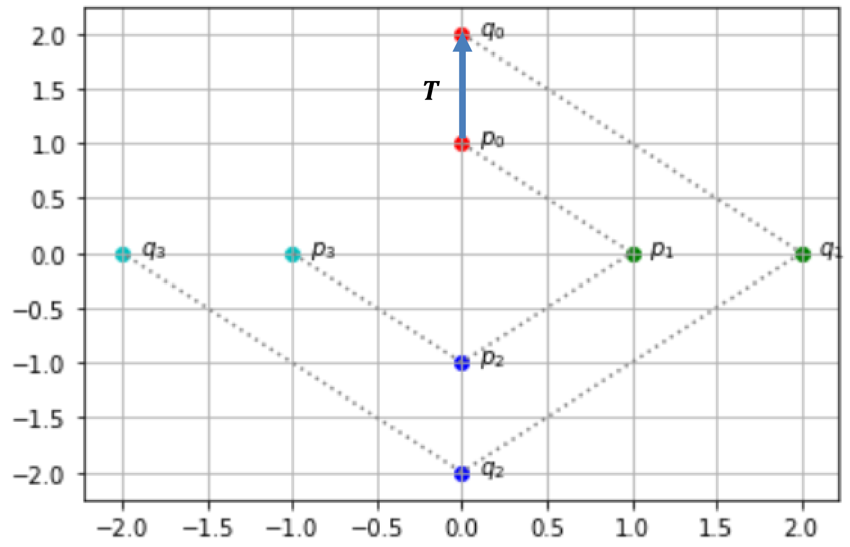
Operations on Vectors

- Using matrices
  - One way of thinking about matrices is that they are collection of vectors
  - For example, we can represent the data set for a given problem as a data matrix
    - Each row is a vector representation of a single example or data point
- Matrices as operators

Multiplication of a vector by a matrix

– Multiplication of a vector with a matrix can be viewed as a geometric transformation of the vector

$$T = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, x = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$y = Tx = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$



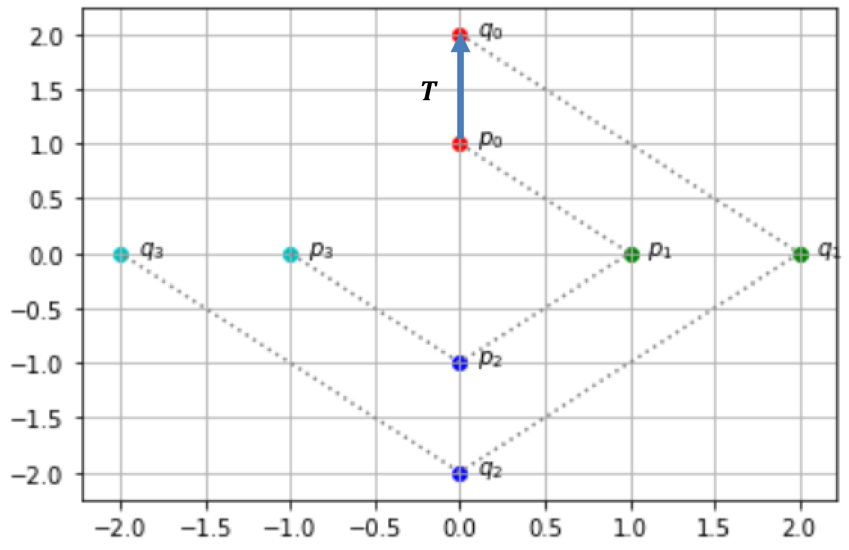
Eigen Vectors

- Those points that are characteristic to a given matrix that undergo only a change in scale are called Eigen vectors

$$\boldsymbol{Tv} = \lambda \boldsymbol{v}$$

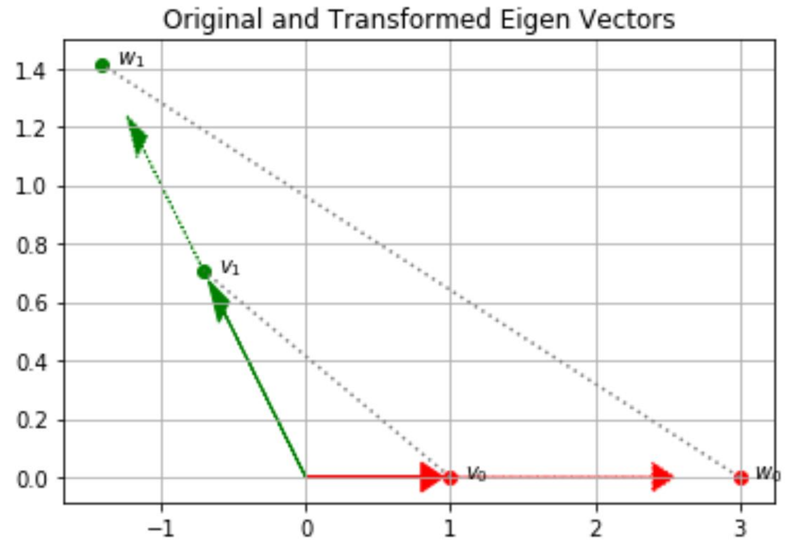
- How to find them: $(\boldsymbol{T} - \lambda \boldsymbol{I})\boldsymbol{v} = 0$ implies $|\boldsymbol{T} - \lambda \boldsymbol{I}| = 0$

$$\boldsymbol{T} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$\boldsymbol{y} = \boldsymbol{Tx} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$



- See: https://github.com/foxtrotmike/PCA-Tutorial/blob/master/Eigen.ipynb

- $\boldsymbol{T} = \begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix}$



- Eigen Vector: $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, Eigen Value: 3

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

  - Note scaling only

- Eigen Vector: $\begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix}$, Eigen Value: 2

$$\begin{bmatrix} 3 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix} = \begin{bmatrix} -1.414 \\ 1.414 \end{bmatrix} = 2 \begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix}$$

Mean

- Mean is the expected value of a given variable
- $\mu_Z = \frac{1}{N}\sum_{i=1}^{N} z_i$
- In vector form

$$\boldsymbol{z} = [z_1 \quad z_2 \quad \cdots \quad z_N]^T$$

$$\mu_Z = \frac{1}{N}\sum_{i=1}^{N} z_i = \frac{1}{N}\boldsymbol{z}^T\mathbf{1}$$

- Mean is the first order moment or expected value of a variable and can be written as

$$E[z] = \sum_i \underset{\text{value}}{z_i}\, \underset{\text{Fraction of the time (or probability) this value occurs}}{p_Z(z_i)}$$

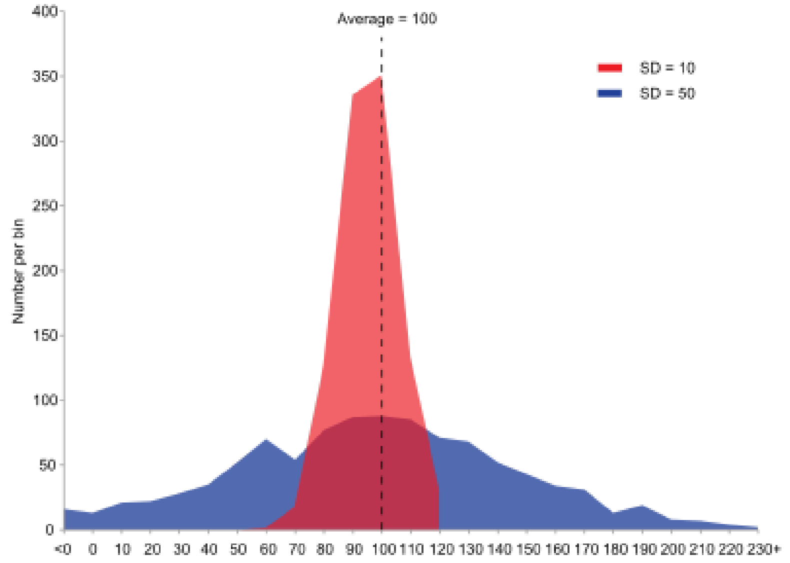
Example:
Let's say in a dice roll sequence, we see the following values (sorted): 1,1,2,3,3,4,4,5,6,6
The average value is:
1(2/10)+2(1/10)+3(2/10)+4(2/10)+5(1/10)+6(2/10) = 3.5

Variance

- Mean of the spread of a variable around its mean
- $var(z) = \frac{1}{N}\sum_{i=1}^{N}(z_i - \mu_z)^2 = \frac{1}{N}(\boldsymbol{z} - \mu_z)^T(\boldsymbol{z} - \mu_z)$
  - $\boldsymbol{z}$ is an N-dimensional vector composed of the values of all data points in the sample
- If mean is zero then $var(z) = \frac{1}{N}\boldsymbol{z}^T\boldsymbol{z} = \frac{1}{N}\|\boldsymbol{z}\|^2$
- $var(z) = E[(z - \mu_z)^2]$
  - Second moment or expected value of deviation from the mean

$$\boldsymbol{z} = [z_1 \quad z_2 \quad \cdots \quad z_N]^T$$

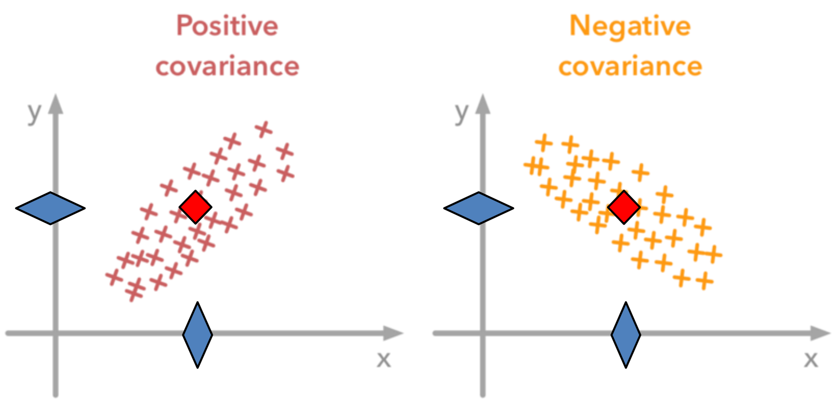
$$\mu_z = \frac{1}{N}\sum_{i=1}^{N} z_i = \frac{1}{N}\boldsymbol{z}^T\boldsymbol{1}$$



https://www.khanacademy.org/math/ap-statistics/summarizing-quantitative-data-ap/more-standard-deviation/v/review-and-intuition-why-we-divide-by-n-1-for-the-unbiased-sample-variance

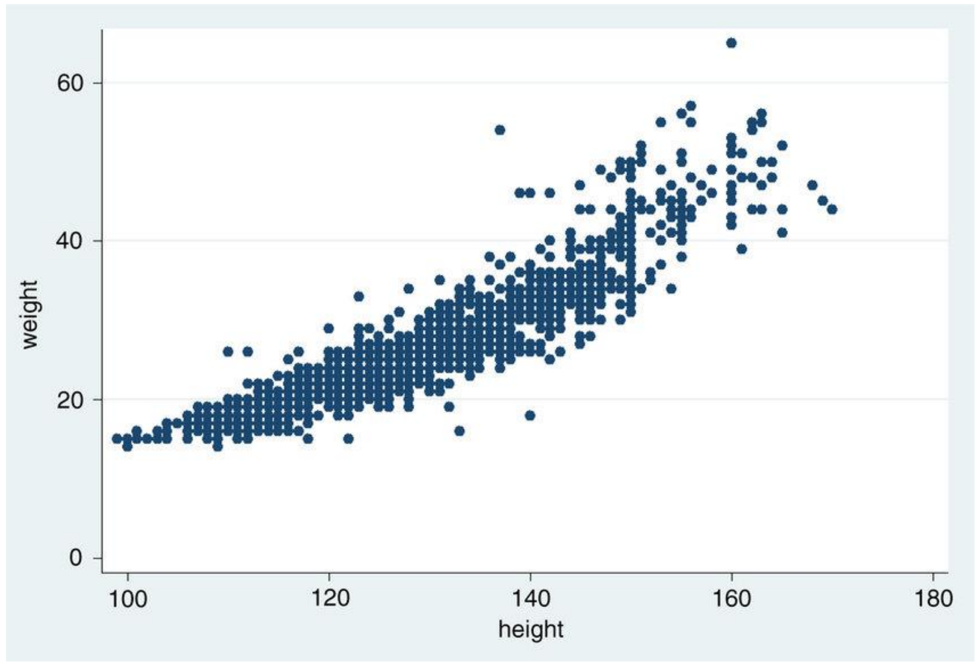
Co-Variance

- ***Given two random variables, to what extent are they linearly related to each other***
- $cov(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) = \frac{1}{N}(\boldsymbol{x} - \mu_x)^T(\boldsymbol{y} - \mu_y)$
  - Covariance is positive if, on average,
    - When one variable is above its mean then the other variable is above its mean too
    - When one variable is below its mean then the other variable is below its mean too
  - Covariance is negative if, on average,
    - When one variable is above the mean, the other is below its mean
- If the means of the two variables are zero: $cov(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{N}\boldsymbol{x}^T\boldsymbol{y}$
  - Maximum when the vectors are co-linear or parallel
- $cov(x,y) = E\left[(y - \mu_y)(x - \mu_x)\right]$
- Thus, $var(z) = cov(z,z)$



Correlation

- What is the association between two random variables?
  - Example: How are height and weight associated with each other?
- A related (and more general) concept from Information Theory is Mutual Information



Quantifying Correlation

- We can quantify the degree of linear association between two random variables through correlation coefficient

covariance $\mathrm{cov}_{XY} = \sigma_{XY} = E[(X - \mu_X)(Y - \mu_Y)]$

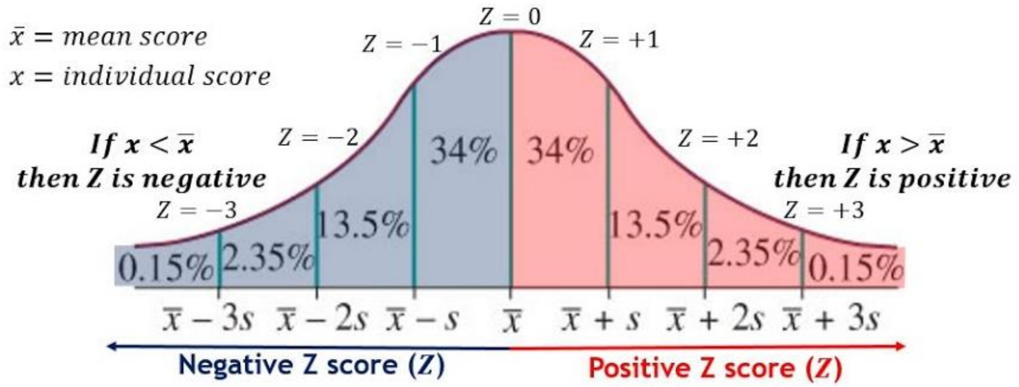
correlation $\mathrm{corr}_{XY} = \rho_{XY} = E[(X - \mu_X)(Y - \mu_Y)]/(\sigma_X \sigma_Y)$

- Notice if we "standardize" both variables to each have zero mean and unit standard deviation, then the correlation between them is equal covariance!
  - This can be achieved by standard scaling transform: $\frac{x-\mu_x}{\sigma_x}$
  - Implemented in sklearn
  - Once transform, each variable will have the same "scale" or range (if it is Gaussian distributed)

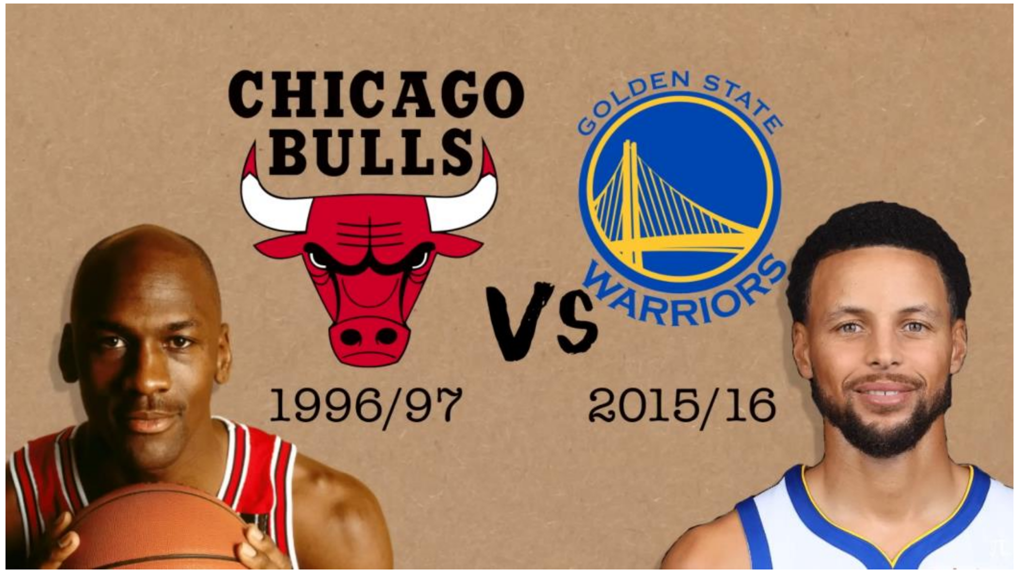
https://en.wikipedia.org/wiki/Covariance_and_correlation

Scaling transform

- $x' = T(x) = \frac{x - \mu_x}{\sigma_x}$
- Called
  - Z-Scoring
  - Standard Scaling



- Removes the effect of scaling across different variables
- Tells us how "extreme" a value is in terms of its variation
- Allows us to compare different variables with different variations

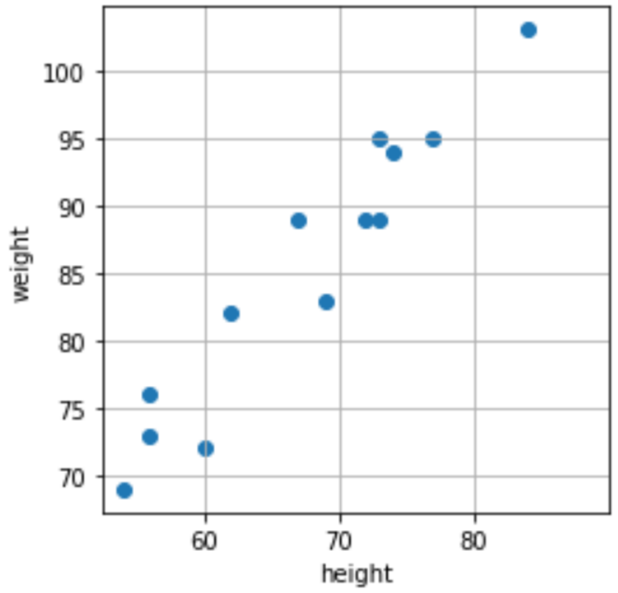


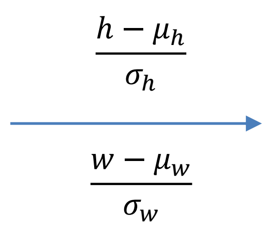
The Z-Factor Video by Numberphile

Covariance Matrix of a dataset

- Matrix of all pairwise covariances of all variables
- For two variables
  - $\boldsymbol{C} = \begin{bmatrix} cov(y,y) & cov(z,y) \\ cov(y,z) & cov(z,z) \end{bmatrix}$
- In general, if we have d-features then this will be a dxd matrix

Covariance Matrix Example

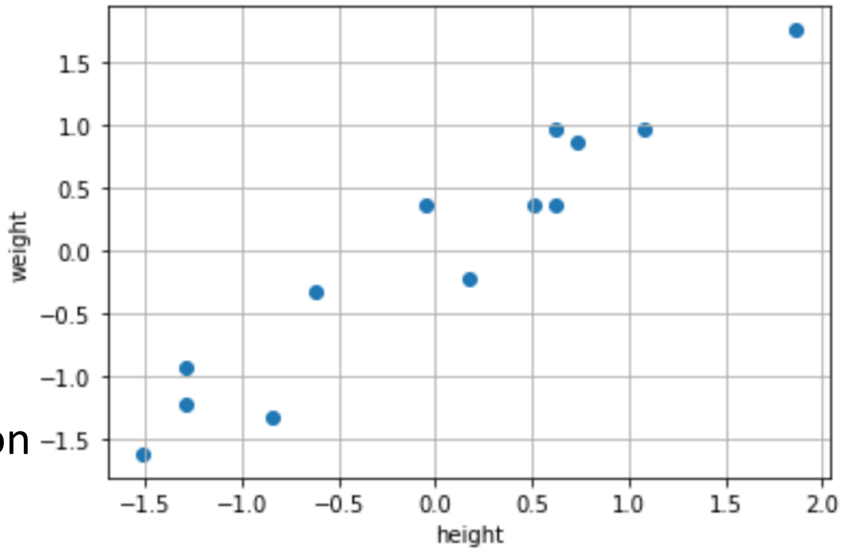




$$\frac{h-\mu_h}{\sigma_h}$$

$$\frac{w-\mu_w}{\sigma_w}$$

Standardization



```
The mean is [67.46 85.31]
The standard deviation is: [ 8.86 10.06 ]
The variance is: [ 78.56 101.14]
```

The co-variance matrix is: $\begin{bmatrix} 78.56 & 85.55 \\ 85.55 & 101.14 \end{bmatrix}$

```
The mean is [0 0]
The standard deviation is: [ 1 1 ]
The variance is: [ 1 1]
Total variance: 1+1 = 2.0
```

The co-variance matrix is: $\begin{bmatrix} 1 & 0.96 \\ 0.96 & 1 \end{bmatrix}$

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

Variance and Information

- Information content
  - The self-information of measuring X as outcome x
    - How surprising is an outcome?

$$\mathrm{I}_X(x) := -\log\left[p_X(x)\right] = \log\left(\frac{1}{p_X(x)}\right)$$

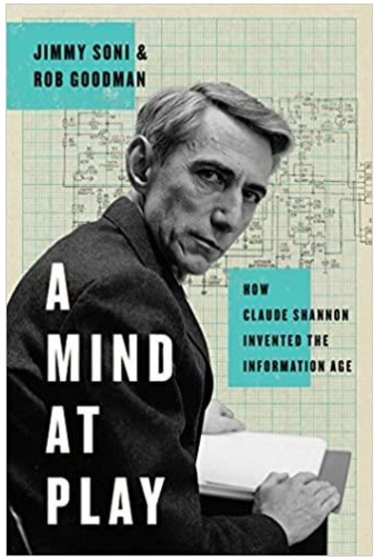
  - Shannon entropy: Expected information content of a random variable (measured in bits)
    - What is the average surprise of different values of a random variable?

$$\begin{aligned}\mathrm{H}(X) &= \sum_x -p_X(x)\log p_X(x)\\ &= \sum_x p_X(x)\,\mathrm{I}_X(x)\\ &\overset{\text{def}}{=} \mathrm{E}\left[\mathrm{I}_X(X)\right],\end{aligned}$$

Definition: Expected value

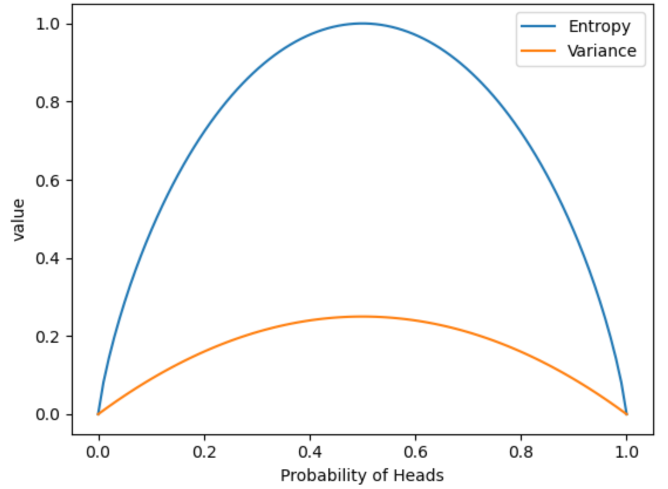
$$E[X] = \sum_x x p_X(x)$$

Claude Shannon



Relationship between Variance and Entropy for a Coin Toss (Bernouli Process) with probability of heads equal to p

```
tol = 1e-10; E=[]; V=[]; P = np.linspace(0,1,100)
for p in P:
    p1 = p; p0=1-p;
    e = -p1*np.log2(p1+tol)-p0*np.log2(p0+tol)
    # v = np.var(np.random.rand(1000)<=p) # sample variance
    v = p1*p0 # bernoulli distribution variance
    E.append(e);V.append(v)
plt.plot(P,E); plt.plot(P,V);
plt.show(); plt.legend(['Entropy','Variance']);
plt.xlabel('Probability of Heads'); plt.ylabel('value')
```



jee, Debabrata Mukher, and Makarand V. Ratnaparkhi. “On the Functional Relationship between Entropy and Variance with Related Applications.” *Communications in Statistics - Theory and Methods* 15, no. 1 (January 1, 1986): 291–311. https://doi.org/10.1080/03610928608829122.

When variables inform on their fellow variables…

Term & its (Close Counterpart)	Formula	Explanation
Entropy (Variance)	$H(X) = -\sum_{x \in X} P(x) log P(x)$	If I have a random variable X with which generates outcomes x with probability $P(x)$, then $H(X)$ quantifies how much surprise or information (in bits) is expected in the outcomes of X. A higher $H(X)$ means that outcomes can be expected to be more unpredictable.
KL Divergence (Distance)	$D_{KL}(P\|\|Q) = \sum_{x \in X} P(x) log \frac{P(x)}{Q(x)}$	If I have two probability distributions P and Q over the same variable X, then KL Divergence quantifies the difference or divergence between them. It tells us how much information (in bits) is lost when Q is used to approximate P.
Cross-Entropy (hinge loss)	$H(P,Q) = -\sum_{x \in X} P(x) log Q(x)$	If I have two probability distributions P and Q over the same variable X, then $H(P,Q)$ quantifies how "bad" is Q as an estimate of P (in terms of the number of additional bits required to encode P using Q).
Mutual Information (Correlation)	$I(X;Y) = \sum_{y \in X} \sum_{x \in X} P(x,y) log \frac{P(x,y)}{P(x)P(y)}$	If I have two random variables X and Y with a joint probability $P(x,y)$ and individual probabilities $P(x)$ and $P(y)$, then mutual information quantifies how much knowing Y tells me about X (in terms of reducing the uncertainty in my knowledge of X). It is a measure of statistical dependence.
Total Correlation (Multiple Correlation Coefficient)	It's a misnomer (actually not correlation) but Multi-variate Mutual Information	If I have d random variables, then total correlation quantifies the amount of information gained about one random variable through the other variables together, beyond what is gained from them individually.

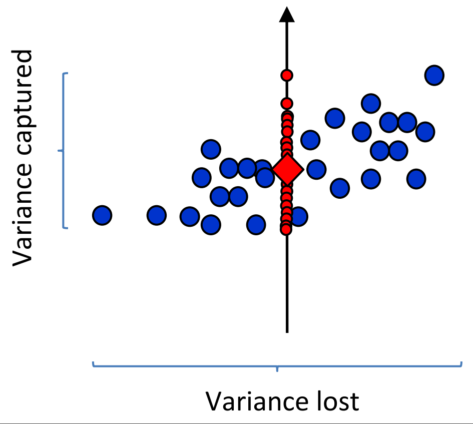
Principal Component Analysis: Uses

- Dimensionality Reduction
- Visualization and Exploratory Data Analysis
- Anomaly/Outlier Detection
- Understanding association between different variables
- Preprocessing
  - Noise Removal
  - Removal of unwanted effects
- Feature extraction for classification (with Grid Search)
- Data/Image Compression
- Conceptual basis for understanding a variety of related algorithms
  - Kernel PCA, ICA, SVD, NMF, Clustering methods, Topic Modelling, Canonical Correlation Analysis, PLS, Linear Discriminant Analysis, Multidimensional Scaling, t-SNE, UMAP, Bottleneck neural networks/Autoencoders, Dictionary Learning, Locally Linear Embeddings, Batch Correction/Normalization …

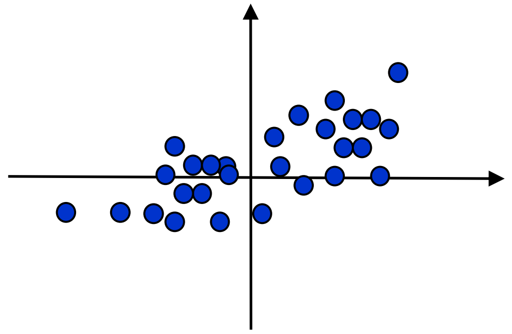
Data Dimensionality Reduction

- How can we reduce dimensions?
  - Drop features?
    - Equivalent to projecting data onto canonical axes unit vectors
      - Each point is originally expressed as a linear combination of two canonical axis vectors
    - Once we do this, we lose information about the other feature unless both features are related
    - Loss in variance

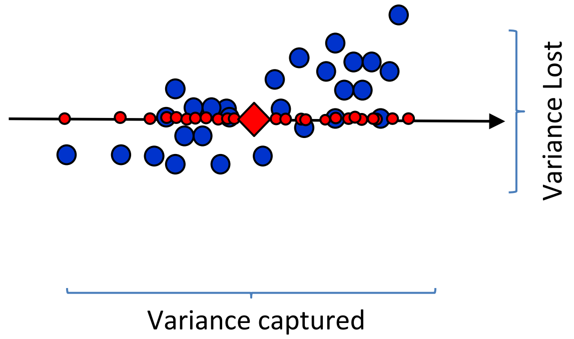
$z = \boldsymbol{w}^T\boldsymbol{x} = \begin{bmatrix}0\\1\end{bmatrix}^T \boldsymbol{x}$



$\boldsymbol{x} = \begin{bmatrix}x_1\\x_2\end{bmatrix} = x_1\begin{bmatrix}0\\1\end{bmatrix} + x_2\begin{bmatrix}1\\0\end{bmatrix}$

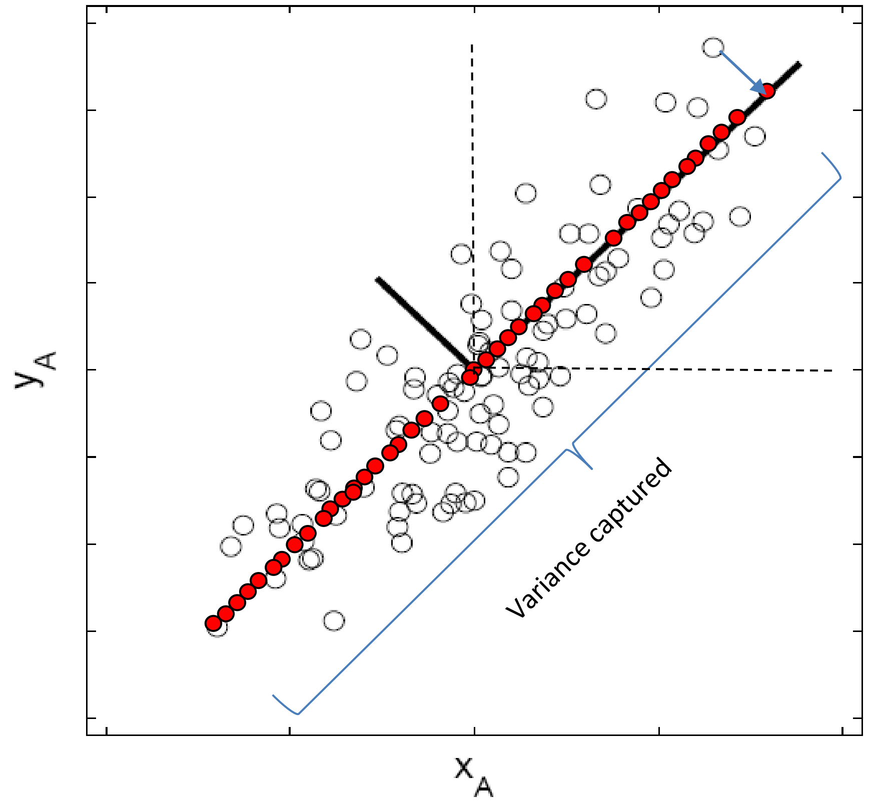


$z = \boldsymbol{w}^T\boldsymbol{x} = \begin{bmatrix}1\\0\end{bmatrix}^T \boldsymbol{x}$



Dimensionality Reduction as Projections

- Projections can be used for reducing dimensions
  - However, projecting data onto a unit vector loses information
  - We want to reduce the amount of information loss
  - Solution: Find and project along a direction (unit vector) along which information loss is minimum
    - A direction along which most of the variance is captured
  - How to do it?

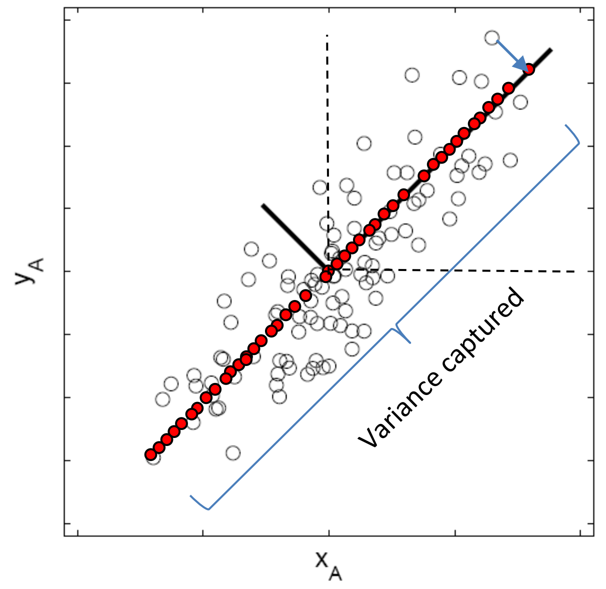


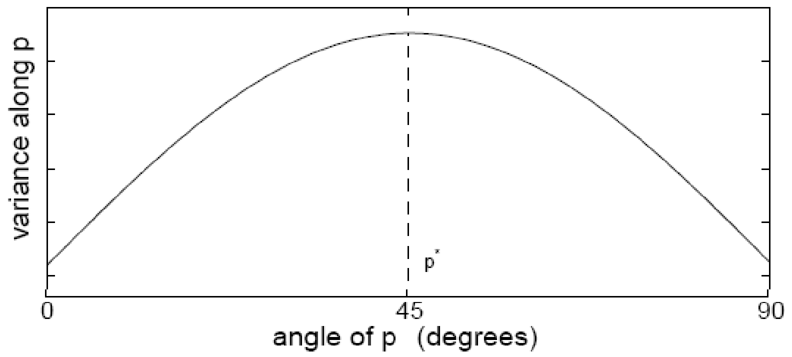
How to do it: Naïve Implementation

- Set p = 0
- For p from 0 to π in steps
  - Calculate projection unit vector
    - $\boldsymbol{w_p} = \begin{bmatrix} \cos(p) \\ \sin(p) \end{bmatrix}$
  - Project your data onto $z_i = \boldsymbol{w}_p^T \boldsymbol{x}_i$
  - Find the variance of the projected data

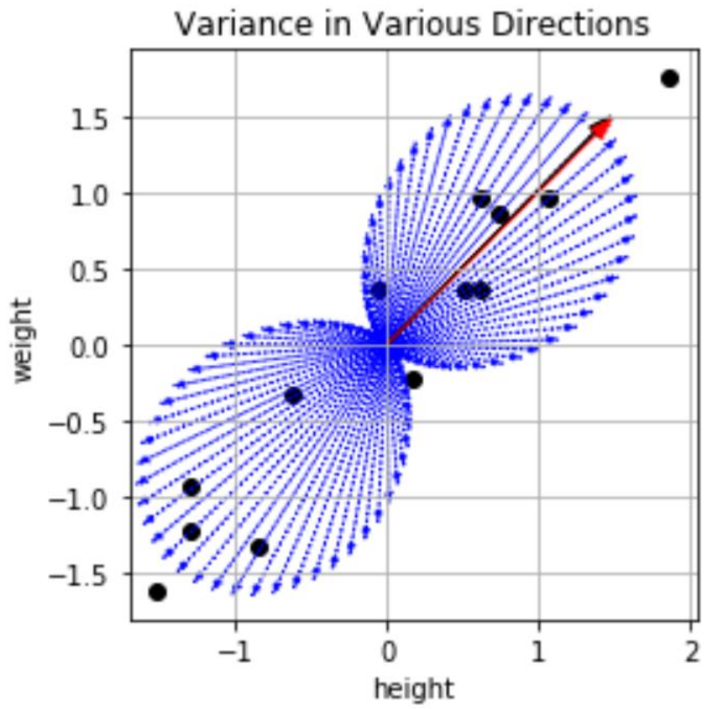
- Plot the variance across p
- Find the p that gives maximum variance

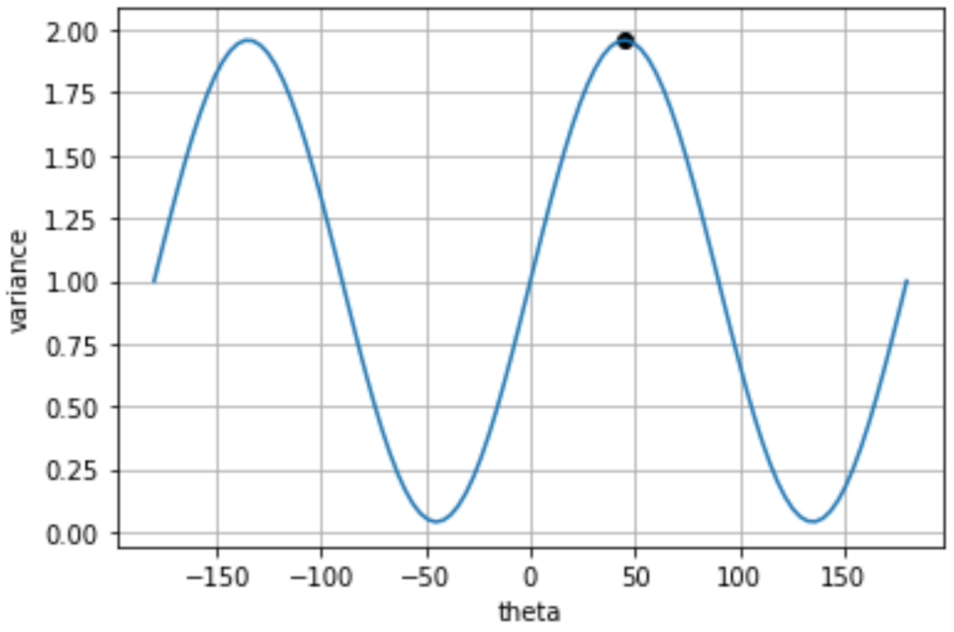
- Issues?





Using the naïve implementation

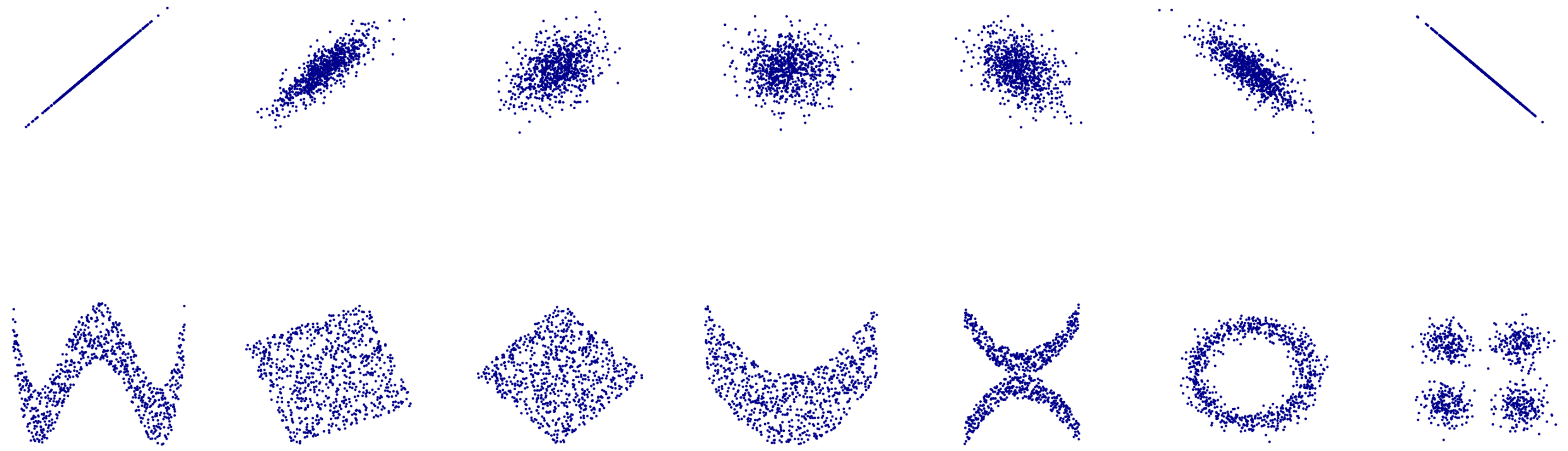




```
Direction of Maximum Variance: [0.70, 0.71]
```

Quiz Time: Find Principal Components

- What are the principal components for each data set below?



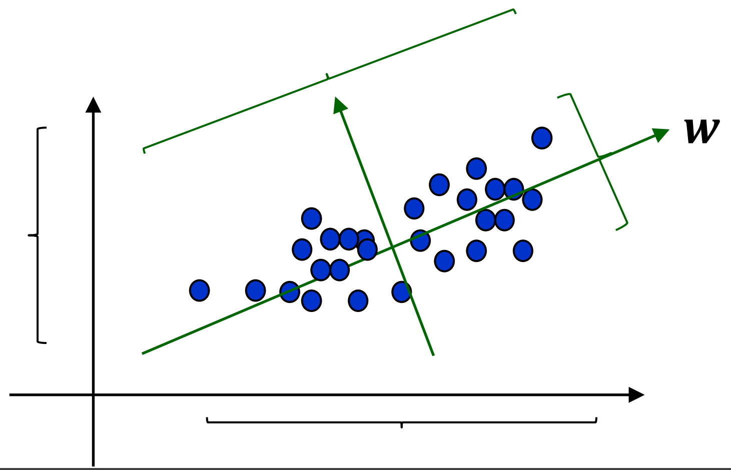
So what is PCA?

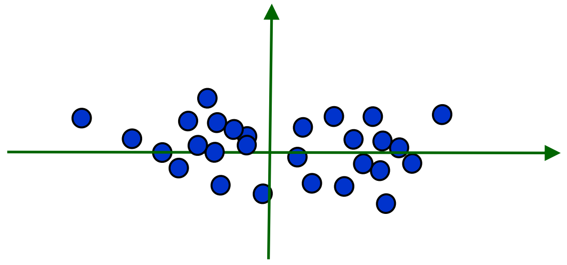
- A method for transforming data
  - Projecting the data onto directions (called principal directions or axis) such that the variance of the data along that direction is maximal
  - Projection of $\boldsymbol{x}$ on the direction of $\boldsymbol{w}$: $z = \boldsymbol{w}^T\boldsymbol{x}$
  - Projection of all data points in $\boldsymbol{X}_{(nxd)}$ along the direction of $\boldsymbol{w}$: $\boldsymbol{z} = \mathbf{X}\boldsymbol{w}$
  - Find $\boldsymbol{w}$ such that $Var(\boldsymbol{z})$ is maximum.
    - Or in other words:

$$\max_{\boldsymbol{w}} Var(\boldsymbol{z})$$

subject to:

$$\|\boldsymbol{w}\|^2 = \boldsymbol{w}^T\boldsymbol{w} = 1$$





PCA in the light of the REO Framework

- Representation
  - Linear projection along a direction vector

$$z = \boldsymbol{w}^T\boldsymbol{x}$$

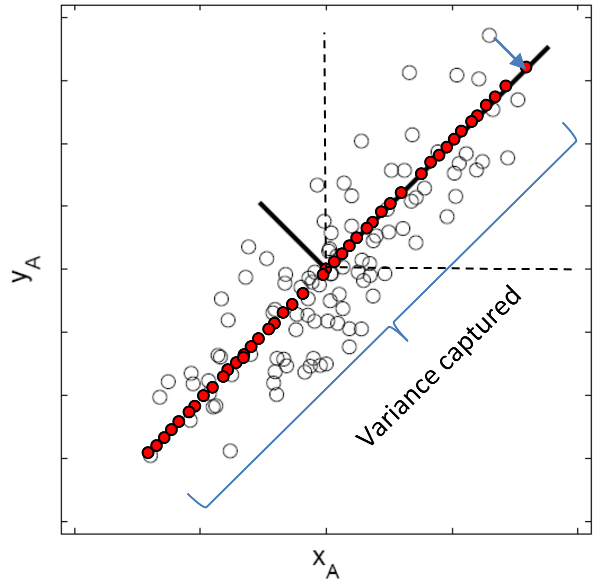
- Evaluation
  - What's a good direction?

$$\max_{\boldsymbol{w}} Var(\boldsymbol{z}) = Var\left(\boldsymbol{X}_{(\boldsymbol{n}\times\boldsymbol{d})}\boldsymbol{w}_{(\boldsymbol{d}\times\boldsymbol{1})}\right) = E[(\boldsymbol{z}-\boldsymbol{\mu_z})^{\boldsymbol{2}}]$$

subject to:

$$\|\boldsymbol{w}\|^2 = \boldsymbol{w}^T\boldsymbol{w} = 1$$

- Optimization
  - Solve using gradient descent
  - Or analytically!
    - ***Only if we had a formula for*** $Var(\boldsymbol{z})$***!!***



Formula for finding variance after projection

$$var(z) = var(\mathbf{w}^T\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}[(\mathbf{w}^T\mathbf{x}_i - \mathbf{w}^T\mu_x)^2]$$

$$= \frac{1}{N}\sum_{i=1}^{N}[(\mathbf{w}^T\mathbf{x}_i - \mathbf{w}^T\mu_x)(\mathbf{w}^T\mathbf{x}_i - \mathbf{w}^T\mu_x)]$$

$$= \frac{1}{N}\sum_{i=1}^{N}[(\mathbf{w}^T\mathbf{x}_i - \mathbf{w}^T\mu_x)(\mathbf{x}_i^T\mathbf{w} - \mu_x^T\mathbf{w})]$$

$$= \frac{1}{N}\sum_{i=1}^{N}[\mathbf{w}^T(\mathbf{x}_i - \mu_x)(\mathbf{x}_i - \mu_x)^T\mathbf{w}]$$

$$= \mathbf{w}^T\frac{1}{N}\sum_{i=1}^{N}[(\mathbf{x}_i - \mu_x)(\mathbf{x}_i - \mu_x)^T]\mathbf{w}$$

$$= \mathbf{w}^T\mathbf{C}\mathbf{w}$$

Here, $\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N}[(\mathbf{x}_i - \mu_x)(\mathbf{x}_i - \mu_x)^T]$ is the $d \times d$ sized "covariance matrix".

$$var(z) = \frac{1}{N}\sum_{i=1}^{N}(z_i - \mu_z)^2$$

$$\mu_z = \frac{1}{N}\sum_{i}^{N} z_i = \frac{1}{N}\sum_{i}^{N} w^T\boldsymbol{x}_i = w^T\frac{1}{N}\sum_{i}^{N}\boldsymbol{x}_i = w^T\boldsymbol{\mu_x}$$

$$\boldsymbol{\mu_x} = \frac{1}{N}\sum_{i}^{N}\boldsymbol{x}_i$$

Thus,

$$var(z) = \frac{1}{N}\sum_{i=1}^{N}(w^T\boldsymbol{x_i} - w^T\boldsymbol{\mu_x})^2$$

Principal Component Analysis

- We want to find a unit vector **w** that maximizes the variance along the projection
- This leads to the following constrained optimization problem

$$\max_{\boldsymbol{w}} Var(z) = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{C}\boldsymbol{w}$$

subject to:

$$\|\boldsymbol{w}\|^2 = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} = 1$$

- Using the method of Lagrange Multipliers to convert it to an Unconstrained optimization problem

$$\max_{\mathbf{w_1}} \ \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{C}\boldsymbol{w} - \lambda\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} - 1\right)$$

Method of Lagrange Multipliers

Constrained Optimization Problem

$$\max_{\mathrm{u}} f(u)\ s.t.\ g(u) = 0$$

$$\Updownarrow$$

$$\max_{\mathrm{u},\alpha} f(u) - \alpha g(u)$$

Unconstrained Optimization Problem

https://en.wikipedia.org/wiki/Lagrange_multiplier

- Taking the derivative of this with respect to ***w*** and substituting to zero, we get:

$$\boldsymbol{C}\boldsymbol{w} = \lambda\boldsymbol{w}$$

Principal Component Analysis

- The direction of maximum variance is $\boldsymbol{w}$, given by:

$$\boldsymbol{C}\boldsymbol{w} = \lambda\boldsymbol{w}$$

- $\boldsymbol{w}$ is the Eigen Vector Corresponding to the covariance matrix $\boldsymbol{C}$ with Eigen valueλ
  - It is called the principal direction

- Thus: The direction of maximum variance of the data X is the same as the eigen vector of its Covariance matrix (with the largest eigen value)

To refresh, see: https://github.com/foxtrotmike/PCA-Tutorial/blob/master/Eigen.ipynb

Further Principal Directions?

- So far we have found only the first principal component, How do we find the others?
- In general, if we desire to find the kth principal direction, we solve:

$\max_{\boldsymbol{w}} Var(z) = \boldsymbol{w}_k^{\boldsymbol{T}} \boldsymbol{C} \boldsymbol{w}_k$
subject to:
$\|\boldsymbol{w}_k\|^2 = \boldsymbol{w}_k^T \boldsymbol{w}_k = 1$
$\boldsymbol{w}_k^T \boldsymbol{w}_j = 0 \ \forall j < k$ (orthogonality constraint to avoid information redundancy)

- Fortunately, similarly the covariance matrix of the data C gives us all these principal directions as its *d* eigen vectors

An algorithmic view of how PCA Works

- **Input**: $\boldsymbol{X}_{N\times d}$
- **Output**: A transformation matrix **W** which can be used for dimensionality reduction
- **Parameters**: Selection of principal components
  - Proportion of variance
  - Number of principal components (k)
  - Which principal components to retain

- **Internal Working**
  - Normalize or standardize data
    - Calculate feature wise-mean and standard deviation and normalize data to zero mean and unit standard deviation (to achieve invariance of scales – esp when different features have different scales or ranges)
  - Find Covariance Matrix
  - Find Principal Components (Eigen Value Problem)
  - Select Principal Components
    - Using Scree Graph or Intuition
  - Reduce dimensionality by Projection along selected components

- Code: https://github.com/foxtrotmike/PCA-Tutorial/blob/master/pca-lagrange.ipynb

Algorithm for PCA

sklearn PCA

- Each of the N samples is stored as a d-dimensional vector $\boldsymbol{x}^{i} = [x_1^i \quad \cdots \quad x_d^i]^T$
- The data matrix is formed as $\boldsymbol{X}_{d \times N} = [\boldsymbol{x}^{1} \quad \cdots \quad \boldsymbol{x}^{N}]$
- Centralize (or standardize) each sample in the data as $\bar{\boldsymbol{x}}^{i} = \boldsymbol{x}^{i} - \boldsymbol{\mu}_x$ $\quad \bar{X} = [\bar{x}^{1} \quad \cdots \quad \bar{x}^{N}]$
- Compute the $d \times d$ Covariance Matrix $\boldsymbol{C} = \dfrac{\bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^T}{\boldsymbol{N}-\boldsymbol{1}}$
- Find the Eigen Values $\lambda_1, \lambda_2 \ldots \lambda_d$ & *d*-dimensional Eigen Vectors $\boldsymbol{w_1}, \boldsymbol{w_2} \ldots \boldsymbol{w_d}$, of ***C*** using $\boldsymbol{C}\lambda_i = \boldsymbol{w}_i\lambda_i$ or Singular Value Decomposition (SVD) and sort the eigen values in decreasing magnitudes. Normalize the eigen vectors to have unit norm.
- Calculate the required dimension *k* based on proportion of variance approach or scree graph
- Form $\boldsymbol{W = [\, w_1 \, w_2 \ldots w_k \,]_{(d \times k)}}$
- A centralized or standardized vector ***x'*** can be projected using $\boldsymbol{z = W^T x'}$
- A projected vector can be used to reconstruct the standardized vector ***x'*** using $\boldsymbol{x' = Wz}$ followed by decentralization $\boldsymbol{x_r = x' + \mu_x}$

`pca.fit(X)`

`pca.explained_variance_`

`pca.explained_variance_ratio_`

`pca.transform(X)`

`pca.inverse_transform(Z)`

The co-variance matrix is: $\begin{bmatrix} 1 & 0.96 \\ 0.96 & 1 \end{bmatrix}$ (Symmetric!)

Eigen vector 1 (Principal Direction 1):

$$w_1 = \begin{bmatrix} 0.7071 \\ 0.7071 \end{bmatrix}, \ \lambda_1 = 1.96$$

Variance of data after projecting along w_1: 1.96 (eq. to α_1)

Eigen vector 2 (Principal Direction 2):

$$w_2 = \begin{bmatrix} -0.7071 \\ 0.7071 \end{bmatrix}, \ \lambda_2 = 0.04$$

Variance of data after projecting along w_1: 0.04 (eq. to α_2)

Total Variance after data projection is 1.96+0.04 = 2.0
Total Variance after projection is equal to Total Variance before projection.

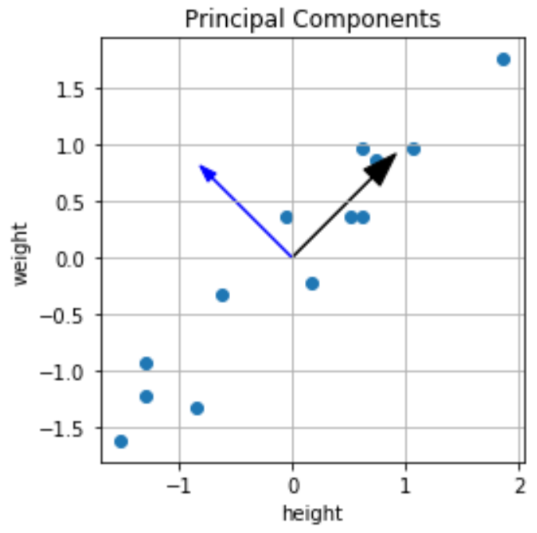
Fraction of variance captured along each PC:
Using PC-1: 1.96/2 = 0.98
Using PC-1 and PC-2: (1.96+0.04)/2 = 1.0

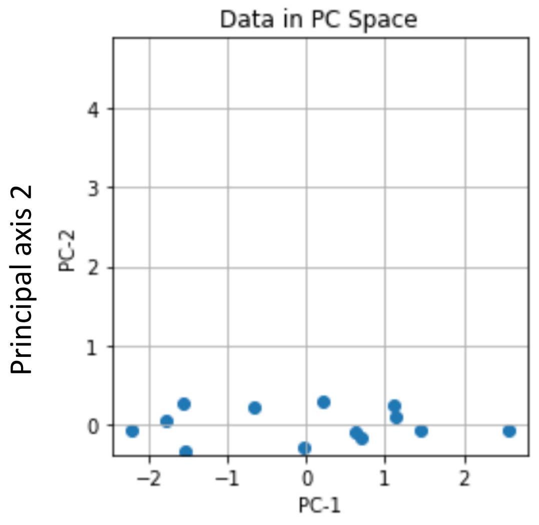
The two PC vectors (principal directions) are orthogonal to each other $w_1^T w_2 = 0$

The PC Matrix is $W = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix}$

The inverse of W is: $W^{-1} = \begin{bmatrix} 0.7071 & 0.7071 \\ -0.7071 & 0.7071 \end{bmatrix} = W^T$

Thus, $W^T W = I$ (The transpose of W matrix is inverse of W)





Principal axis 1

Things to note

- For this data
  - There are two principal components: The one with the largest variance (eigen value) is called the first principal component whereas the other one is called the second principal component.
  - The variance along the first principal component is higher in comparison to the second.
  - The variance along the first projected direction is higher than the variance along original features which is 1.0 after normalization. Thus, the principal component is a direction that captures more information than any of the original features alone.
  - The norm of each of the principal components is 1.0.
  - The two principal components are orthogonal to each other (i.e., their dot product is zero)
  - After project, the data is centered and the resulting features are orthogonal (statistically uncorrelated) to each other, i.e., PC-1 does not carry any information about PC-2
  - The principal component matrix and its transpose are inverses of each other, i.e., $\mathbf{W}^{\mathbf{T}}\mathbf{W} = \mathbf{I}$
  - The eigen values correspond to the amount of captured variance: The fraction of variance captured along a direction is exactly equal to the fraction of eigen values. Thus, the first principal component corresponds to the largest eigen value and so on.
  - The plot of the fraction of captured variance up to k principal components (called the scree plot) can be used to select how many principal components to retain when reducing dimensionality. For the original data used in this example, upto 98% variance is along the first principal component. Therefore, if the second principal component is dropped, the loss of information will be only ~2%.

PCA for dimensionality reduction

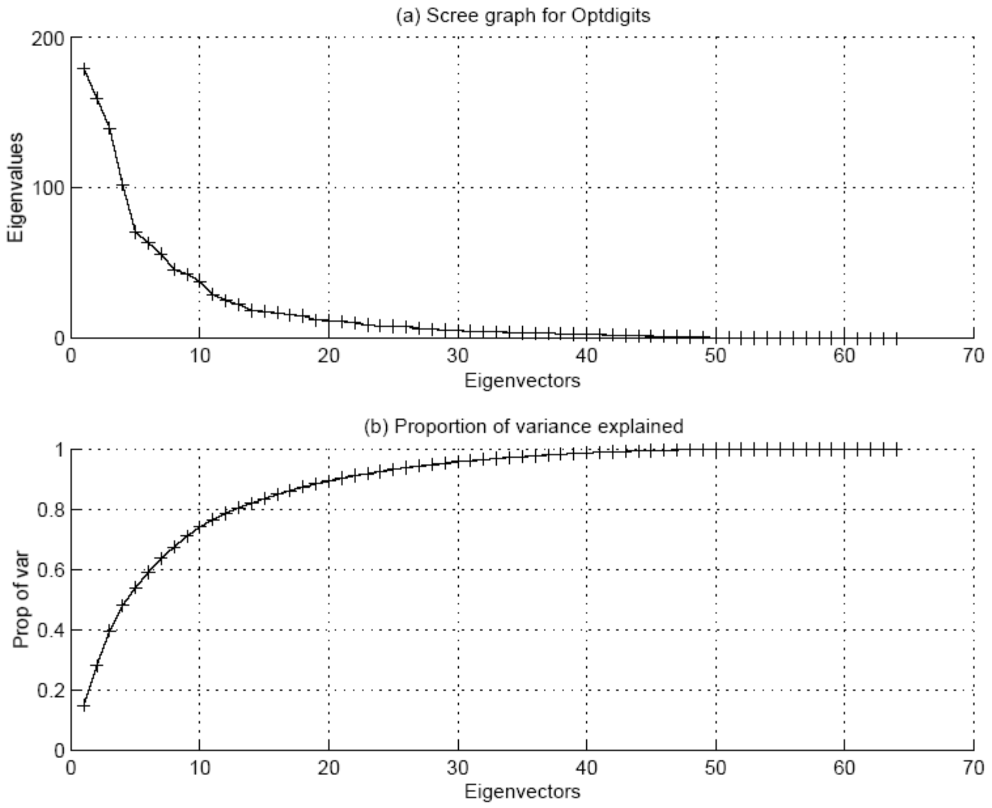
- Eigen values of the covariance matrix with small magnitudes have small contribution to the total variance of the data and these can be discarded without major loss of information
  - We can retain 90% variance of the data by storing the largest eigen values and eigen vectors which contribute 90% of the variance and projecting our data on these bases
- We can thus calculate Proportion of Variance (PoV) explained by the k eigen vectors with the largest eigen values as follows:

$$PoV(k) = \frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when λ_i are sorted in descending order

- Typically, stop at PoV > 0.9
- Scree graph plots PoV vs *k*
  - Stop at "elbow"

- Now the *d*-dimensional data vector *x* with associated mean vector μ can be projected using the *k x d* dimensional ***W*** matrix containing the *k* selected eigen vectors to obtain a *k* < *d* dimensional data vector *z* using: $\boldsymbol{z} = \boldsymbol{W}^T\boldsymbol{x}$

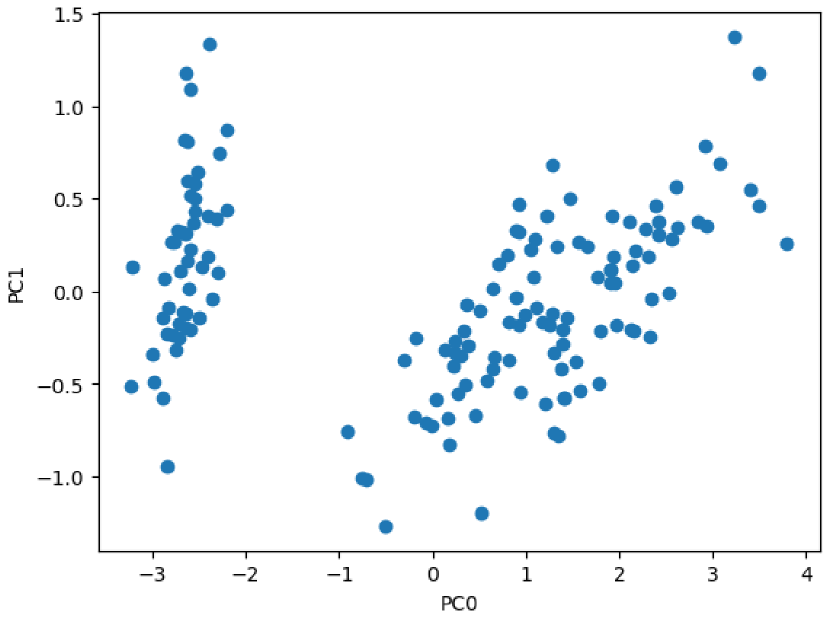


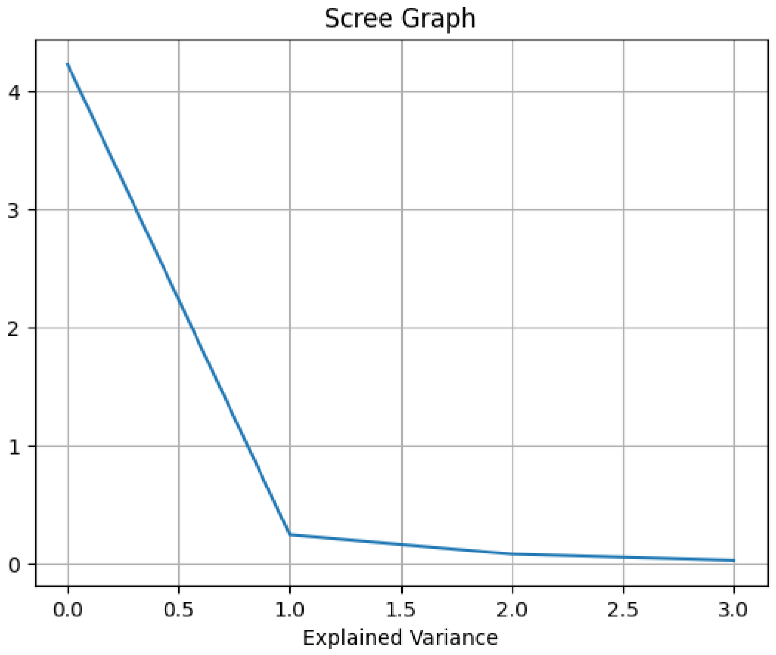
How to code?

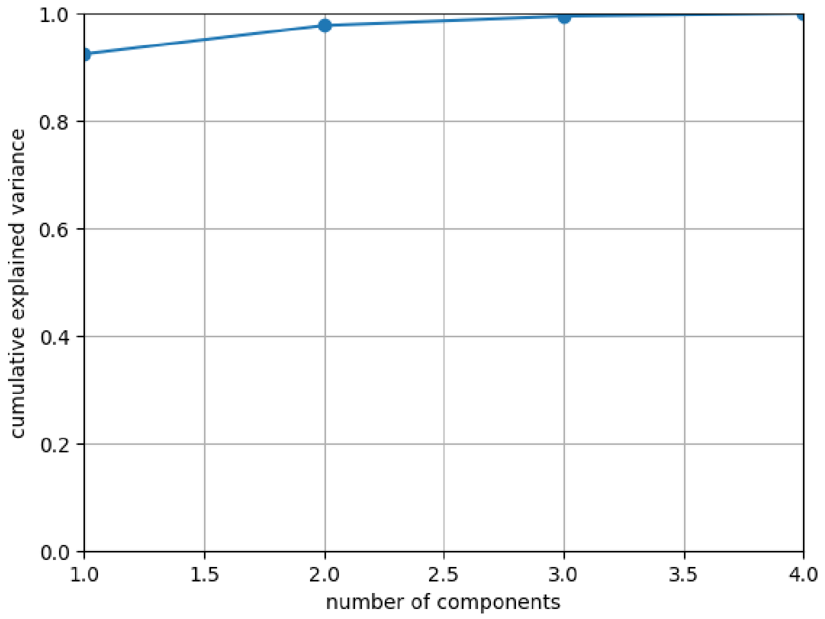
- **`Fitting PCA to training data`**
  - `from sklearn.decomposition import PCA`
  - `pca = PCA(n_components=4)`
  - `pca.fit(X) #rows are samples, columns are features`
- **`Projection`**
  - `Z = pca.transform(X)`
- **`Visualization`**
- **`Scree Graph`**
  - `plt.plot(np.cumsum(pca.explained_variance_ratio_),'o-’)`
- **`Reconstruction`**
  - `Xr = pca.inverse_transform(Z)`

```python
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
from sklearn.decomposition import PCA #import PCA
pca = PCA(n_components=4)
pca.fit(X) #training PCA
projected = pca.transform(X) #projecting the data onto Principal components

# plot scatter plot of the data in terms of the first two PCs
plt.scatter(projected[:,0],projected[:,1])
plt.xlabel('PC0');plt.ylabel('PC1')

#plot the variance and the scree plot
plt.figure()
plt.plot(pca.explained_variance_); plt.grid();
plt.xlabel('Explained Variance')
plt.title('Scree Graph')

plt.figure()
plt.plot(np.arange(len(pca.explained_variance_ratio_))+1,np.cumsum(pca.explained_variance_ratio_),'o-')
plt.axis([1,len(pca.explained_variance_ratio_),0,1])
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance');
plt.grid()
plt.show()
```

PCA for Iris dataset

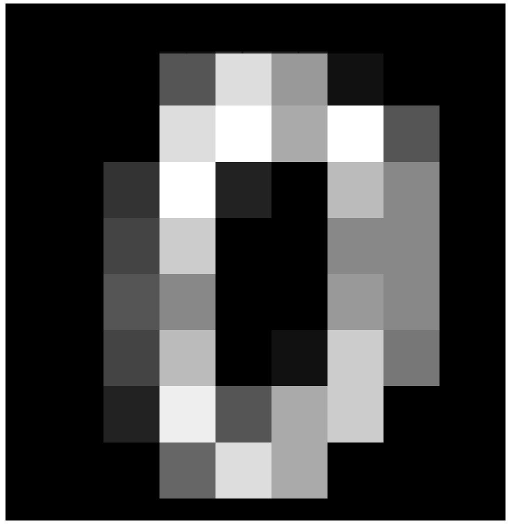


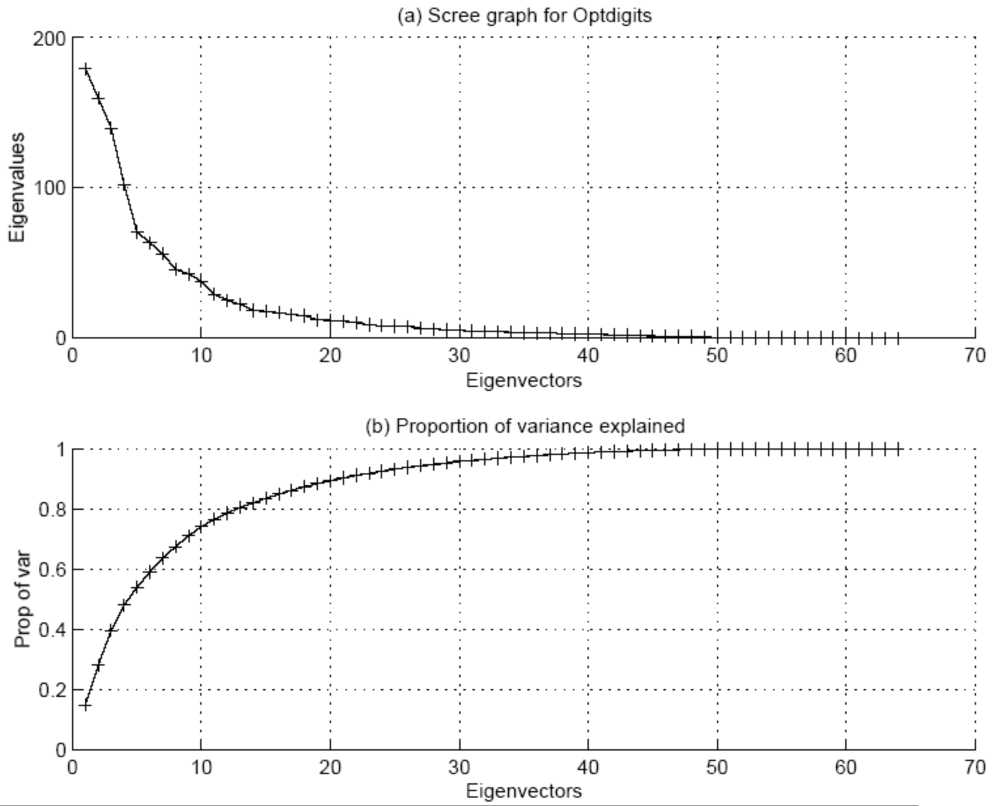


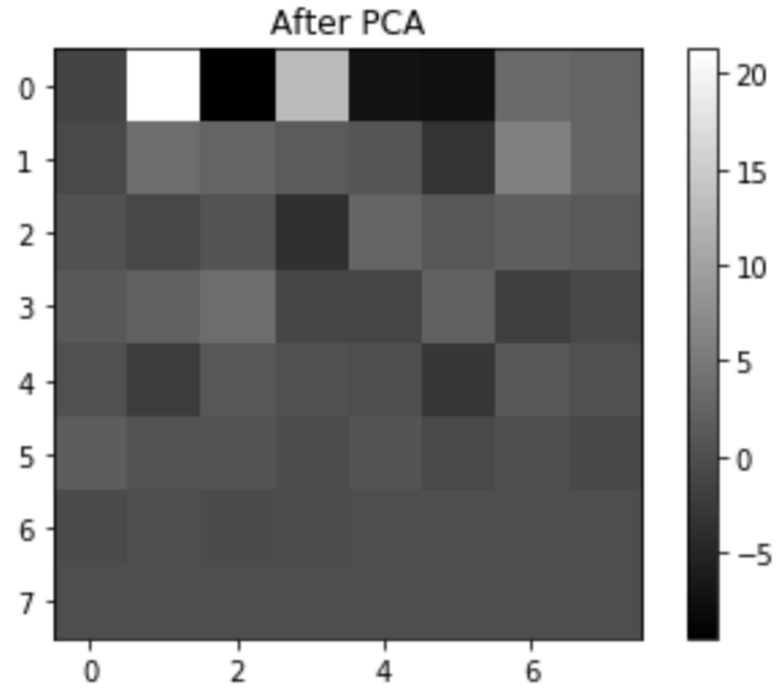


Example

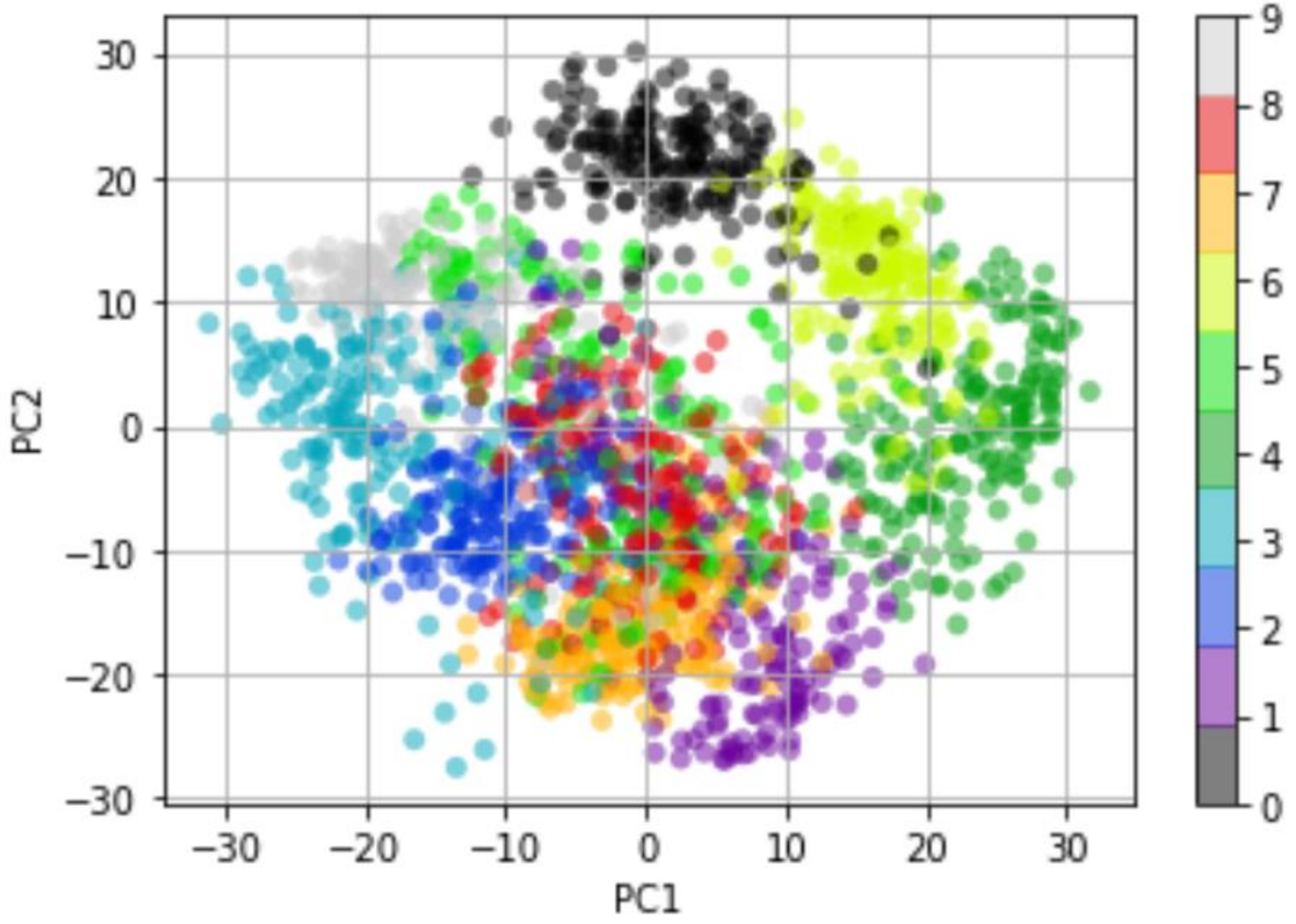
- MNIST visualization
- X: Nx (d=64)
- Scree Graph




(a) Scree graph for Optdigits
Eigenvalues
Eigenvectors
(b) Proportion of variance explained
Prop of var
Eigenvectors



Visualization



PCA: Reconstruction

- We know that $\boldsymbol{z} = \boldsymbol{W}^T\boldsymbol{x}$ (assuming $\boldsymbol{x}$ is centered) therefore

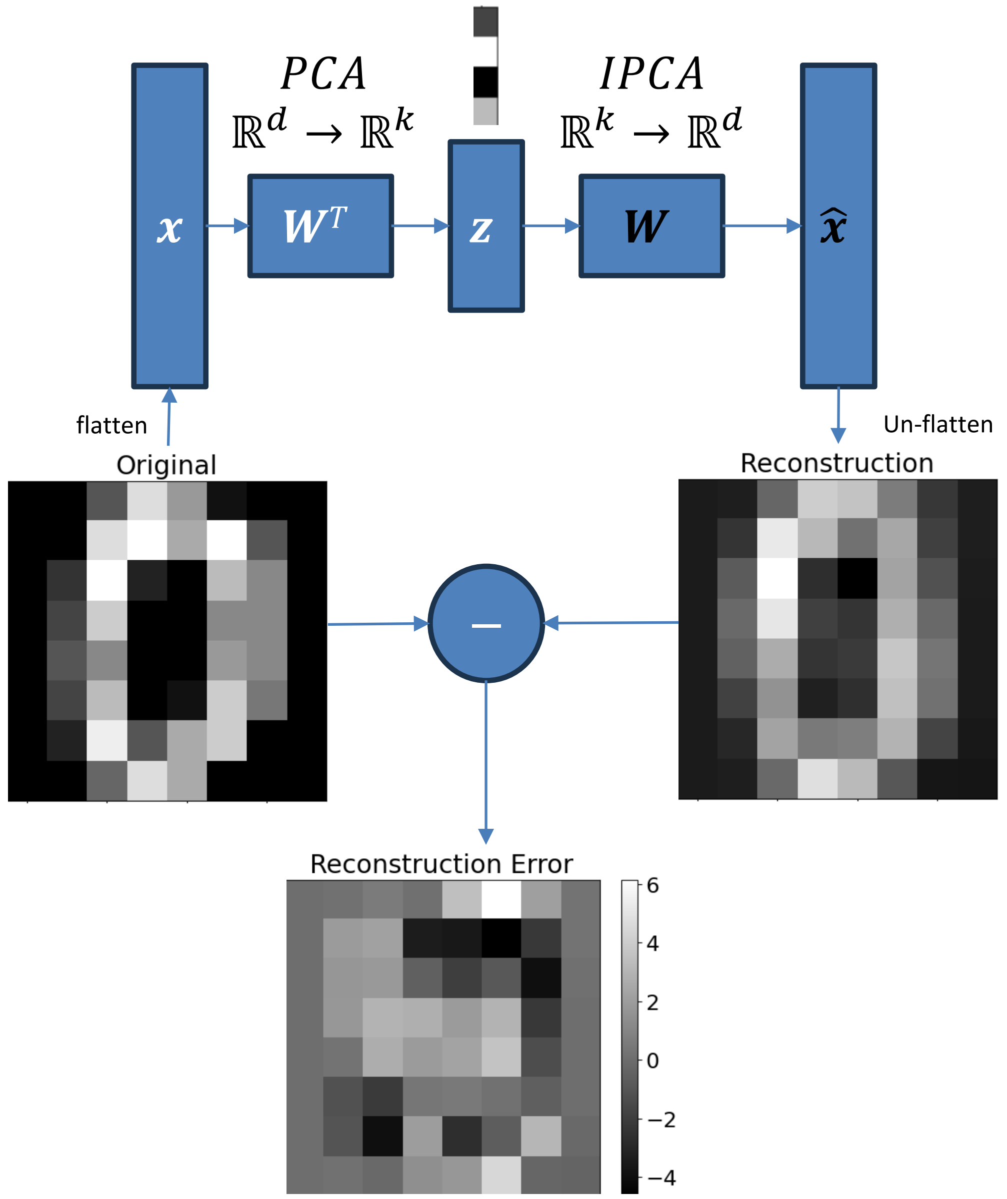
$$\hat{\boldsymbol{x}} = (\boldsymbol{W}^T)^{-1}\boldsymbol{z}$$
$$\Rightarrow \hat{\boldsymbol{x}} = \boldsymbol{W}_{(d\times k)}\boldsymbol{z} \quad \because \boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}$$

- The reconstruction error is given by

$$E_{rec} = \sum_{i=1}^{N} \left\|\hat{\boldsymbol{x}}^{\boldsymbol{i}} - \boldsymbol{x}^{\boldsymbol{i}}\right\|$$

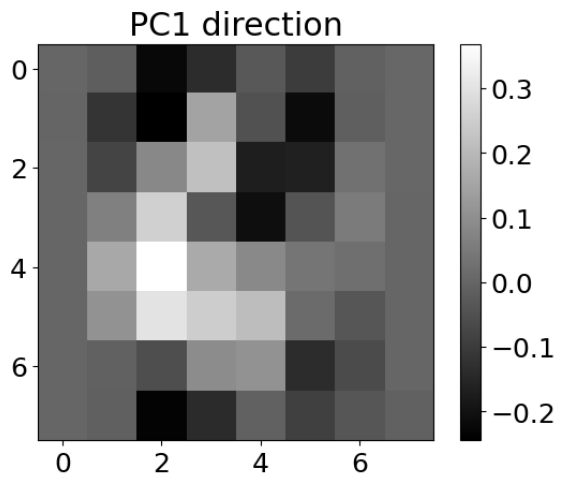
- Another way of interpreting PCA is that it finds orthogonal direction vectors such that after projecting data onto to them, the reconstruction error is minimal.

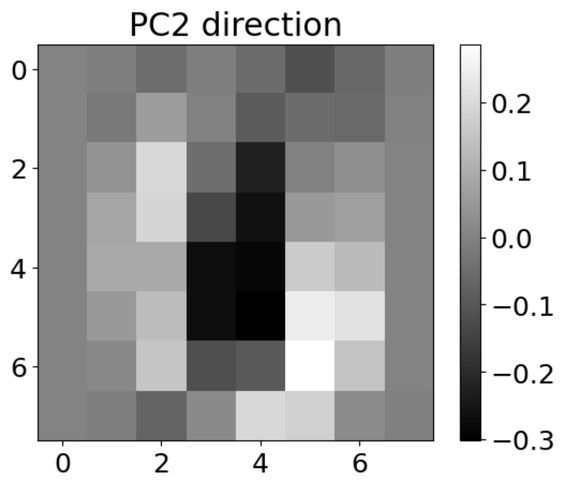
$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} \left\|\hat{\boldsymbol{x}}^{\boldsymbol{i}} - \boldsymbol{x}^{\boldsymbol{i}}\right\| \; \boldsymbol{s.th.} \boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{I}$$



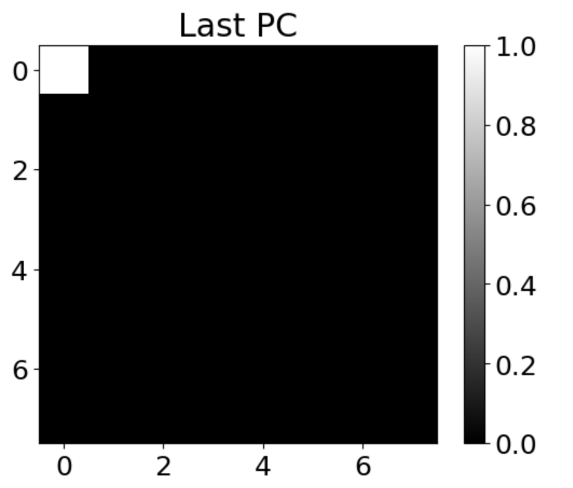
Visualizing the Principal Directions

- Notice that in the case of the MNIST example, each eigen vector will itself be 64 dimensional and can be visualized as an 8x8 image. These images can be called “eigen” digits as they show the **characteristic variations** in pixels in the given dataset.





...

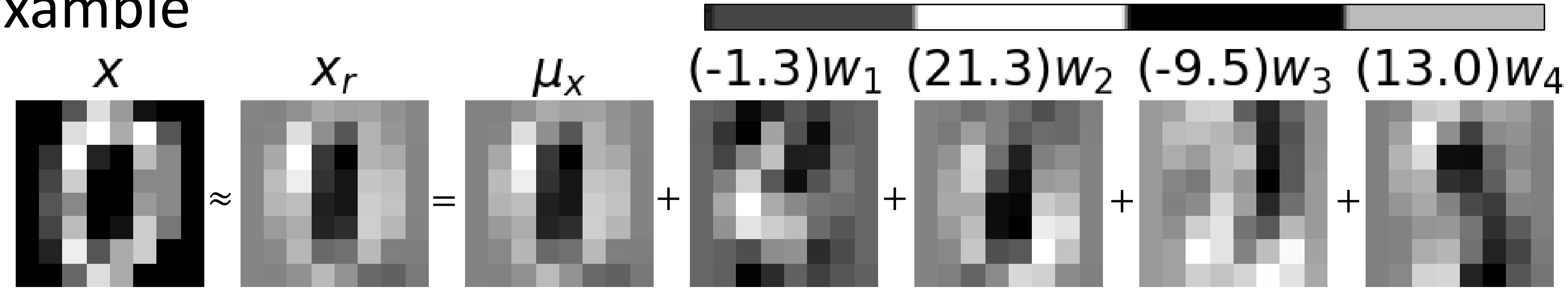


An example as linear combination of eigen vectors

- We have expressed each data point in k-dimensions, where each dimension corresponds to a certain principal component eigen vector. This allows us to write the reconstruction as a linear combination of these vectors as follows:

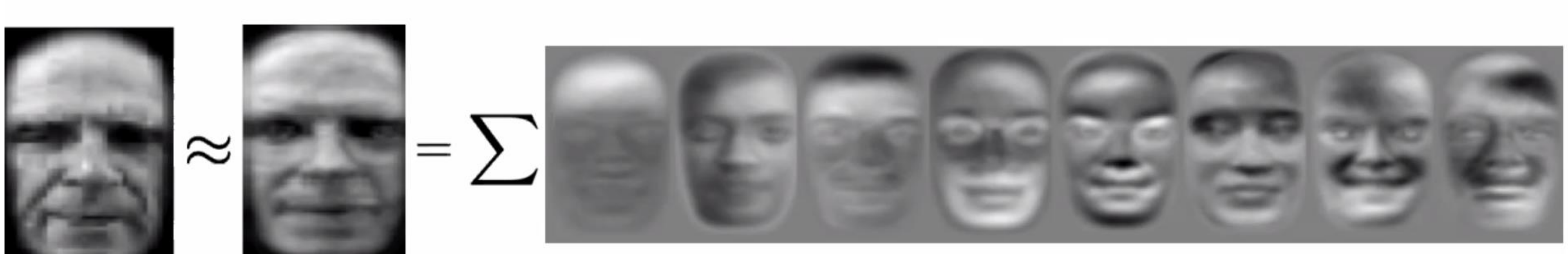
$$\mathbf{x}_{reconstructed} = \mu_{\mathbf{x}} + \mathbf{W}_{(d,k)}\mathbf{z} = \mu_{\mathbf{x}} + z_1\mathbf{w}_1 + z_2\mathbf{w}_2 + \ldots + z_k\mathbf{w}_k$$

- Example



Eigen Faces

- PCA Widely used in face detection and recognition
- Allows expressing a face image in terms of a small number of components



Eigen Faces Code: https://scikit-learn.org/stable/auto_examples/decomposition/plot_faces_decomposition.html

Turk, M. "Pentland. Eigenfaces for recognition." K. Cogn. Neurosci 4 (1991): 72-86.

Advanced: Snapshot Method for when d>>N

- If the input dimension (d) is large then the size of the covariance matrix is also large making its calculations computationally demanding
- It is known that for a d x N matrix the maximum number of non-zero eigenvectors is min(d-1,N-1)
- If N < d, then we can compute the eigen vectors w_i' of

$$C'_{(N\times N)} = \frac{\bar{X}^T X}{N-1}$$

instead of C. The eigen values for both C and C' are same and the eigen vectors of C can be obtained from those of C' using

$$w_{i_{(d\times 1)}} = \bar{X}_{(d\times N)} w'_{i\,(N\times 1)}$$

Advanced: Snapshot Method

- Each of the N samples is stored as a d-dimensional vector $x^i = \begin{bmatrix} x_1^i & \cdots & x_d^i \end{bmatrix}^T$
- The data matrix is formed as $X = \begin{bmatrix} x^1 & \cdots & x^N \end{bmatrix}$
- Compute the mean m=[m$_1$ … m$_d$]^T from *X* using $m_i = \dfrac{1}{N}\sum_{j=1}^{P} x_i^j$
- Centralize each sample in the data as $\bar{x}^i = x^i - m$ $\bar{X} = \begin{bmatrix} \bar{x}^1 & \cdots & \bar{x}^N \end{bmatrix}$
- Compute Covariance Matrix $C' = \dfrac{\bar{X}^T \bar{X}}{N-1}$
- Find the Eigen Values (sorted in decreasing values) $\lambda'_1, \lambda'_2 \ldots \lambda'_d$ & *d*-dimensional Eigen Vectors $w'_1, w'_2 \ldots w'_d$, of *C′* using $C'\lambda' = w'\lambda'$.
- Calculate the required dimension *k* based on proportion of variance-based approach explained earlier for a given threshold
- Form $W = [\, w_1\, w_2 \ldots w_k\,]_{(d \times k)}$ $w_i = \bar{X} w'_i$
- Unit-Normalize the vector w_i
- A vector *x* can be projected using $z = W^T(x - m)$

Advanced: Theorem Proofs Used but not given

- Spectral Theorem: As **C** (covariance matrix) is symmetric positive definite thus its Eigen vectors with different Eigen values are orthogonal and the resulting projected data directions are uncorrelated.
  - It also states that the eigen values are non-negative.
  - Also, as the eigen vectors in W are orthogonal, $WW^T = I$ or W's inverse is W^T
- Proof that Eigen values of the covariance matrix are equal to the variance of the data after projection along those directions
- The objective function of PCA is convex: As the variance is calculated as $Var(z) = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{C}\boldsymbol{w}$, and if C (the covariance matrix) is positive semi-definite then the objective function $-Var(z) = -\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{C}\boldsymbol{w}$ is not non-convex.

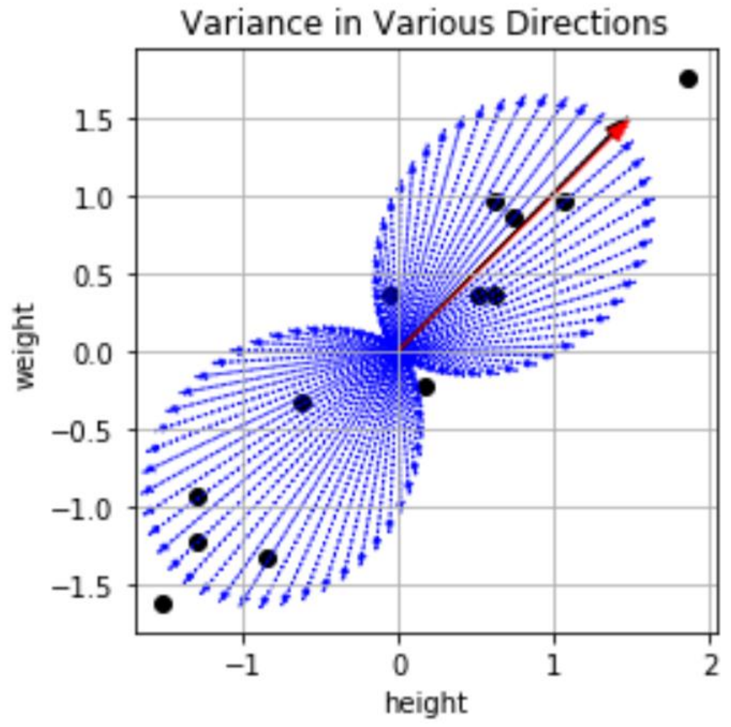
REO for PCA with SRM

- Basic Principle: Find orthogonal directions of maximal variance
- Representation: $f(\boldsymbol{x_i}) = \boldsymbol{w^T x_i}$
  - Project the data onto a direction vector $\boldsymbol{w}$
- Evaluation
  - Regularization: Minimize the norm of $\boldsymbol{w}$ or make sure $\boldsymbol{w^T w} = \|\boldsymbol{w}\|^2 = 1$
  - Empirical error term is the "Reconstruction Error" or "Lost Variance"
    - "Lost Variance": Loss of information (or variance) after projection:

      Total Variance (V) – Variance in data after projection ($Var(z) = \boldsymbol{w^T C w}$)

$$\boldsymbol{E}(\boldsymbol{X}; \boldsymbol{w}) = V - \boldsymbol{w^T C w}$$

- Optimization
  - $\min_{\boldsymbol{w}} \frac{\lambda}{2} \boldsymbol{w^T w} + \left(V - \boldsymbol{w^T C w}\right)$
  - Closed form solution after taking derivative: $\boldsymbol{Cw} = \lambda \boldsymbol{w}$ leads to Eigen value problem
    - Find a "characteristic" vector $\boldsymbol{w}$ for the matrix $\boldsymbol{C}$ such that multiplying $\boldsymbol{w}$ by $\boldsymbol{C}$ has a scaling effect on $\boldsymbol{w}$
    - The Eigen Value formulation also allows to determine multiple "orthogonal" eigen vectors. The eigen values λ tell about the importance of each and can be used to pick PCs based on "explained variance"
  - Can be kernelized leading to Kernel PCA!
- Practical Notes:
  - Always plot the scree plot or the proportion of variance plot
  - Look at the eigen vectors



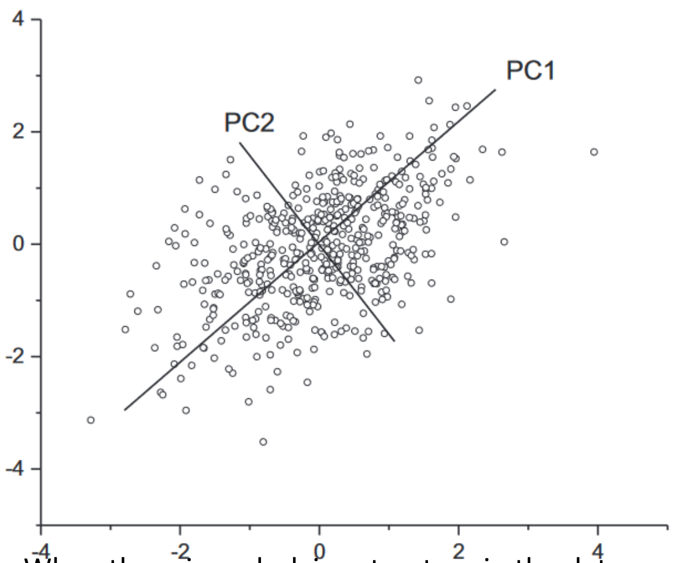
Variance after projection

$$\begin{aligned} \mathrm{Var}(z) = \mathrm{Var}(\boldsymbol{w^T x}) &= \mathrm{E}[(\boldsymbol{w^T x} - \boldsymbol{w^T \mu})^2] \\ &= \mathrm{E}[(\boldsymbol{w^T x} - \boldsymbol{w^T \mu})(\boldsymbol{w^T x} - \boldsymbol{w^T \mu})] \\ &= \mathrm{E}[\boldsymbol{w^T (x - \mu)(x - \mu)^T w}] \\ &= \boldsymbol{w^T}\, \mathrm{E}[\boldsymbol{(x - \mu)(x - \mu)^T}]\boldsymbol{w} = \boldsymbol{w^T} \mathrm{C}\, \boldsymbol{w} \end{aligned}$$

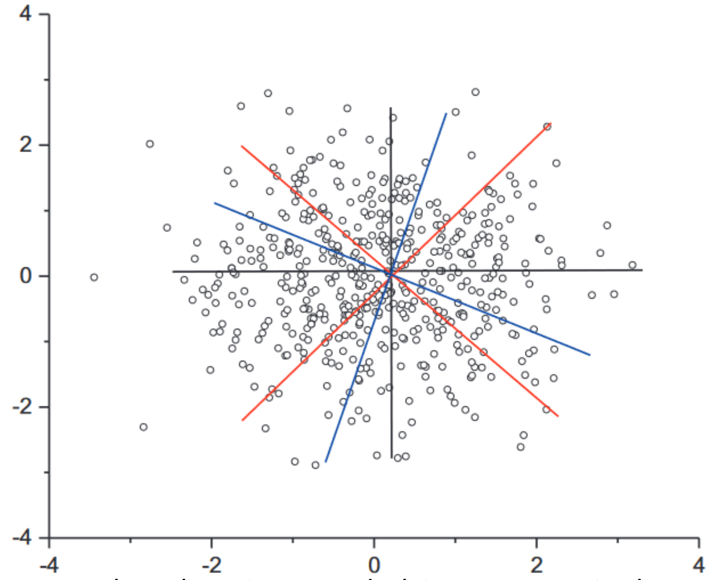
$$\boldsymbol{C} = \frac{1}{N}\boldsymbol{X X^T} \xrightarrow{\textit{Kernel Trick}} \boldsymbol{C} = \frac{1}{N}\boldsymbol{K}$$

$\boldsymbol{X}$ is Nxd matrix with of N examples each with d features. Each feature has zero mean and unit standard deviation. $\boldsymbol{K}$ is NxN (centered) kernel matrix.

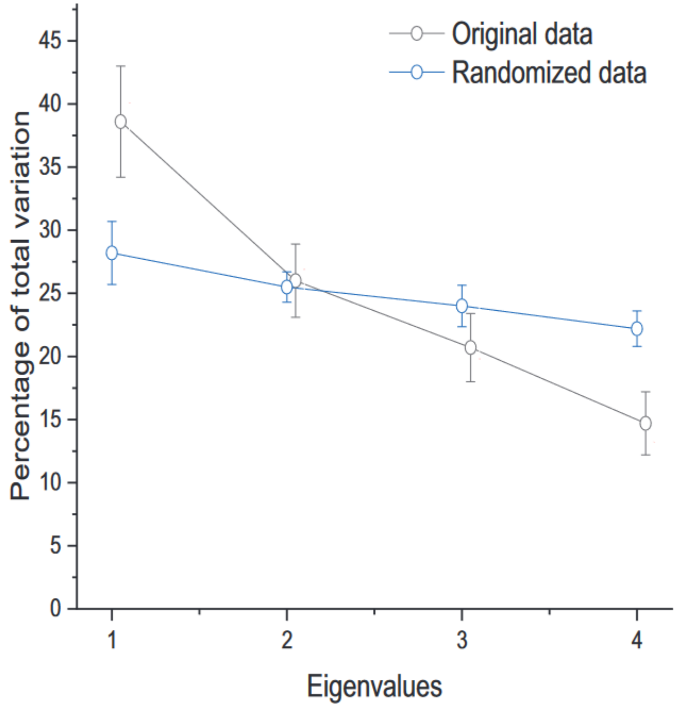
Be careful with your principal components



When there is underlying structure in the data, the PC's are distinct with different levels of variance captured along each



When there is NO underlying structure in the data, the PC's are NOT distinct

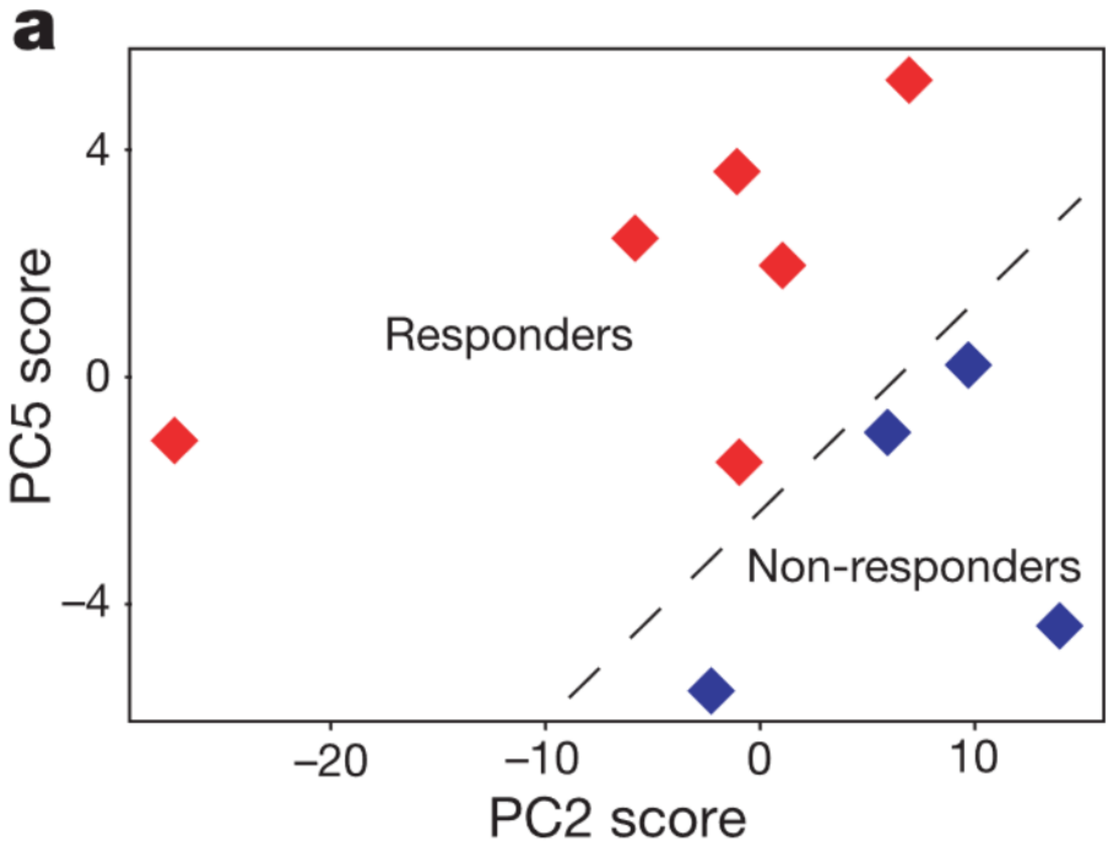


- Different components need to be distinct from each other to be interpretable otherwise they only represent random directions.
- **Sample correlation matrices will always result in a pattern of decreasing eigenvalues even if there is no structure**.
- Tests are, therefore, needed to discern real patterns from illusionary ones: PC-scores calculated from non-distinct PC's have very large standard errors and cannot be used for reasonable interpretations.

"By randomizing the values of each "feature" between "examples" … the structure of the original data is broken up and the test is, thus, how large value of these statistics can you get in a data set without structure just by chance."

Björklund, Mats. "Be Careful with Your Principal Components." *Evolution* 73, no. 10 (2019): 2151–58. https://doi.org/10.1111/evo.13835.

How many PCs to choose?



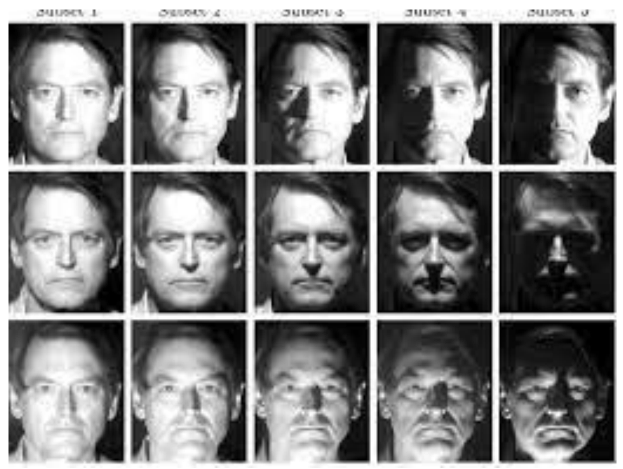
Thanks to Roy Goodacre for pointing me to this work ☺

Andrew Clayton, T., John C. Lindon, Olivier Cloarec, Henrik Antti, Claude Charuel, Gilles Hanton, Jean-Pierre Provost, et al. "Pharmaco-Metabonomic Phenotyping and Personalized Drug Treatment." *Nature* 440, no. 7087 (April 2006): 1073–77. https://doi.org/10.1038/nature04648.

Shen, Bo, Xiao Yi, Yaoting Sun, Xiaojie Bi, Juping Du, Chao Zhang, Sheng Quan, et al. "Proteomic and Metabolomic Characterization of COVID-19 Patient Sera." *Cell* 182, no. 1 (July 9, 2020): 59-72.e15. https://doi.org/10.1016/j.cell.2020.05.032.

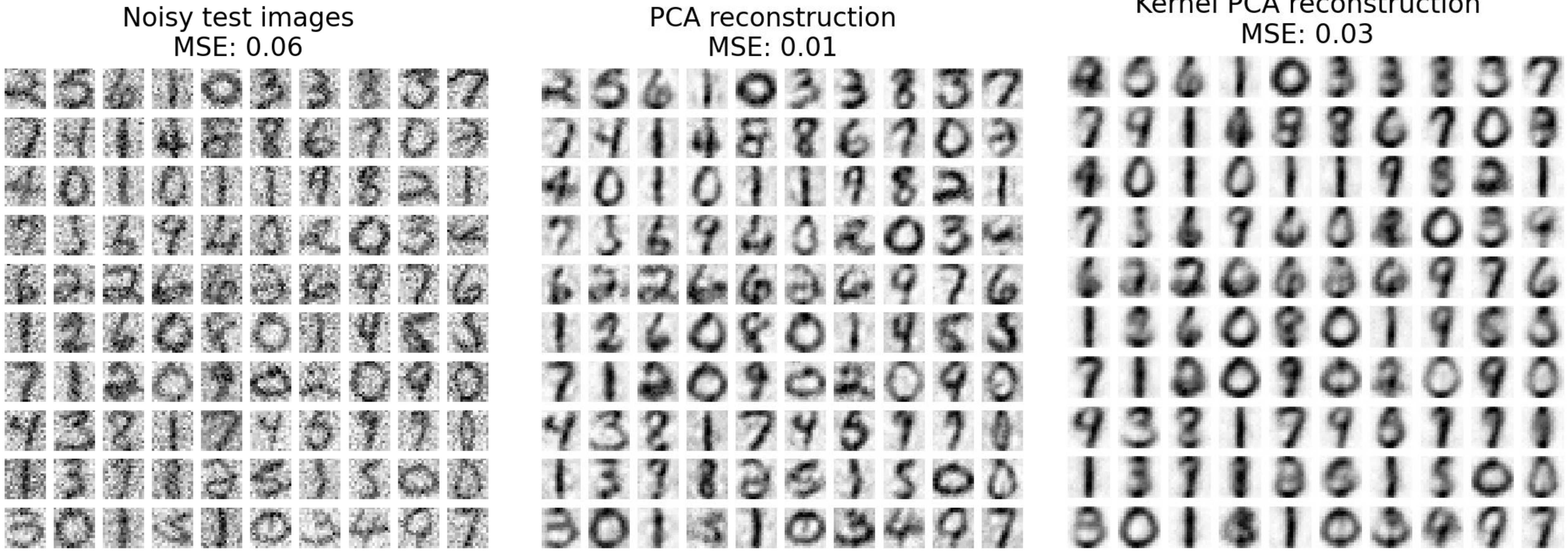
It’s <u>not</u> always the first few principal components

- Remember PCA and its eigen vectors aren’t magical
- They are just showing characteristics of the data
  - The principal directions are literally eigen (or characteristic) vectors of the covariance matrix of the data!
  - Specifically, the first PC is along the direction of the biggest variance
  - But what is the source of that variation?
    - Is it differences between classes?
    - Is it an unrelated factor?
      - For image data, the first principal component can be associated with variation in illumination or lighting which leads to the biggest change in pixel values but may not communicate any information about the underlying difference in classes depending upon the application!
      - The last components can be variations due to noise!



Changes in illumination can be the source of the biggest variation in images. Consequently, in order to perform face recognition that is robust to such “non-causal” variations, we may choose to drop the first few PCs!

PCA can compress and denoise



- By removing principal components associated with noise or unwanted variations and then reconstructing data, we can achieve denoising.

Image denoising using kernel PCA

PNAS COMMENTARY OPEN ACCESS

Check for updates

Why the simplest explanation isn’t always the best

Eva L. Dyer[a] and Konrad Kording[b,1]

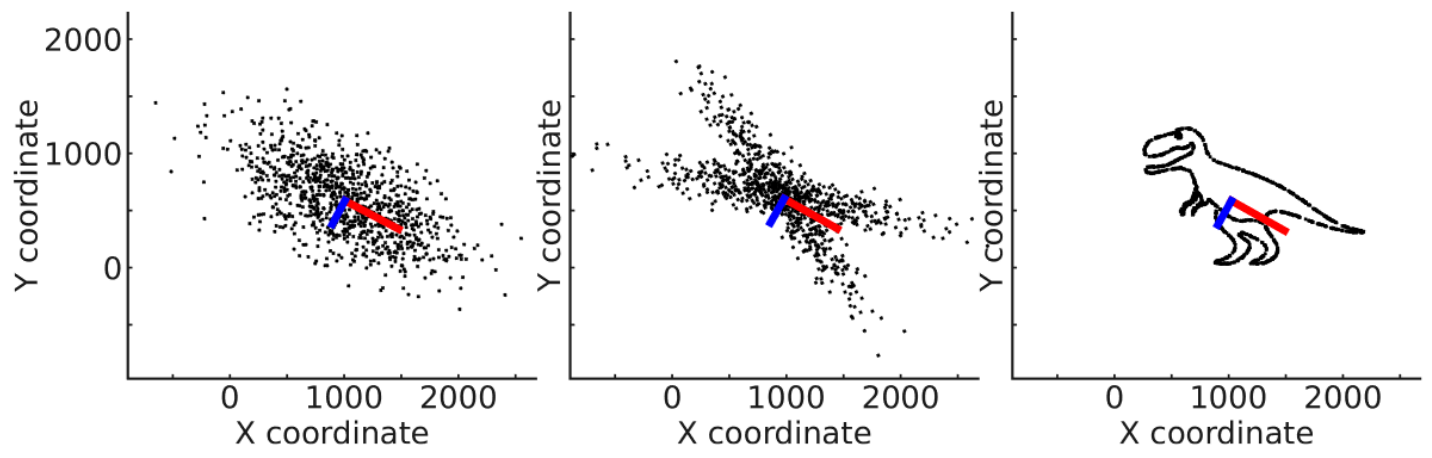


Fig. 1. PCA can see structure that does not exist and miss structure that exists. All these datasets have the same principal components. (*Left*) if the data are Gaussian, then PCA is the ideal technique, extracting all the structure that is there. (*Middle*) when data are not Gaussian, PCA may “see” dimensions that do not exist, in this case stemming from there being multiple Gaussians. In such a case, relaxing the assumption of orthogonality could allow a model to extract the relevant aspects. (*Right*) when data are highly structured but not simple (also see ref. 3), PCA will not discover the relevant structure, but will see structure that is in a way not even there. Indeed, in this case of a single line graph another technique, such as isomap, would discover that the whole dinosaur is just a single line, or a 1D-manifold embedded in 2D.

Dyer, Eva L., and Konrad Kording. “Why the Simplest Explanation Isn’t Always the Best.” Proceedings of the National Academy of Sciences 120, no. 52 (December 26, 2023): e2319169120. https://doi.org/10.1073/pnas.2319169120.

PCA Conclusions

- **Other variants**
  - Incremental PCA
  - Robust PCA
  - Kernelized PCA

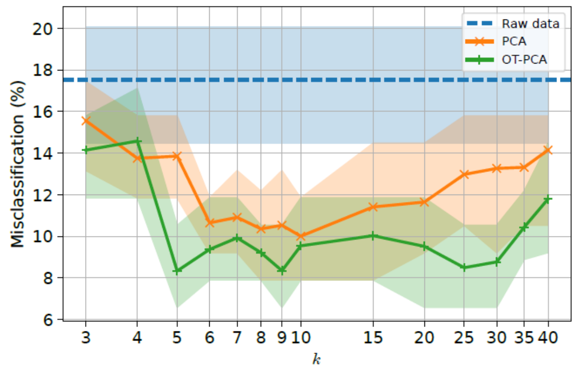
- **PCA Tutotrial Notes**
- https://github.com/foxtrotmike/PCA-Tutorial/blob/master/Eigen.ipynb
- https://github.com/foxtrotmike/PCA-Tutorial/blob/master/pca-lagrange.ipynb

Advanced: PCA as an optimal transport problem

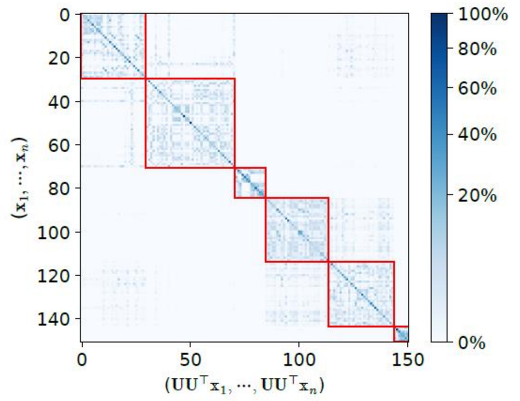
- A key requirement for dimensionality reduction is to incorporate global dependencies among original and embedded samples while preserving clusters in the embedding space.
- Reformulate a subspace recovery problem with an OT objective and show that it indeed yields the standard PCA when using least-squares cost and no regularization

$$\min_{\substack{\boldsymbol{\pi}\in\Pi(\frac{1}{n}\mathbf{1}_n,\frac{1}{n}\mathbf{1}_n)\\ \mathbf{U}\in\mathrm{St}(d,k)}} \sum_{i,j=1}^{n,n} \|\mathbf{x}_i - \mathbf{U}\mathbf{U}^\top\mathbf{x}_j\|_2^2\,\pi_{ij} - \varepsilon\,\mathrm{H}(\boldsymbol{\pi}). \quad (1)$$

- Example: The *Breast* dataset contains n = 151 samples with d = 54675 gene expressions each. The goal is to classify these data into 6 classes corresponding to breast cancer subtypes and normal tissues.



Better classification in comparison to a 1-NN over the same dataset



Recovery of the clustering from the transport plan

Collas, Antoine, Titouan Vayer, Rémi Flamary, and Arnaud Breloy. "Entropic Wasserstein Component Analysis." arXiv.org, March 9, 2023. https://arxiv.org/abs/2303.05119v1.

Advanced: Loadings and PCA via SVD

- Loadings are obtained by multiplying the Eigen vectors by their square root of the eigen values
  - Give the contribution (loading) of each original features to a PC
  - loading matrix = pca.components_.T * sqrt(pca.explained_variance_).
- Principal components are the eigen vectors of the covariance matrix of the data
- Projections of the data on the principal components are also known as *PC scores*

https://scentellegher.github.io/machine-learning/2020/01/27/pca-loadings-sklearn.html
https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca

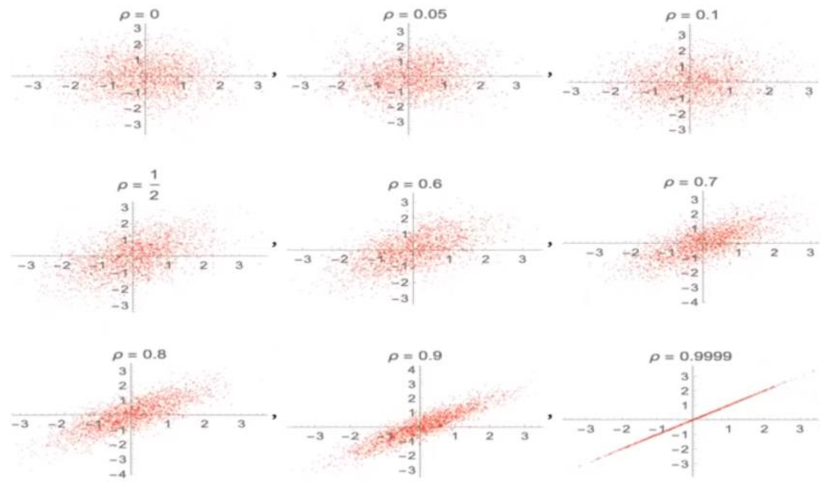
Advanced: PCA in terms of SVD

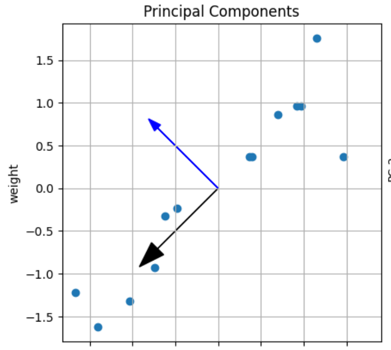
- Given a matrix X
- Standardize it (standard scalar)
- U,S,V = svd(Xn)
  - V are the principal components
    - Z = Xn@V.T will project the data onto the principal components to give projection scores
    - Equivalent to Z = U*S
  - While U itself doesn't measure the correlation between the data and the PCs, you can indirectly assess the relationship between the data and the PCs by examining the weights (coefficients) in U. High absolute values in the coefficients for a particular PC suggest that the data points have a strong relationship with that PC, indicating that the PC captures a significant portion of the variation in the data.

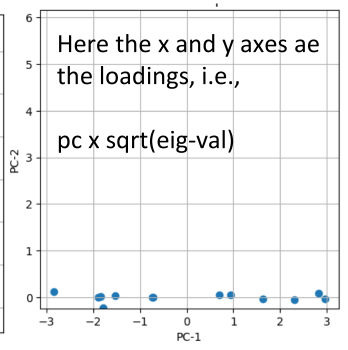
Advanced: A different way of interpreting PCA

- PCA uses correlation to project the data points along principal components ($z = W^T x$)
  - The problem with correlation
    - The correlation of 0.5 is closer to zero than it is to 1.0.
    - Always look at the scatter plot if someone tells you a correlation value
- We need to re-scale the correlation so that when data is plotted in the PCA space it actually aligns with the true notion of information (papers below)
- This will tell us what groups of samples dominate each PC or to what degree each PC "explains" a certain sample
- A heuristic approximation of mutual information is:
    - $\left(MI_{X,V}\right)_{i,j} = -\text{sgn}\left(U_{i,j}\right)\log_2\sqrt{1-U_{i,j}^2}$ where $U_{i,j}$ is the correlation between data point i and PC j.
  - This method serves a dual purpose:
    - it measures how many bits of information are shared between a given sample and a principal component
      - "How much are you willing to bet on Y knowing X" is MI(Y;X)
    - it gives a sense of how well an association between groups of samples and each individual sample and a principal component can be distinguished from chance.
  - MI also used in "Correlation Explanation" methods

```
#re-scaling PCA plot
Xn = StandardScaler().fit_transform(X)
U,S,_ = np.linalg.svd(Xn)
d = Xn.shape[1]
U = U[:,:Xn.shape[1]]
Z = U*S
plt.scatter(Z[:,0],Z[:,1]) #regular PCA plot
Z_mi = -np.sign(U)*0.5*np.log2(1-U**2)#*S/np.sqrt(d-1)
#uncomment last part to get equivalent of loadings
plt.scatter(Z_mi[:,0],Z_mi[:,1]) #with mi scaling
```

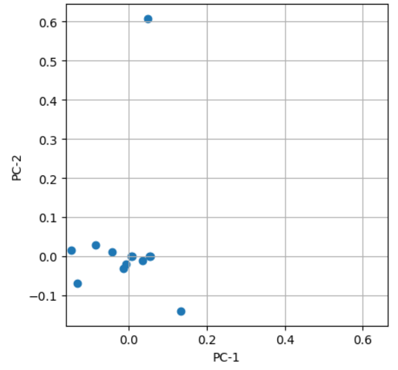


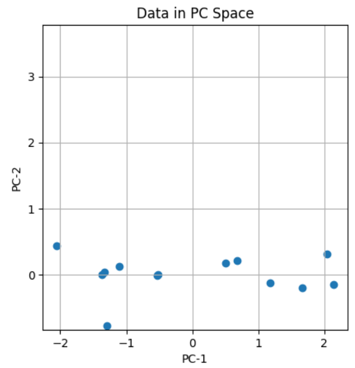




Here the x and y axes ae the loadings, i.e.,

pc x sqrt(eig-val)





https://github.com/foxtrotmike/PCA-Tutorial/blob/master/pca-lagrange.ipynb

With Mutual Information scaling indicating shared bitwise information (augmented by the sign) between samples and principal components.

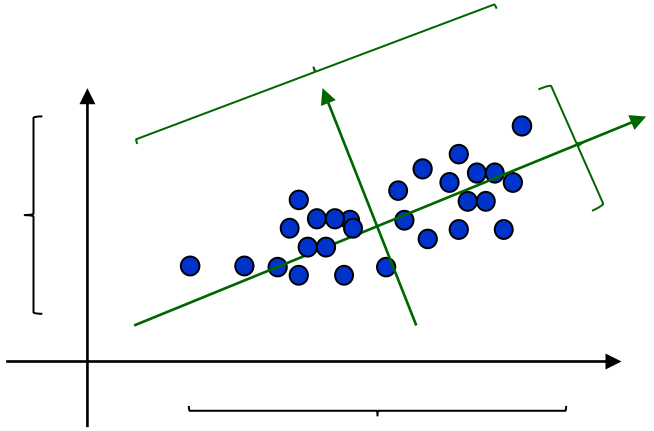
Video: https://youtu.be/o9Ac85xdjE4?t=525

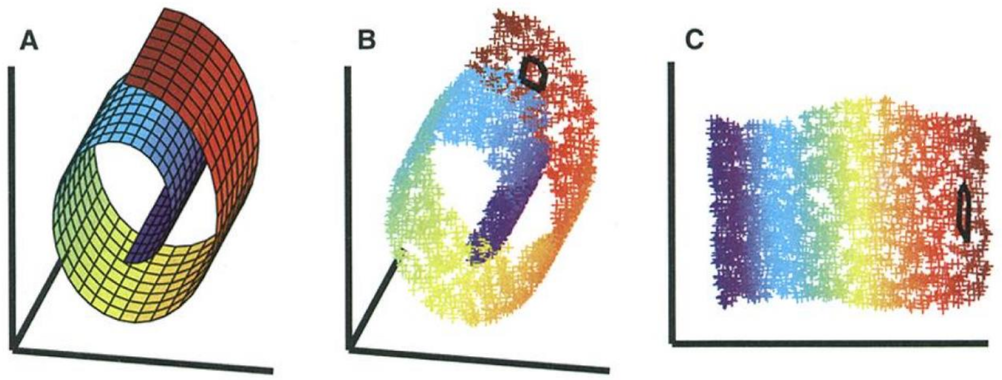
Taleb, Nassim Nicholas, Pierre Zalloua, Khaled Elbassioni, Andreas Henschel, and Daniel E. Platt. "Informational Rescaling of PCA Maps with Application to Genetic Distance." arXiv.org, March 14, 2023. https://arxiv.org/abs/2303.12654v1.

Platt, Daniel E., Hovig Artinian, Francis Mouzaya, Wissam Khalil, Francois G. Kamar, Elizabeth Matisoo-Smith, Francesc Calafell, Nassim Nicolas Taleb, and Pierre Zalloua. "Autosomal Genetics and Y-Chromosome Haplogroup L1b-M317 Reveal Mount Lebanon Maronites as a Persistently Non-Emigrating Population." *European Journal of Human Genetics* 29, no. 4 (April 2021): 581–92. https://doi.org/10.1038/s41431-020-00765-x.

Dimensionality Reduction: Important Conceptual Note

- A number of variables can be correlated in real datasets
- Thus, the effective dimensionality of the dataset can be lower than what you see in terms of number of features
- Thus, data lives on a "manifold"
- That is why dimensionality reduction models help
- **These manifolds may not be linear**
- **Taxonomy of dimensionality reduction methods**
  - **Linear**: PCA, LDA, Canonical Component Analysis
  - **Nonlinear**
    - **Manifold Learning** Approaches: uncovering the low-dimensional structure hidden within high-dimensional data
      - Kernel PCA, MDS, Locally linear embeddings, t-SNE …
    - **Probabilistic Approaches**
      - ICA
    - **Topological Data Analysis**: Preservation of the topological structure of the data
      - UMAP
  - Supervised Dimensionality Reduction

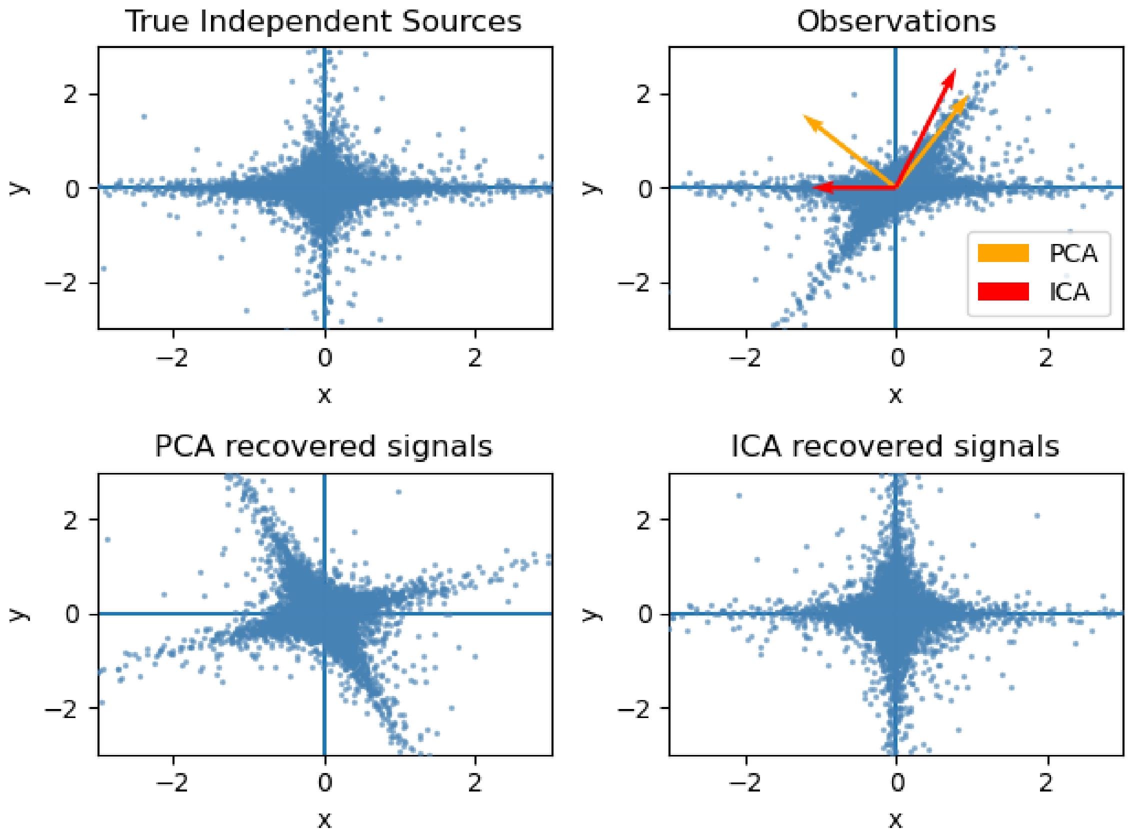




Roweis, Sam T., and Lawrence K. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding." *Science* 290, no. 5500 (December 22, 2000): 2323–26. https://doi.org/10.1126/science.290.5500.2323.

Independent Component Analysis

- Being linear PCA produces orthogonal or un-correlated components
- ICA generalizes this idea by identifying projection directions with minimal "mutual information", i.e., knowing one doesn't tell anything about the other
- Used in signal analysis and "blind source separation"



Code: https://scikit-learn.org/stable/auto_examples/decomposition/plot_ica_vs_pca.html

Manifold Learning: MDS

- Multi-dimensional Scaling
- Representation
  - Input
    - Pairwise Distance matrix between examples based on d features d_{ij}
  - Output
    - A k-dimensional representation of each example: $\boldsymbol{x}_i$
- Evaluation
  - Reduce the number of dimensions in the data while preserving pairwise distances between points

$$\underset{x_1,...,x_M}{\operatorname{argmin}} \sum_{i<j} (\|x_i - x_j\| - d_{i,j})^2$$

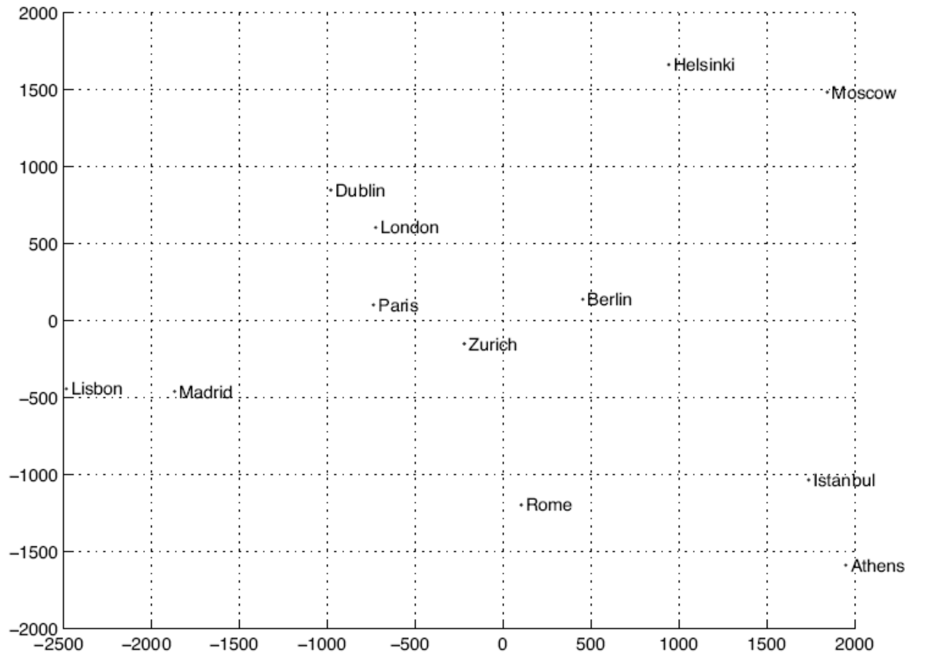
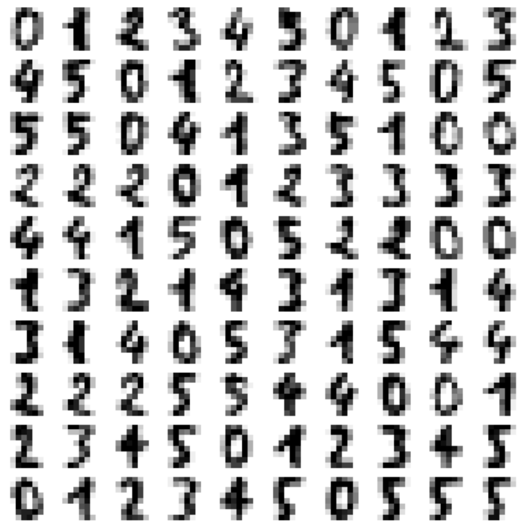


Figure 6.6 Map of Europe drawn by MDS. Pairwise road travel distances between these cities are given as input, and MDS places them in two dimensions such that these distances are preserved as well as possible.

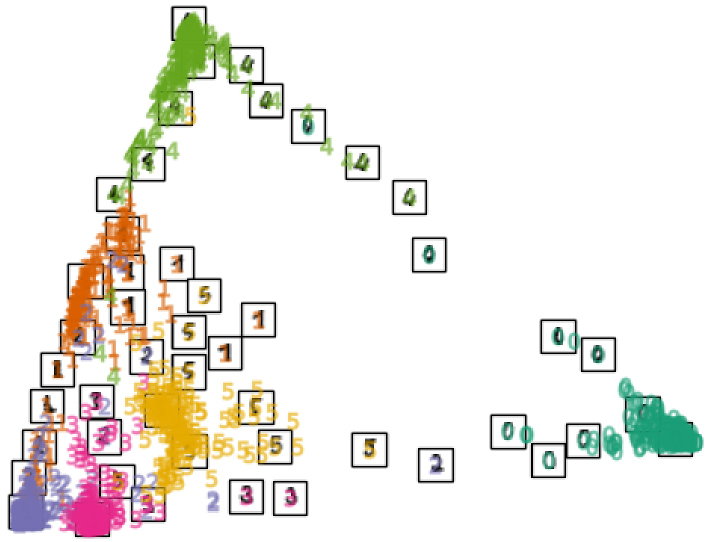
[Read More: Wikipedia](#)
[MDS in Sklearn](#)

Manifold Learning Based Methods

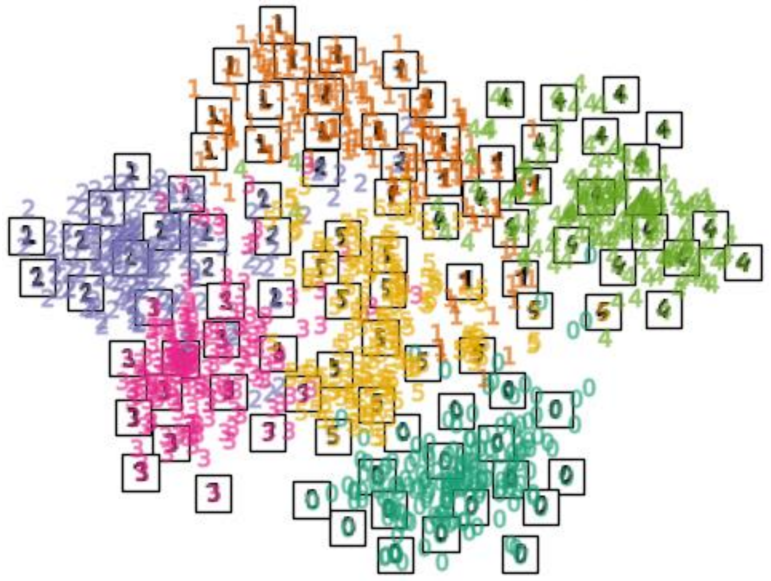
A selection from the 64-dimensional digits dataset



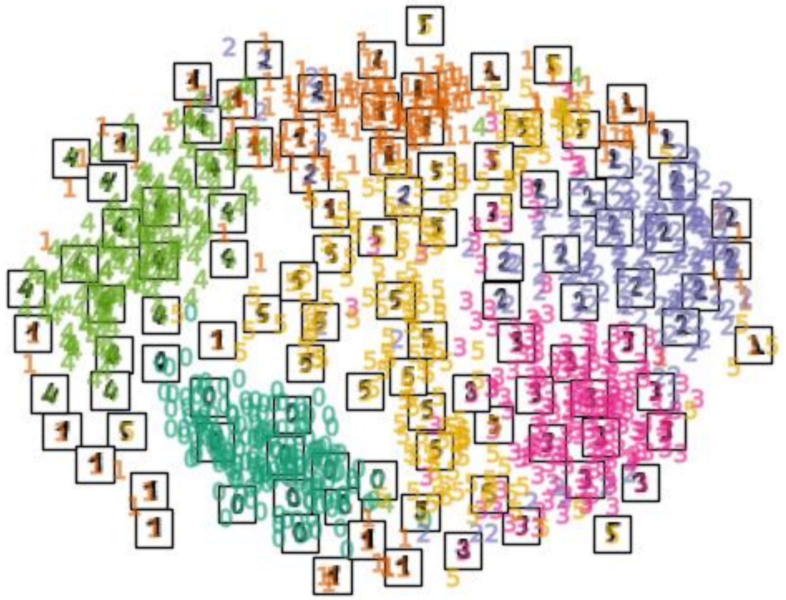
Spectral embedding (time 0.175s)



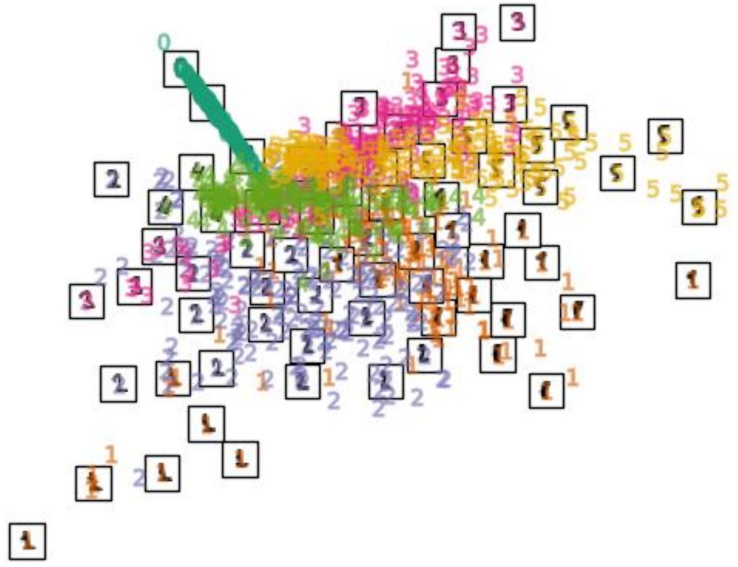
NCA embedding (time 2.629s)



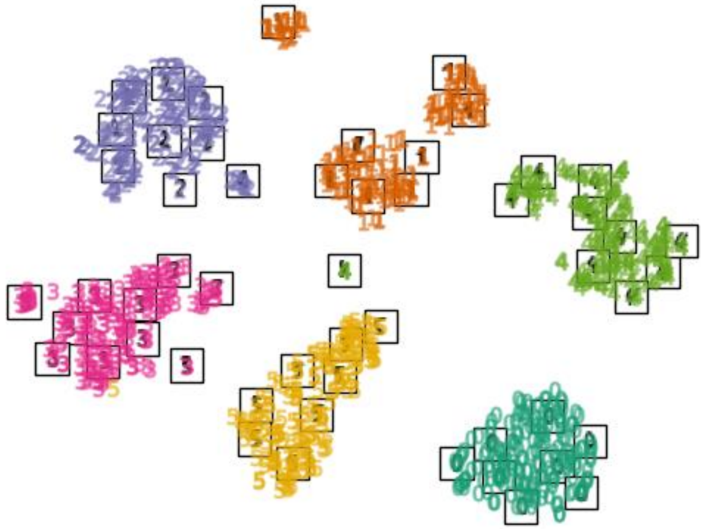
MDS embedding (time 3.386s)



Standard LLE embedding (time 0.161s)



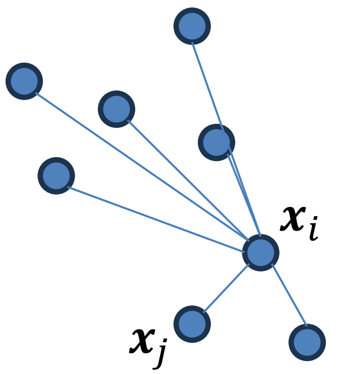
t-SNE embedding (time 2.585s)

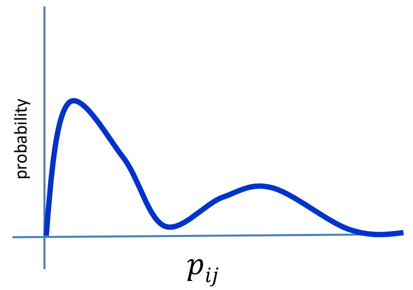


Code: https://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html
[Locality constrained linear coding]()

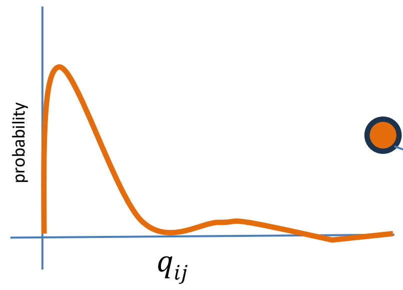
Manifold Learning: How does t-SNE work?

- t-distributed stochastic neighbor embedding
  - Fundamental Idea: The “local” probability distributions of similarity between points before and after projection should be the same
- Representation
  - Input: d-dimensional feature representation $\boldsymbol{x}_i$
  - Output: k-dimensional feature representation $\boldsymbol{y}_i$
- Evaluation
  - Reduce data dimensionality while preserving the local probability distribution of the data
- Practical Notes
  - It preservers the “local structure”
  - Hyperparameters: Number of dimensions, “Perplexity / Spread of neighborhood”
    - Sklearn implementation

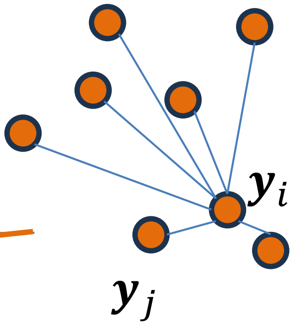




Probability of similarity between points before projection p_{ij}



Probability of similarity between points after projection



For $i \neq j$, define

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)} \quad q_{ij} = \frac{(1+\|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_k \sum_{l\neq k}(1+\|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

Notice each point has its own spread

Evaluation: min $\mathrm{KL}\,(P\,\|\,Q) = \sum_{i\neq j} p_{ij}\log\frac{p_{ij}}{q_{ij}}$

Probability of similarity between points before projection

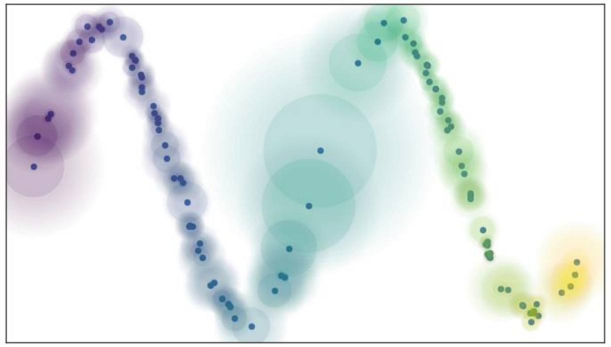
Probability of similarity between points after projection

Kullback–Leibler (KL) divergence (aka relative entropy) Measuring how much different one probability distribution is from another

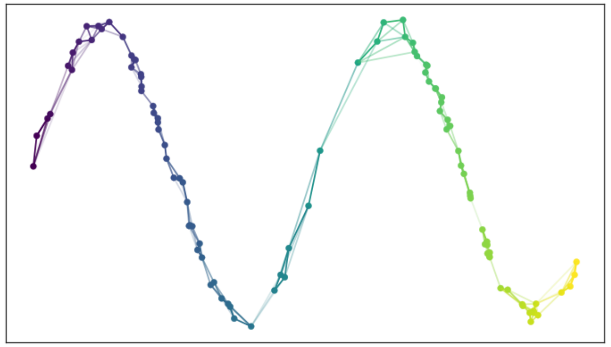
Additional resources: Wikipedia Page
Very good tutorial: https://towardsdatascience.com/t-sne-clearly-explained-d84c537f53a

Topological Data Analysis: UMAP

- Uniform Manifold Approximation and Projection
  - Basic Principle: Preserve the topological structure (edges, “faces” etc) of the data during dimensionality reduction
- Representation
  - Input: d-dimensional feature representation $\boldsymbol{x}_i$
  - Output: k-dimensional feature representation $\boldsymbol{y}_i$
- Evaluation
  - Minimize the cross-entropy between two “fuzzy simplical set” representations of the original data and the reduced dimensionality data
  - It constructs a high-dimensional graph representing the manifold by connecting each point to its nearest neighbors, then optimizes a low-dimensional graph to be as structurally similar as possible, using cross-entropy between fuzzy set representations of the graphs to approximate this optimization.
- Practical Notes
  - Unlike t-SNE which is focused on preservation of local structure, UMAP can balance preservation of global and local structures
  - Hyperparameters: Number of neighbors in the graph, minimum distance (more robust to hyperparameter values in comparison to t-SNE)
  - Both Unsupervised and Supervised



Data occurring on a manifold. A “fuzzy open set” representation is developed in which the local influence of each point extends to at least its immediate neighbor.



A graph is developed in which the nodes are connected based on the local topological structure of the data and the weighted edges show the probability of connection between nodes

Minimize Cross Entropy between graph representations in high and low dimensional spaces

Intuitively: Minimize much information is lost when using one distribution (low-dimensional one in this case) to approximate another (high-dimensional one)

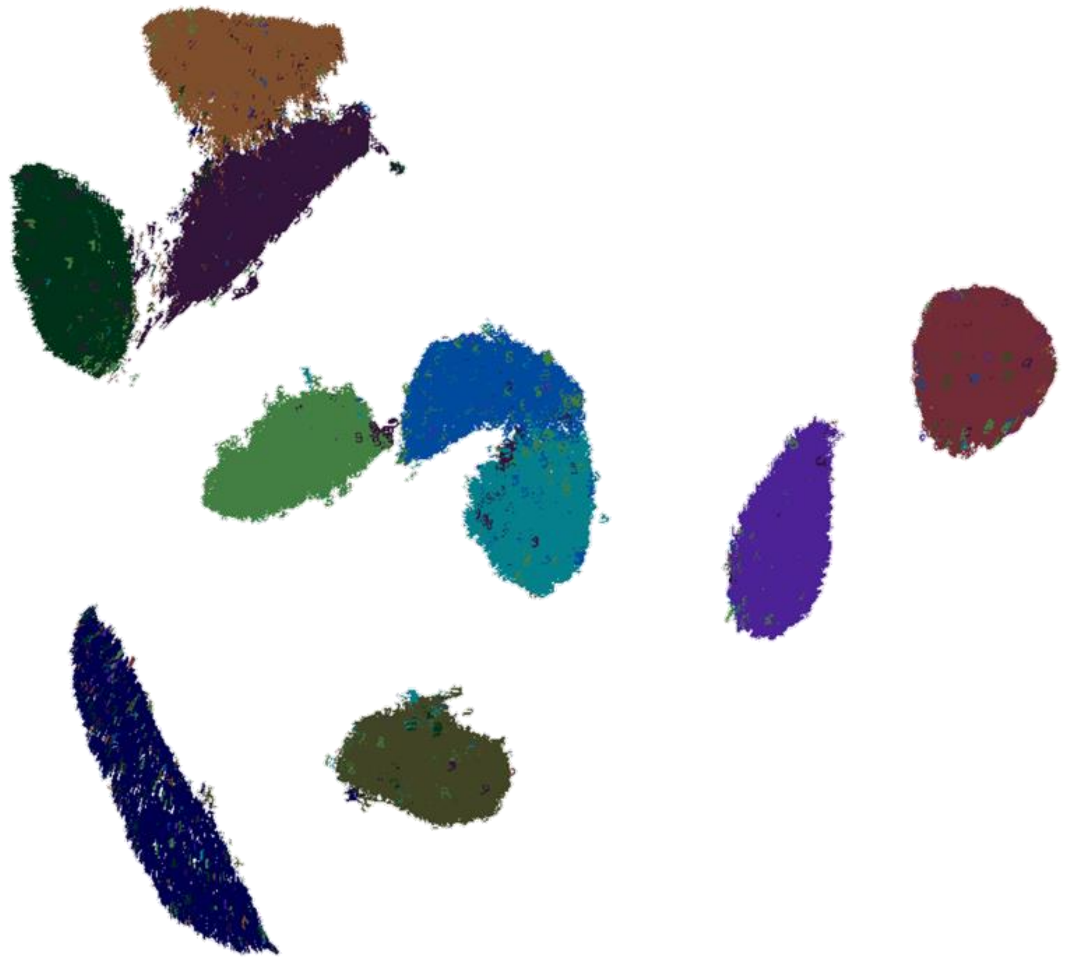
$$C(G_{high}, G_{low}) = \sum_{i,j}[G_{high}(i,j)\log(\frac{G_{high}(i,j)}{G_{low}(i,j)}) + (1 - G_{high}(i,j))\log(\frac{1-G_{high}(i,j)}{1-G_{low}(i,j)})]$$

where:

- $G_{high}(i,j)$ is the probability of connection between points i and j in the high-dimensional space, and
- $G_{low}(i,j)$ is the probability of connection between points i and j in the low-dimensional space.

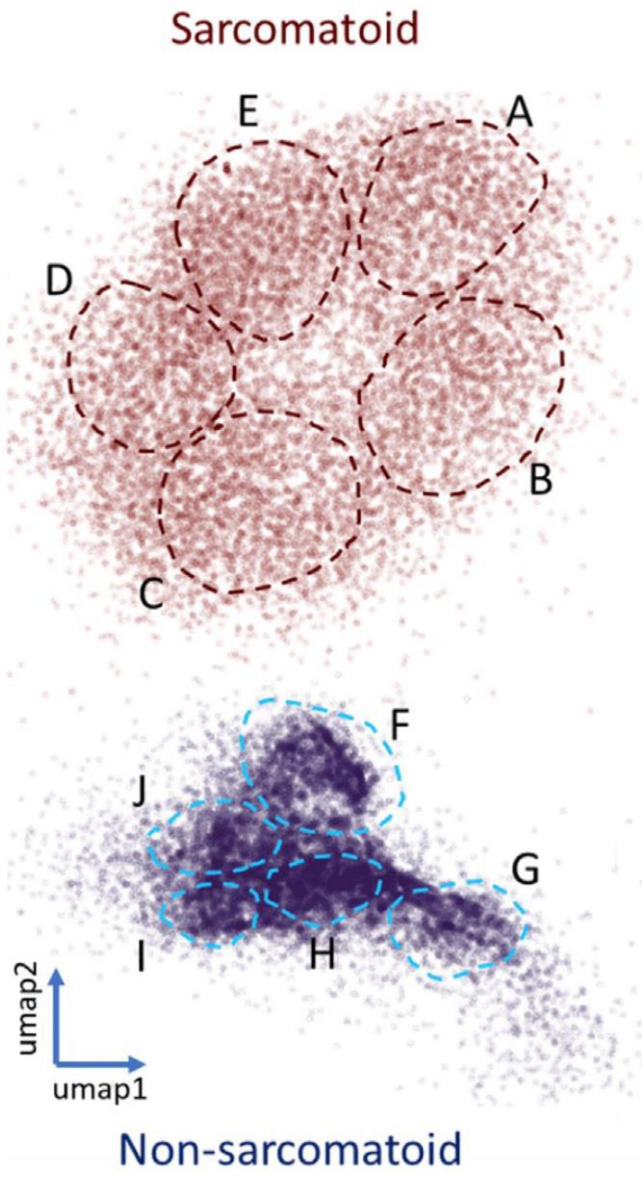
https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

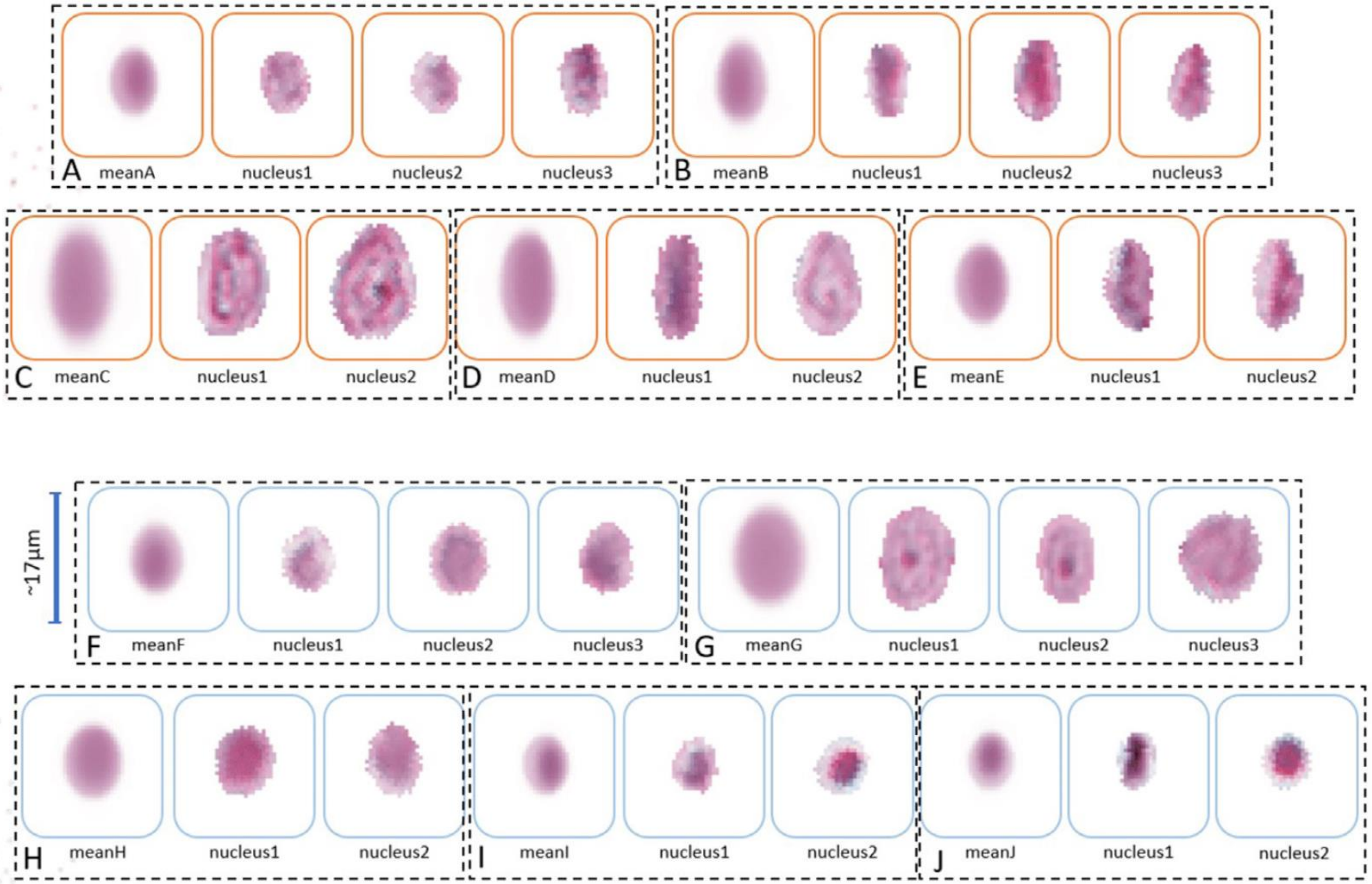
UMAP Over MNIST



https://grantcuster.github.io/umap-explorer/

UMAP of Mesothelioma cells



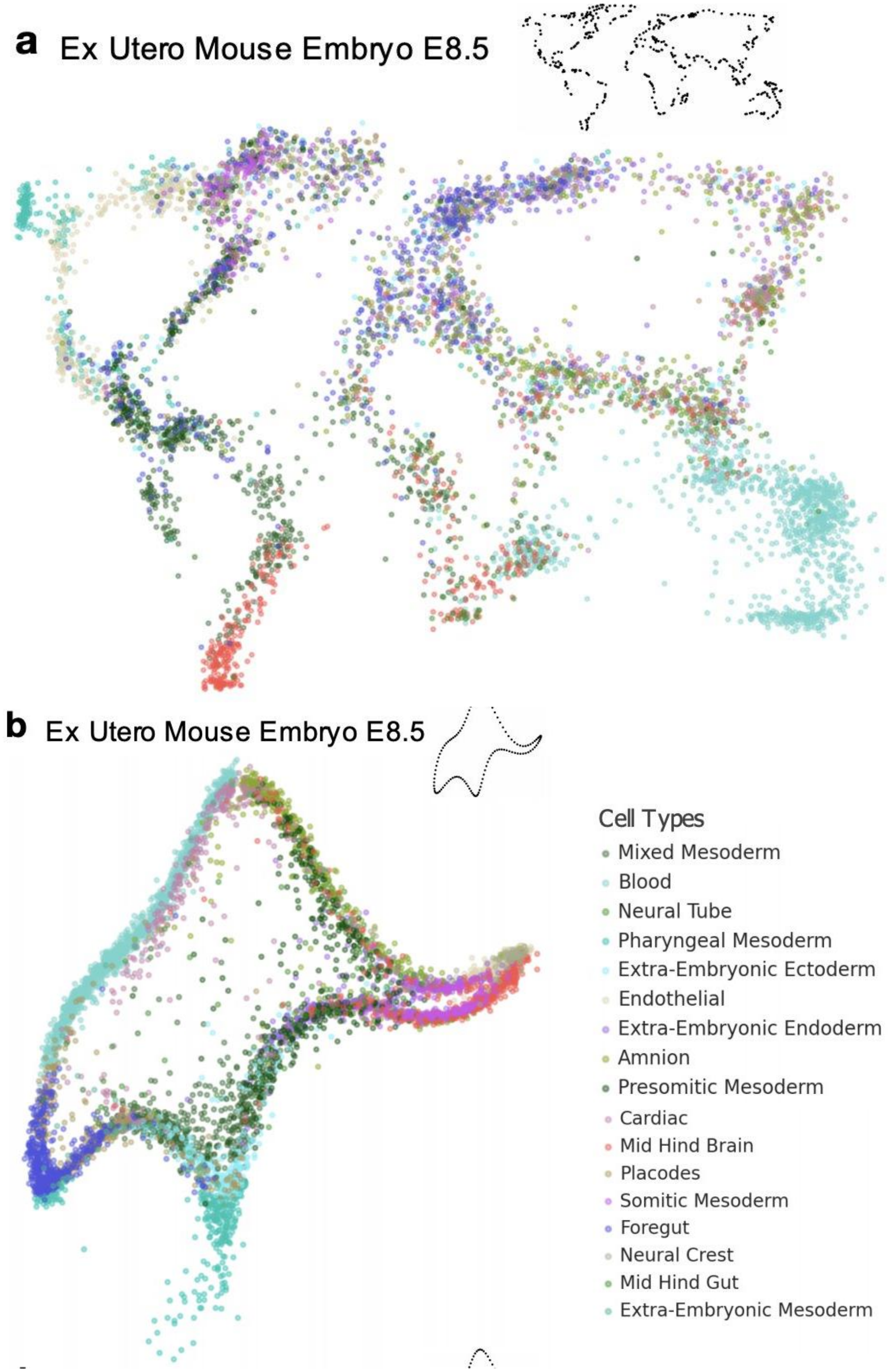

A meanA nucleus1 nucleus2 nucleus3
B meanB nucleus1 nucleus2 nucleus3
C meanC nucleus1 nucleus2
D meanD nucleus1 nucleus2
E meanE nucleus1 nucleus2
~17μm
F meanF nucleus1 nucleus2 nucleus3
G meanG nucleus1 nucleus2 nucleus3
H meanH nucleus1 nucleus2
I meanI nucleus1 nucleus2
J meanJ nucleus1 nucleus2

Eastwood, Mark, …, Fayyaz Minhas. "MesoGraph: Automatic Profiling of Mesothelioma Subtypes from Histological Images." *Cell Reports Medicine* 0, no. 0 (October 9, 2023). https://doi.org/10.1016/j.xcrm.2023.101226.

Limitations of dimensionality reduction

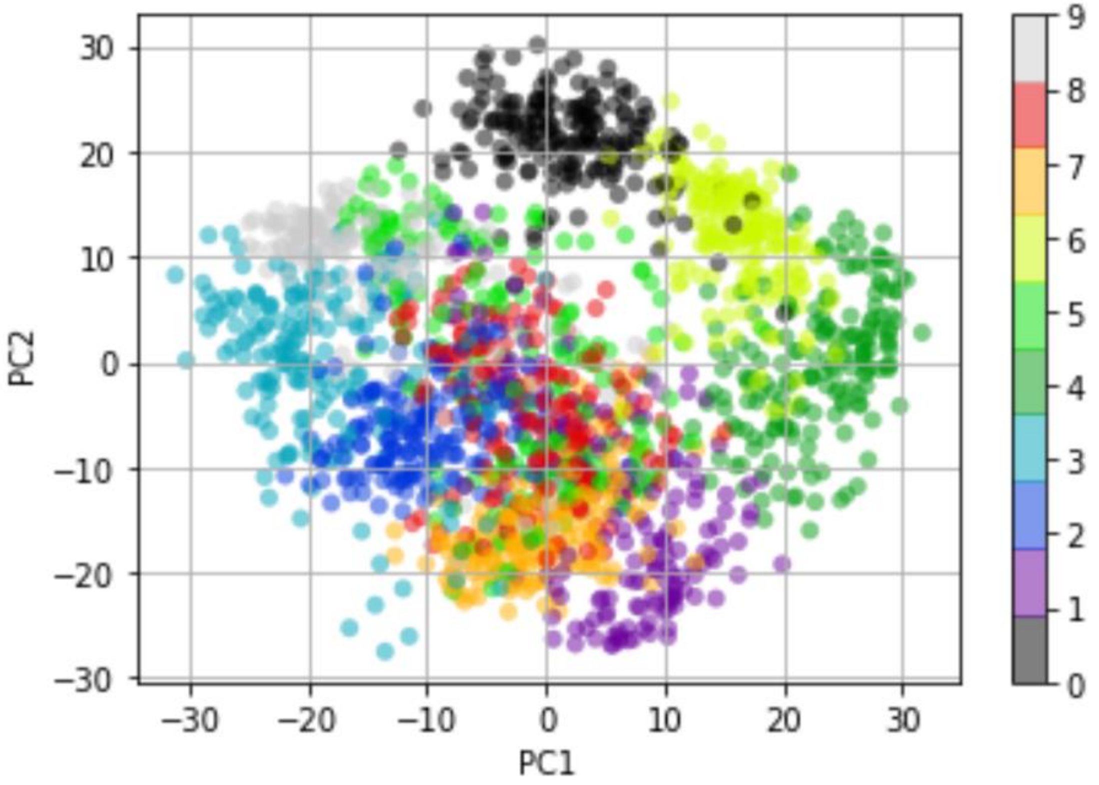
- All dimensionality reduction methods introduce distortions of some sorts
  - Shown on the right is the same 20,000 dimensional data reduced to 2-dimensions using UMAP and shown as a world map and as an elephant
- If I don't see something in reduced dimensions
  - It may still exist
- If I do see something in reduced dimensions
  - It may not be there
- Further confirmatory/validation checks are needed
- Tweet: https://twitter.com/lpachter/status/1431325969411821572

Chari, Tara, and Lior Pachter. "The Specious Art of Single-Cell Genomics." PLOS Computational Biology 19, no. 8 (August 17, 2023): e1011288. https://doi.org/10.1371/journal.pcbi.1011288.



Supervised Dimensionality Reduction

- Most dimensionality reduction approaches are un-supervised in that they do not use any “class” information or other supervisory signal in their projection
- However, there are supervised dimensionality reduction methods but they are not used that frequently
- We can use PCA to produce an “embedding” of examples over training data which can then be used over test data for classification but the PCA itself doesn’t use labels
- **Important Note:** Never use test labels or any other “target” information (directly or indirectly)



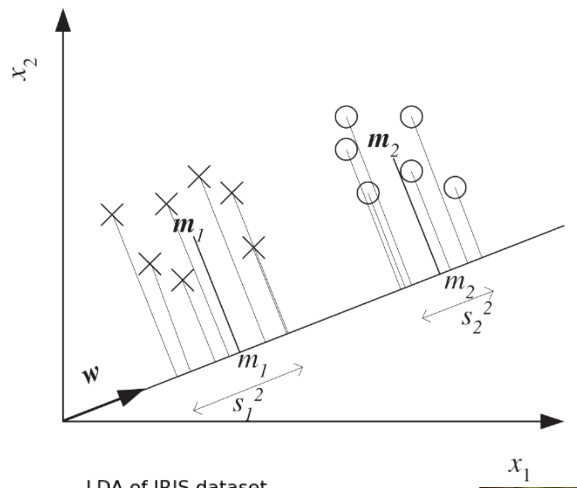
Linear Discriminant Analysis

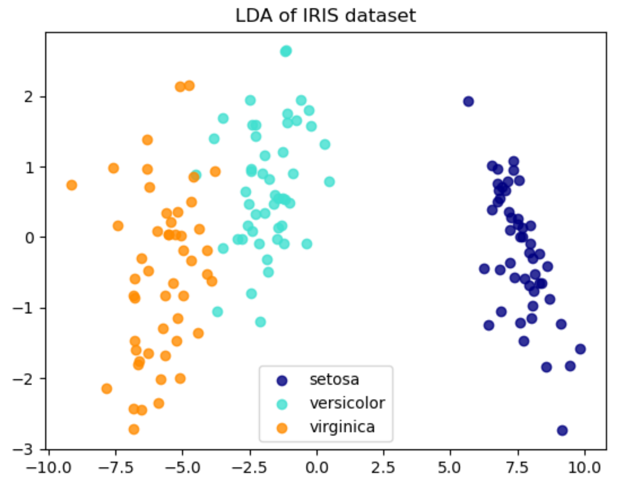
- Goal/Principle: Find the direction vector that maximally separate examples of two different classes while projecting data of each class as compactly as possible
- Representation
  - Linear: $\mathrm{z} = \mathrm{w}^{\mathrm{T}}\mathrm{x}$
- Evaluation

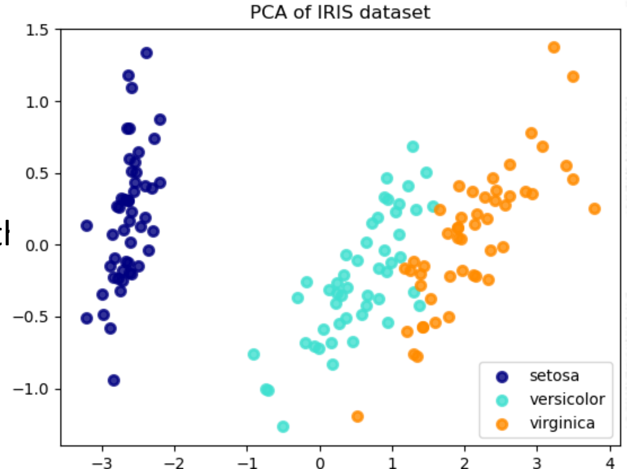
$$\max_{\boldsymbol{w}} S = \frac{difference\ between\ classes\ after\ projection}{variance\ within\ classes\ after\ projection}$$
$$= \frac{(\boldsymbol{w}^T\boldsymbol{m}_1 - \boldsymbol{w}^T\boldsymbol{m}_2)}{\boldsymbol{s}_1^2 + \boldsymbol{s}_2^2} = \frac{(\boldsymbol{w}^T\boldsymbol{m}_1 - \boldsymbol{w}^T\boldsymbol{m}_2)}{\boldsymbol{w}^T\boldsymbol{C}_1\boldsymbol{w} + \boldsymbol{w}^T\boldsymbol{C}_2\boldsymbol{w}}$$

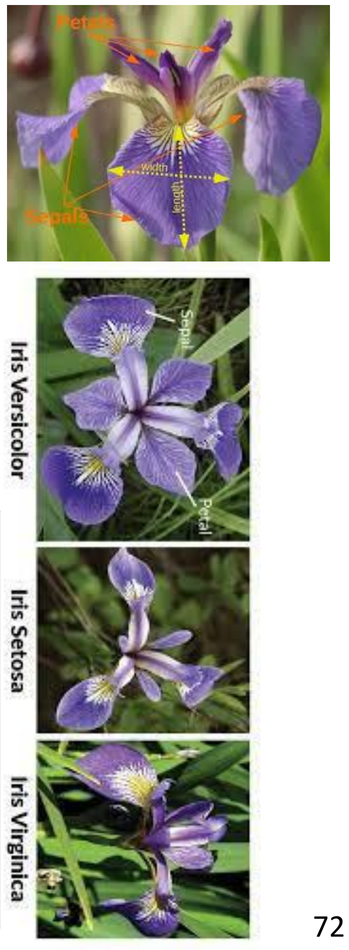
- Multi-class LDA produces a weight vector that is (Number of classes -1) in dimensionality
  - In essence, this tells us the direction of maximal separability under the assumption th the data is gaussian distributed

https://en.wikipedia.org/wiki/Linear_discriminant_analysis
https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html



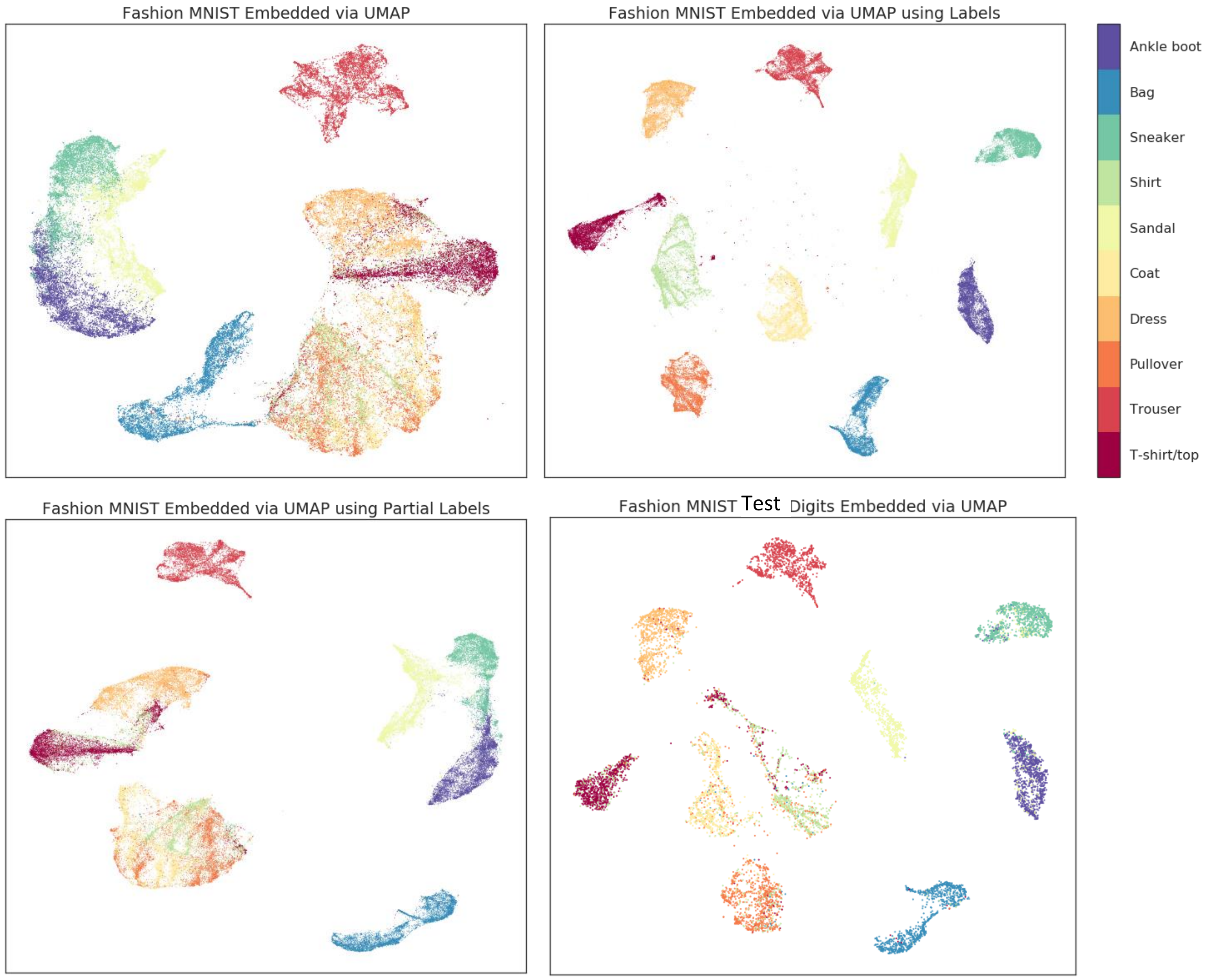






Supervised UMAP

- Guides the embedding process so that the data points with the same label are closer to each other in the reduced space, enhancing the separation between different classes.
- Can be used for semi-supervised or transductive learning in which label information is only available for a few examples
- Can also be used as a feature extractor

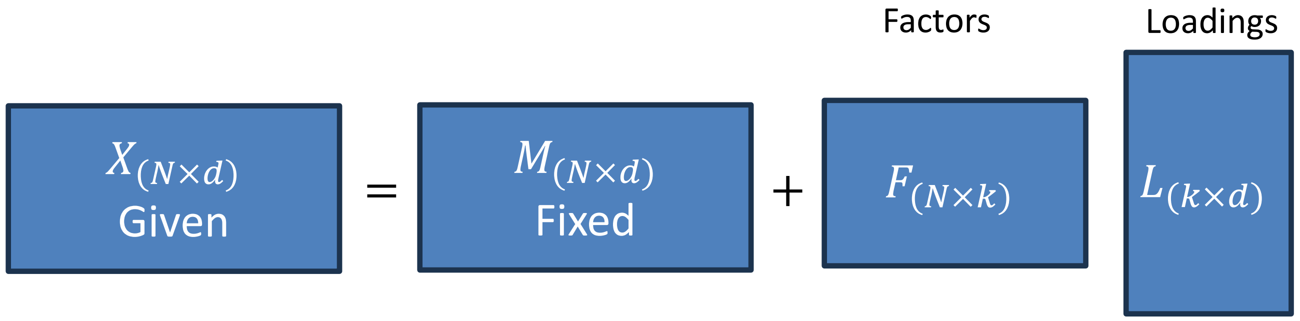


https://umap-learn.readthedocs.io/en/latest/supervised.html

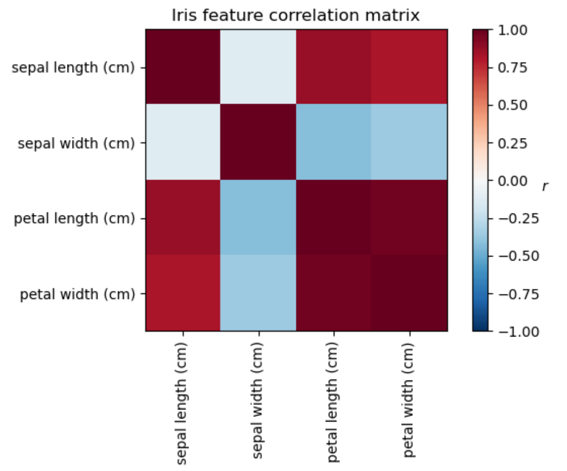
Other Models: Factor Analysis

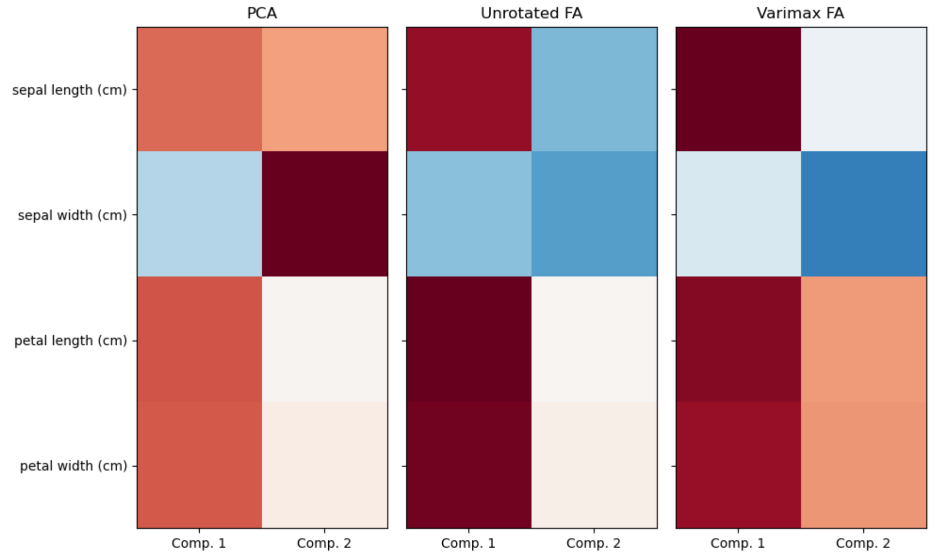
- Goal
  - Describe the variability among observed variables in terms of a potentially lower number of unobserved (latent) variables called factors to reveal the underlying structure of the data
- Representation
  - In matrix form: $X_{(N\times d)} = M + F_{(N\times k)}L_{(k\times d)} + \epsilon$
  - Express the (centralized) observations in terms of its
- Evaluation (No orthogonality requirement)

$$\min_{(F,L)}\left\|X_{(N\times d)} - (M + F_{(N\times k)}L_{(k\times d)})\right\|$$



https://scikit-learn.org/stable/auto_examples/decomposition/plot_varimax_fa.html





$$x_{i,m} - \mu_i = l_{i,1}f_{1,m} + \cdots + l_{i,k}f_{k,m} + \varepsilon_{i,m}$$

where

- $x_{i,m}$ is the value of the ith observation of the mth individual,
- μ_i is the observation mean for the ith observation,
- $l_{i,j}$ is the loading for the ith observation of the jth factor,
- $f_{j,m}$ is the value of the jth factor of the mth individual, and
- $\varepsilon_{i,m}$ is the (i,m)th *unobserved stochastic error term* with mean zero and finite variance.

Other Methods: Topic Models

- Goal
  - Can we uncover what are the “topics” in a given dataset
    - From document modelling: How can we describe each document in terms of its topics
  - Discover latent semantic structure in data
- Non-Negative Matrix Factorization
  - Break down a given matrix in terms of non-negative factors
  - Representation (assuming centralized data)

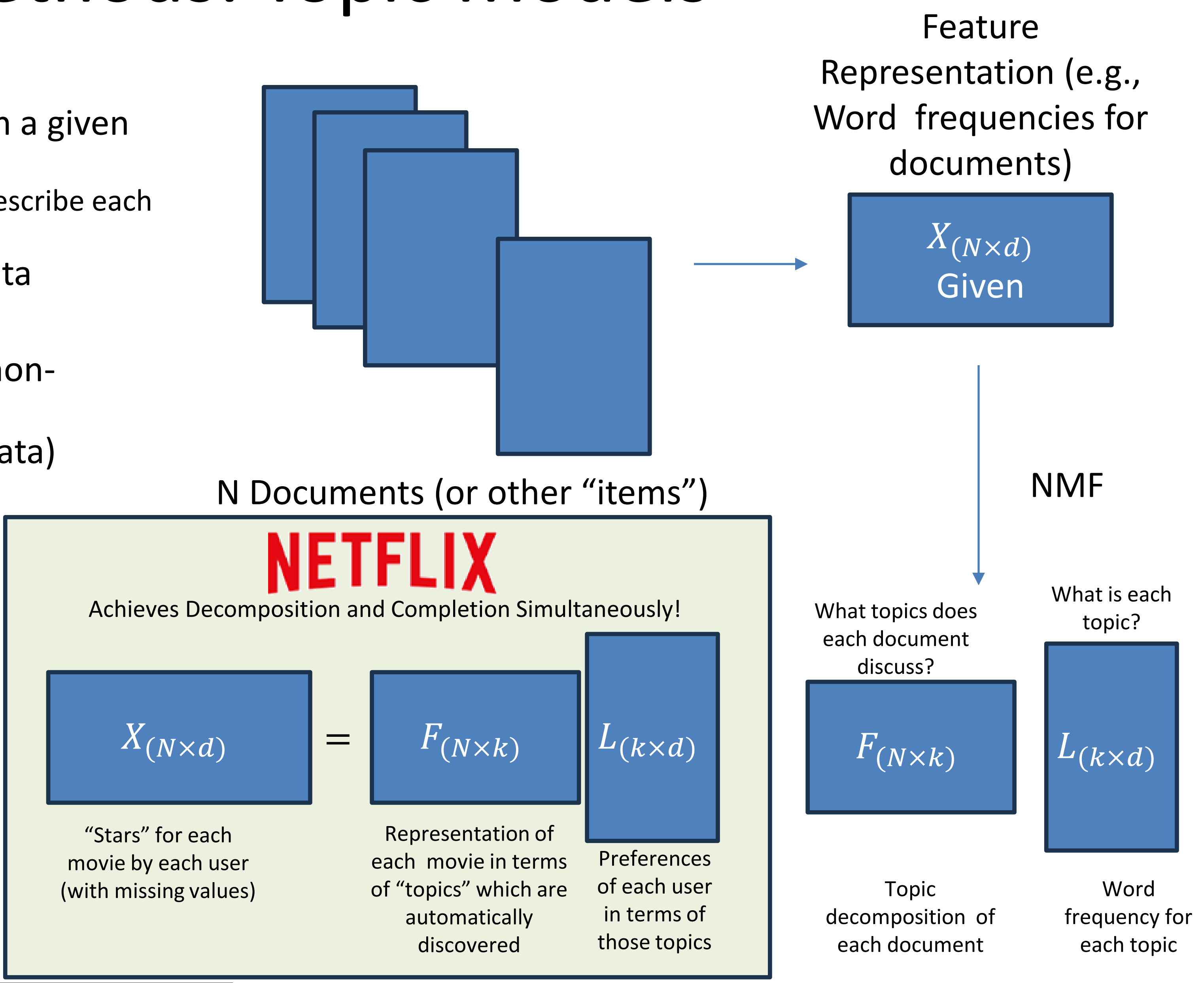
$$X_{(N\times d)} = F_{(N\times k)}L_{(k\times d)}$$

  - Evaluation (assuming $M = 0$)

$$\min_{(F,L)} \left\| X_{(N\times d)} - F_{(N\times k)}L_{(k\times d)} \right\|$$

s.th. $F \geq 0, L \geq 0$

  - Other Approaches:
    - Latent Dirichlet Allocation (LDA)
    - Correlation Explanation

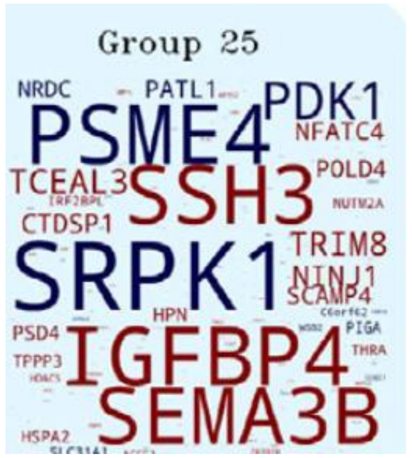


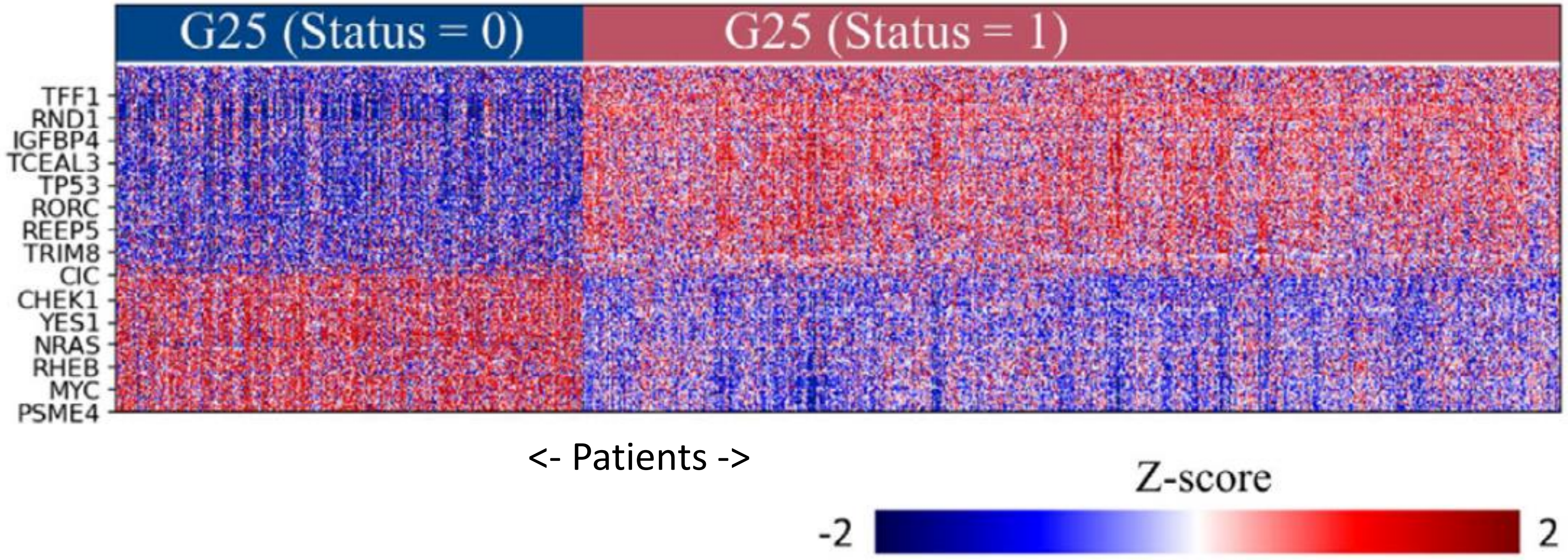
Feature Representation (e.g., Word frequencies for documents)
$X_{(N\times d)}$ Given
N Documents (or other “items”)
NMF
What topics does each document discuss?
What is each topic?
$F_{(N\times k)}$
$L_{(k\times d)}$
Topic decomposition of each document
Word frequency for each topic
NETFLIX
Achieves Decomposition and Completion Simultaneously!
$X_{(N\times d)}$ = $F_{(N\times k)}$ $L_{(k\times d)}$
“Stars” for each movie by each user (with missing values)
Representation of each movie in terms of “topics” which are automatically discovered
Preferences of each user in terms of those topics

https://en.wikipedia.org/wiki/Netflix_Prize

Application

- What are the underlying patterns of gene expression of breast cancer patients and can those be predicted from whose slide imaging?
  - Used CoRex to discover such patterns converting 20,000 genes to 200 binary topics!

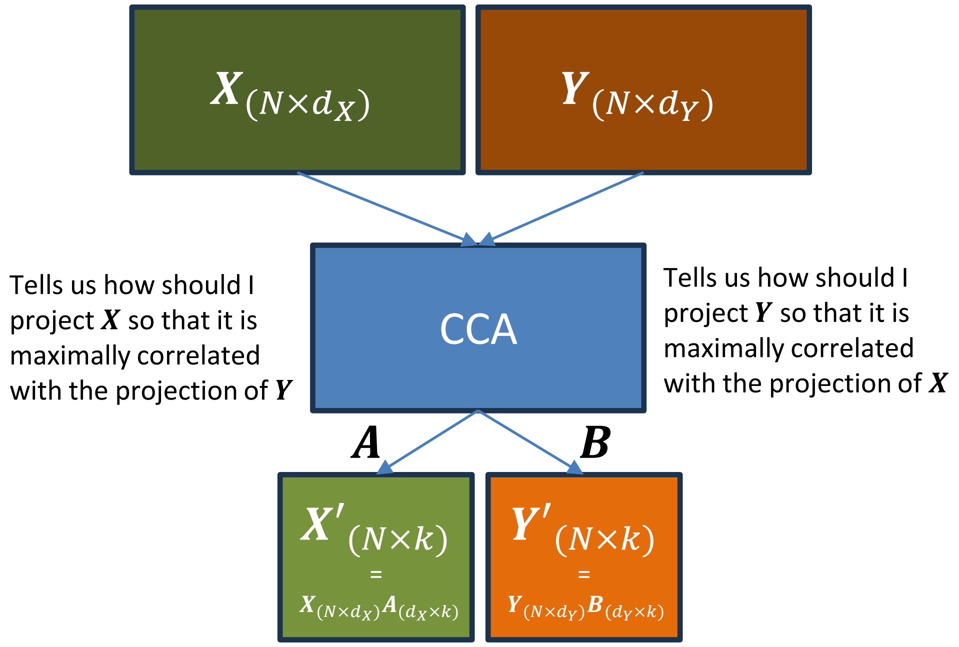




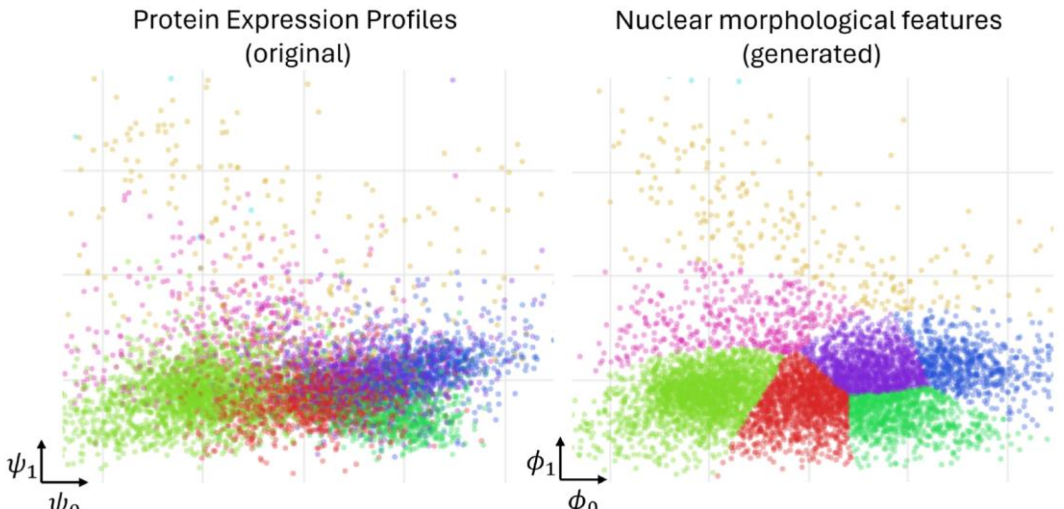
Dawood, Muhammad, Mark Eastwood, Mostafa Jahanifar, Lawrence Young, Asa Ben-Hur, Kim Branson, Louise Jones, Nasir Rajpoot, and Fayyaz ul Amir Afsar Minhas. “Cross-Linking Breast Tumor Transcriptomic States and Tissue Histology.” *Cell Reports Medicine* 4, no. 12 (December 19, 2023). https://doi.org/10.1016/j.xcrm.2023.101313.

Cross Decomposition: Canonical Correlation Analysis

- Goal: Uncover the degree of association between two feature spaces?
  - Assume we have two sets of features for N examples and we would like to identify the direction(s) for each set such that the if the data are projected onto the corresponding directions, the resulting vectors are maximally correlated
- Representation (for single dimension reduction, k=1)
  - Input Data Matrices: $\boldsymbol{X}_{(N\times d_x)}, \boldsymbol{Y}_{(N\times d_y)}$
  - Output:
    - $\boldsymbol{X}' = \boldsymbol{X}_{(N\times d_x)}\boldsymbol{a}_{(d_x\times 1)}, \boldsymbol{Y}' = \boldsymbol{Y}_{(N\times d_y)}\boldsymbol{b}_{(d_y\times 1)}$
- Evaluation
  - Maximize Correlation between projected dimensions
  - $\boldsymbol{max}_{(\boldsymbol{a},\boldsymbol{b})}\ \boldsymbol{corr}(\boldsymbol{X}', \boldsymbol{Y}')$ (subject to some normalization constraints)
- Implementation
  - https://scikit-learn.org/0.16/modules/generated/sklearn.cross_decomposition.CCA.html



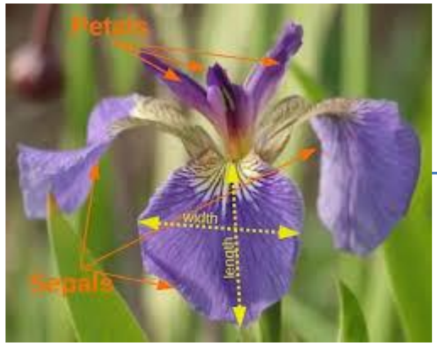
$\boldsymbol{X}_{(N\times d_X)}$
$\boldsymbol{Y}_{(N\times d_Y)}$
Tells us how should I project $\boldsymbol{X}$ so that it is maximally correlated with the projection of $\boldsymbol{Y}$
CCA
Tells us how should I project $\boldsymbol{Y}$ so that it is maximally correlated with the projection of $\boldsymbol{X}$
$\boldsymbol{A}$
$\boldsymbol{B}$
$\boldsymbol{X}'_{(N\times k)} = \boldsymbol{X}_{(N\times d_X)}\boldsymbol{A}_{(d_X\times k)}$
$\boldsymbol{Y}'_{(N\times k)} = \boldsymbol{Y}_{(N\times d_Y)}\boldsymbol{B}_{(d_Y\times k)}$

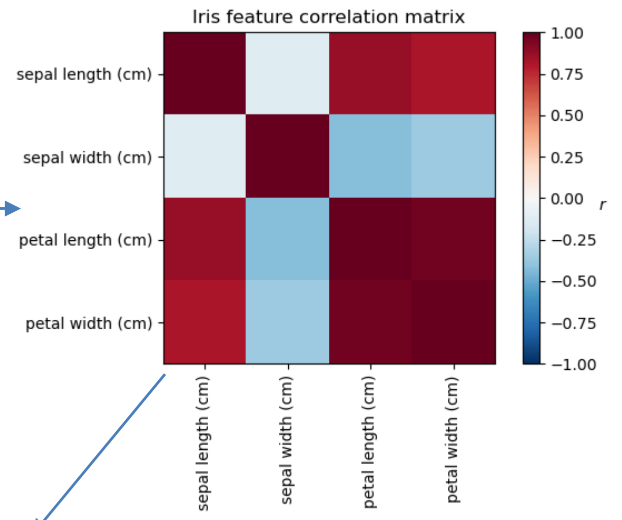


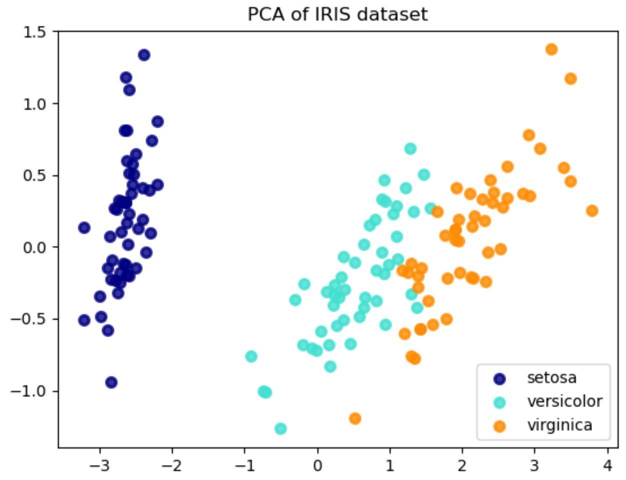
Andrew, Galen, Raman Arora, Jeff Bilmes, and Karen Livescu. "Deep Canonical Correlation Analysis." In Proceedings of the 30th International Conference on Machine Learning, 1247–55. PMLR, 2013. https://proceedings.mlr.press/v28/andrew13.html.
Benton, Adrian, Huda Khayrallah, Biman Gujral, Dee Ann Reisinger, Sheng Zhang, and Raman Arora. "Deep Generalized Canonical Correlation Analysis." In Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), edited by Isabelle Augenstein, Spandana Gella, Sebastian Ruder, Katharina Kann, Burcu Can, Johannes Welbl, Alexis Conneau, Xiang Ren, and Marek Rei, 1–6. Florence, Italy: Association for Computational Linguistics, 2019. https://doi.org/10.18653/v1/W19-4301.

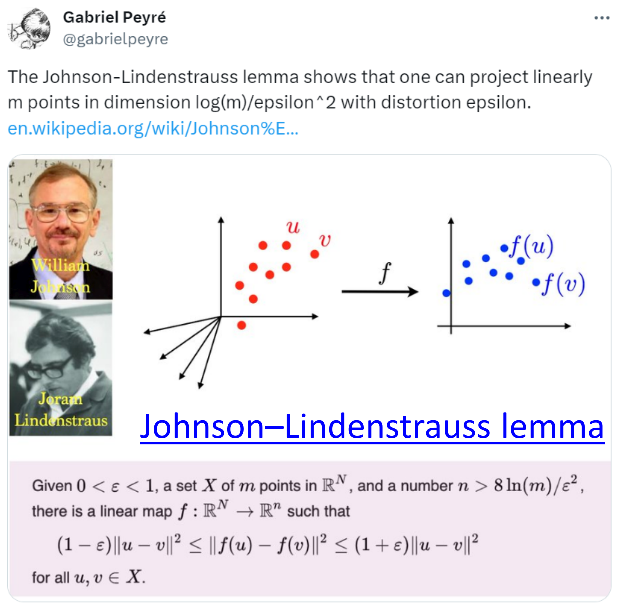
Why dimensionality reduction is even possible!

- Variables in data may carry information about each other
  - Why?
    - Because the underlying physical or information generating source giving rise to examples is the same
- This introduces statistical dependence between our variables and gives structure (manifold or topological) to the data
- Dimensionality reduction approaches exploit such "Mutual Information" and help us uncover the hidden structure and to make sense of our data by identifying (a smaller number of) directions or latent variables that can "explain away" our data








Gabriel Peyré
@gabrielpeyre

The Johnson-Lindenstrauss lemma shows that one can project linearly m points in dimension log(m)/epsilon^2 with distortion epsilon.
en.wikipedia.org/wiki/Johnson%E...

William Johnson
Joram Lindenstrauss

Johnson–Lindenstrauss lemma

Given $0 < \varepsilon < 1$, a set X of m points in $\mathbb{R}^N$, and a number $n > 8\ln(m)/\varepsilon^2$, there is a linear map $f : \mathbb{R}^N \to \mathbb{R}^n$ such that

$$(1-\varepsilon)\|u-v\|^2 \le \|f(u)-f(v)\|^2 \le (1+\varepsilon)\|u-v\|^2$$

for all $u, v \in X$.

End of Lecture

We want to make a machine that will be proud of us.

- Danny Hillis