CSC 311: Introduction to Machine Learning

Lecture 9 - PCA, Matrix Completion, Autoencoders

Rahul G. Krishnan & Amanjit Singh Kainth

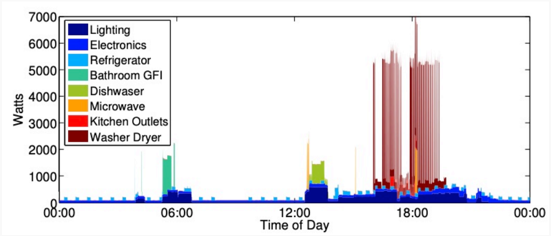
University of Toronto, Fall 2024

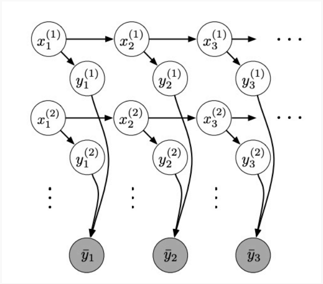
Today

- So far in this course: supervised learning
- Today we start unsupervised learning
    - No labels, so the purpose is to find patterns in data
    - Need to specify what kind of patterns to look for
- **This week:** dimensionality reduction
    - Linear dimensionality reduction (Principal Component Analysis)
        - Matrix completion (needed for the project) is closely related to PCA.
    - Nonlinear dimensionality reduction (autoencoders)
- **Last Week :** clustering

Motivating Examples

Energy disaggregation

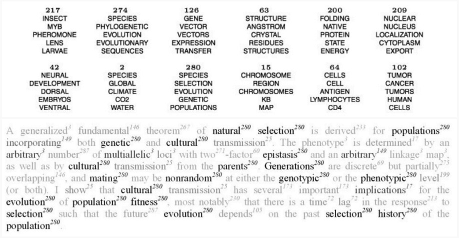




Kolter and Johnson, “REDD: A public data set for energy disaggregation research”

Motivating Examples

Modeling the change in scientific topics over time



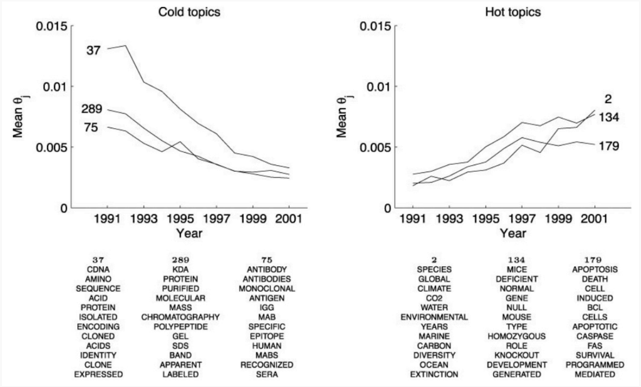
217	274	126	63	200	209
INSECT	SPECIES	GENE	STRUCTURE	FOLDING	NUCLEAR
MYB	PHYLOGENETIC	VECTOR	ANGSTROM	NATIVE	NUCLEUS
PHEROMONE	EVOLUTION	VECTORS	CRYSTAL	PROTEIN	LOCALIZATION
LENS	EVOLUTIONARY	EXPRESSION	RESIDUES	STATE	CYTOPLASM
LARVAE	SEQUENCES	TRANSFER	STRUCTURES	ENERGY	EXPORT
42	2	280	15	64	102
NEURAL	SPECIES	SPECIES	CHROMOSOME	CELLS	TUMOR
DEVELOPMENT	GLOBAL	SELECTION	REGION	CELL	CANCER
DORSAL	CLIMATE	EVOLUTION	CHROMOSOMES	ANTIGEN	TUMORS
EMBRYOS	CO2	GENETIC	KB	LYMPHOCYTES	HUMAN
VENTRAL	WATER	POPULATIONS	MAP	CD4	CELLS

A generalized^{3} fundamental^{146} theorem^{267} of **natural**^{280} **selection**^{280} is derived^{233} for **populations**^{280} incorporating^{149} both **genetic**^{280} and **cultural**^{280} transmission^{25}. The phenotype^{3} is determined^{17} by an arbitrary^{3} number^{287} of multiallelic^{3} loci^{3} with two^{271}-factor^{60} **epistasis**^{280} and an arbitrary^{149} linkage^{3} map^{3}, as well as by **cultural**^{280} transmission^{25} from the **parents**^{280}. **Generations**^{280} are discrete^{69} but partially^{275} overlapping^{146}, and **mating**^{280} may be **nonrandom**^{280} at either the **genotypic**^{280} or the **phenotypic**^{280} level^{199} (or both). I show^{25} that **cultural**^{280} transmission^{25} has several^{173} important^{173} implications^{17} for the **evolution**^{280} of **population**^{280} **fitness**^{280}, most notably^{230} that there is a time^{72} lag^{72} in the response^{213} to **selection**^{280} such that the future^{287} **evolution**^{280} depends^{105} on the past **selection**^{280} **history**^{280} of the **population**^{280}.

Griffiths and Steyvers, “Finding scientific topics”

Motivating Examples

Modeling the change in scientific topics over time



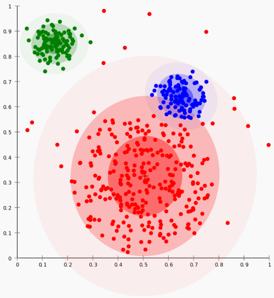
37	289	75	2	134	179
CDNA	KDA	ANTIBODY	SPECIES	MICE	APOPTOSIS
AMINO	PROTEIN	ANTIBODIES	GLOBAL	DEFICIENT	DEATH
SEQUENCE	PURIFIED	MONOCLONAL	CLIMATE	NORMAL	CELL
ACID	MOLECULAR	ANTIGEN	CO2	GENE	INDUCED
PROTEIN	MASS	IGG	WATER	NULL	BCL
ISOLATED	CHROMATOGRAPHY	MAB	ENVIRONMENTAL	MOUSE	CELLS
ENCODING	POLYPEPTIDE	SPECIFIC	YEARS	TYPE	APOPTOTIC
CLONED	GEL	EPITOPE	MARINE	HOMOZYGOUS	CASPASE
ACIDS	SDS	HUMAN	CARBON	ROLE	FAS
IDENTITY	BAND	MABS	DIVERSITY	KNOCKOUT	SURVIVAL
CLONE	APPARENT	RECOGNIZED	OCEAN	DEVELOPMENT	PROGRAMMED
EXPRESSED	LABELED	SERA	EXTINCTION	GENERATED	MEDIATED

Griffiths and Steyvers, “Finding scientific topics”

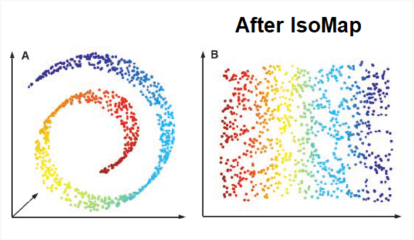
Motivating Examples

The models for those tasks are fairly complicated. In this course, we'll focus on two simpler instances of unsupervised learning:

Clustering



Dimensionality Reduction



Linear Dimensionality Reduction

- We'll start with a simpler form of dimensionality reduction: linear dimensionality reduction $X: N \times D \quad Z: N \times K \quad K << D$
- **Example:** suppose you're a psychologist interested in modeling the variation in human personality
  - You've asked lots of participants to take a survey with lots of personality questions.
  - By figuring out which questions are highly correlated with each other, you can uncover the main factors describing human personality.
- A linear dimensionality reduction model called factor analysis found five key personality traits called the Big Five:
  - extraversion, agreeableness, openness to experience, conscientiousness, neuroticism
- In this lecture, we'll consider a different but closely related model called Principal Component Analysis (PCA).

PCA: Overview

- Principal Component Analysis (PCA) is our first unsupervised learning algorithm, and an example of linear dimensionality reduction.
- **Dimensionality reduction:** map data to a lower dimensional space
  - Save computation/memory
  - Reduce overfitting, achieve better generalization
  - Visualize in 2 dimensions
- Since PCA is a linear model, this mapping will be a **projection**.

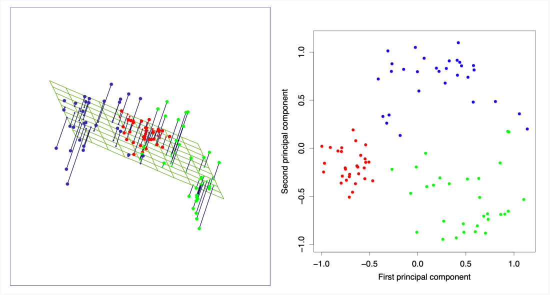


Image credit: Elements of Statistical Learning

Recall

- Every symmetric matrix admits a Spectral Decomposition $A = Q^T \Lambda Q$ where the columns of Q are eigenvectors & the diagonal elements of Λ are eigenvalues.

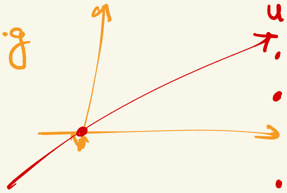
$Av = \lambda v$

- Recall a symmetric matrix is an operator $\Rightarrow$ it transforms points from $\mathbb{R}^D \to \mathbb{R}^D$. The range of the points in the space
  - The eigenvectors form a basis for $\mathbb{R}^D$
  - The eigenvalues describe how much points are stretched in the direction of the eigenvalue.

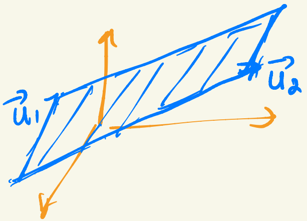
Lets start by thinking of how can project data onto a subspace.

Let us think about subspaces S : a linear region that lies in $\mathbb{R}^D$.

e.g



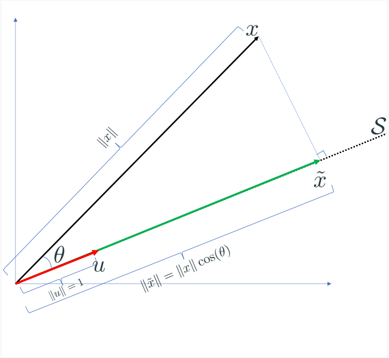
- a line is a subspace of 2D
- we can describe any point on the line as $z \cdot \vec{u}$
- $\vec{u}$ is a basis vector for this subspace.



- a plane is a subspace of 3D
- we can describe any point on the plane as $z_1\vec{u}_1 + z_2\vec{u}_2$
- $\vec{u}_1, \vec{u}_2$ are basis vectors for this subspace.

Euclidean projection

Projection onto a 1-D subspace



- Subspace $\mathcal{S}$ is the line along the unit vector $\mathbf{u}$
  - $\{\mathbf{u}\}$ is a basis for $\mathcal{S}$: any point in $\mathcal{S}$ can be written as $z\mathbf{u}$ for some z.

- Projection of $\mathbf{x}$ on $\mathcal{S}$ is denoted by $\text{Proj}_{\mathcal{S}}(\mathbf{x})$
- Recall: $\mathbf{x}^\top \mathbf{u} = \|\mathbf{x}\| \|\mathbf{u}\| \cos(\theta) = \|\mathbf{x}\| \cos(\theta)$
- $\text{Proj}_{\mathcal{S}}(\mathbf{x}) = \underbrace{\mathbf{x}^\top \mathbf{u}}_{\text{length of proj}} \cdot \underbrace{\mathbf{u}}_{\text{direction of proj}} = \|\tilde{\mathbf{x}}\| \mathbf{u}$

General subspaces

- How to project onto a K-dimensional subspace?
  - **Idea:** choose an orthonormal basis $\{\underline{\mathbf{u}_1}, \underline{\mathbf{u}_2}, \cdots, \underline{\mathbf{u}_K}\}$ for $\mathcal{S}$ (i.e. all unit vectors and orthogonal to each other)
  - Project onto each unit vector individually (as in previous slide), and sum together the projections.
- Mathematically, the projection is given as:

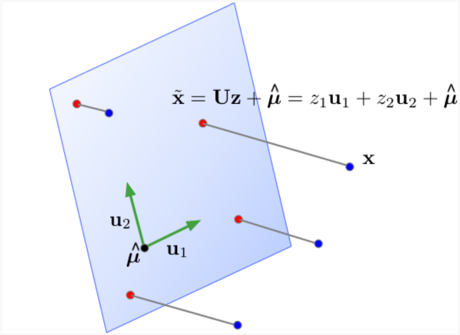
$$\text{Proj}_{\mathcal{S}}(\mathbf{x}) = \sum_{i=1}^{K} z_i \mathbf{u}_i \quad \text{where} \quad z_i = \mathbf{x}^\top \mathbf{u}_i.$$

- In vector form:

$$\text{Proj}_{\mathcal{S}}(\mathbf{x}) = \mathbf{U}\mathbf{z} \quad \text{where} \quad \mathbf{z} = \mathbf{U}^\top \mathbf{x}$$

Projection onto a Subspace

- So far, we assumed the subspace passes through $\mathbf{0}$.
- In mathematical terminology, the "subspaces" we want to project onto are really affine spaces, and can have an arbitrary origin $\hat{\boldsymbol{\mu}}$.

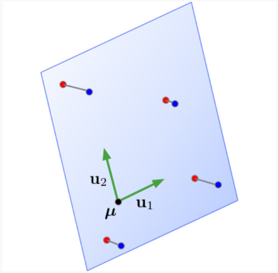

$\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} + \hat{\boldsymbol{\mu}} = z_1\mathbf{u}_1 + z_2\mathbf{u}_2 + \hat{\boldsymbol{\mu}}$
$\mathbf{x}$
$\mathbf{u}_2$
$\mathbf{u}_1$
$\hat{\boldsymbol{\mu}}$

$$\mathbf{z} = \mathbf{U}^\top(\mathbf{x} - \hat{\boldsymbol{\mu}})$$

- In machine learning, $\tilde{\mathbf{x}}$ is also called the reconstruction of $\mathbf{x}$.
- $\mathbf{z}$ is its representation, or code.

Projection onto a Subspace

- If we have a K-dimensional subspace in a D-dimensional input space, then $\mathbf{x} \in \mathbb{R}^D$ and $\mathbf{z} \in \mathbb{R}^K$.
- If the data points $\mathbf{x}$ all lie close to their reconstructions, then we can approximate distances, etc. in terms of these same operations on the code vectors $\mathbf{z}$.
- If $K \ll D$, then it's much cheaper to work with $\mathbf{z}$ than $\mathbf{x}$.
- A mapping to a space that's easier to manipulate or visualize is called a representation, and learning such a mapping is representation learning.
- Mapping data to a low-dimensional space is called dimensionality reduction.



Learning a Subspace

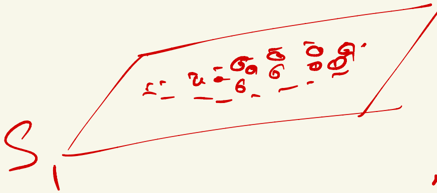
- How to choose a good subspace $\mathcal{S}$?
  - Origin $\hat{\boldsymbol{\mu}}$ is the empirical mean of the data
  - Need to choose a $D \times K$ matrix $\mathbf{U}$ with orthonormal columns.
- Two criteria:
  - Minimize the reconstruction error:

$$\min_{\mathbf{U}} \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2$$

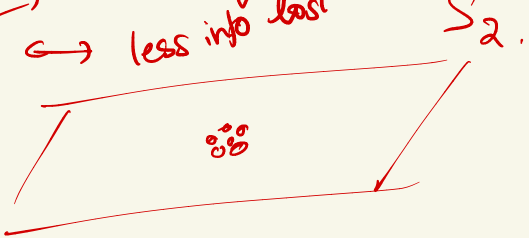
  - Maximize the variance of reconstructions: Find a subspace where data has the most variability.

$$\max_{\mathbf{U}} \frac{1}{N} \sum_{i} \|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2$$

    - Note: The data and its reconstruction have the same means (exercise)!



- higher var $\updownarrow$ = better classification of unique information.
- preserve more covariance $\longleftrightarrow$ less info lost



Learning a Subspace

- These two criteria are equivalent! I.e., we'll show

$$\frac{1}{N}\sum_{i=1}^{N}||\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}||^2 = \text{const} - \frac{1}{N}\sum_{i}||\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}||^2$$

- Recall $\tilde{\mathbf{x}}^{(i)} = \hat{\boldsymbol{\mu}} + \mathbf{U}\mathbf{z}^{(i)}$ and $\mathbf{z}^{(i)} = \mathbf{U}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})$.

- **Observation 1**: Because the columns of $\mathbf{U}$ are orthogonal, $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$, so

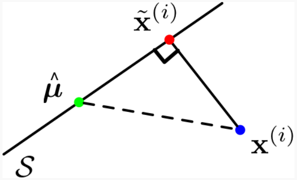
$$||\tilde{\mathbf{x}} - \hat{\boldsymbol{\mu}}||^2 = ||\mathbf{U}\mathbf{z}||^2 = \mathbf{z}^\top\mathbf{U}^\top\mathbf{U}\mathbf{z} = \mathbf{z}^\top\mathbf{z} = ||\mathbf{z}||^2.$$

  $\implies$ norm of centered reconstruction is equal to norm of representation.
  (If you draw it, this is obvious).

Pythagorean Theorem

- Observation 1: $\|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2 = \|\mathbf{z}^{(i)}\|^2$
  - Variance of reconstructions is equal to variance of code vectors: $\frac{1}{N}\sum_i \|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2 = \frac{1}{N}\sum_i \|\mathbf{z}^{(i)}\|^2$ (exercise $\frac{1}{N}\sum_i \mathbf{z}^{(i)} = 0$)

- **Observation 2**: orthogonality of $\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}$ and $\tilde{\mathbf{x}}^{(i)} - \mathbf{x}^{(i)}$
  (Two vectors $\mathbf{a}, \mathbf{b}$ are orthogonal $\iff \mathbf{a}^\top \mathbf{b} = 0$)
- Recall $\tilde{\mathbf{x}}^{(i)} = \hat{\boldsymbol{\mu}} + \mathbf{U}\mathbf{U}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})$.



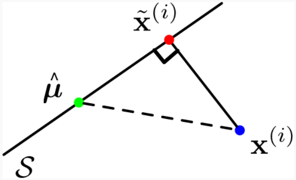
$$
\begin{aligned}
&(\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}})^\top(\tilde{\mathbf{x}}^{(i)} - \mathbf{x}^{(i)}) \\
&= (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top \mathbf{U}\mathbf{U}^\top(\hat{\boldsymbol{\mu}} - \mathbf{x}^{(i)} + \mathbf{U}\mathbf{U}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})) \\
&= (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top \mathbf{U}\mathbf{U}^\top(\hat{\boldsymbol{\mu}} - \mathbf{x}^{(i)}) + (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top \mathbf{U}\mathbf{U}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}) \\
&= 0
\end{aligned}
$$

Pythagorean Theorem

The Pythagorean Theorem tells us:

$$\|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2 + \|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2 = \|\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}\|^2 \qquad \text{for each } i$$

By averaging over data and from observation 2, we obtain



$$\underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2}_{\text{projected variance}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}^{(i)} - \tilde{\mathbf{x}}^{(i)}\|^2}_{\text{reconstruction error}}$$

$$= \underbrace{\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}\|^2}_{\text{constant}}$$

Therefore,

projected variance = constant − reconstruction error

Maximizing the variance is equivalent to minimizing the reconstruction error!

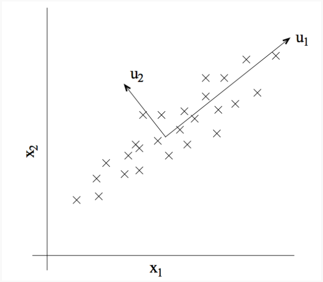
Principal Component Analysis

Choosing a subspace to maximize the projected variance, or minimize the reconstruction error, is called principal component analysis (PCA).

- Consider the empirical covariance matrix:

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top$$

- Recall: $\hat{\boldsymbol{\Sigma}}$ is symmetric and positive semidefinite.
- The optimal PCA subspace is spanned by the top K eigenvectors of $\hat{\boldsymbol{\Sigma}}$.
    - More precisely, choose the first K of any orthonormal eigenbasis for $\hat{\boldsymbol{\Sigma}}$.
    - The general case is tricky, but we'll show this for $K = 1$.
- These eigenvectors are called principal components, analogous to the principal axes of an ellipse.



Deriving PCA

- For $K = 1$, we are fitting a unit vector $\mathbf{u}$, and the code is a scalar $z^{(i)} = \mathbf{u}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})$. Let's maximize the projected variance. From observation 1, we have

$$
\begin{aligned}
\frac{1}{N}\sum_i \|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2 &= \frac{1}{N}\sum_i [z^{(i)}]^2 = \frac{1}{N}\sum_i (\mathbf{u}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}}))^2 \\
&= \frac{1}{N}\sum_{i=1}^N \mathbf{u}^\top(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top\mathbf{u} && (\mathbf{a}^\top\mathbf{b})^2 = \mathbf{a}^\top\mathbf{b}\mathbf{b}^\top\mathbf{a} \\
&= \mathbf{u}^\top\left[\frac{1}{N}\sum_{i=1}^N (\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})(\mathbf{x}^{(i)} - \hat{\boldsymbol{\mu}})^\top\right]\mathbf{u} \\
&= \mathbf{u}^\top\hat{\boldsymbol{\Sigma}}\mathbf{u} \\
&= \mathbf{u}^\top\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top\mathbf{u} && \text{Spectral Decomposition } \hat{\boldsymbol{\Sigma}} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\top \\
&= \mathbf{a}^\top\boldsymbol{\Lambda}\mathbf{a} && \text{for } \mathbf{a} = \mathbf{Q}^\top\mathbf{u} \\
&= \sum_{j=1}^D \lambda_j a_j^2
\end{aligned}
$$

① $\underbrace{\frac{1}{N}\sum_{i}\|\tilde{\mathbf{x}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2}_{LHS}$ $= \mathbf{a}^\top \boldsymbol{\Lambda}\mathbf{a}$ RHS

$= \sum_{j=1}^{D} \lambda_j a_j^2$

$\hat{\Sigma} = \underset{D\times D}{Q^T}\ \underset{\substack{D\times D\\ \text{diagonal}}}{\Lambda}\ \underset{D\times D}{Q}$ ②

$\underset{D\times 1}{a} = \underset{D\times D}{Q^T}\ \underset{D\times 1}{u} \longrightarrow$ what we need to find. u to minimize HS.

e.g $D=3$

$$\underbrace{\boxed{a_1\ a_2\ a_3}}_{a^T}\ \underbrace{\underset{\Lambda}{\begin{bmatrix} \lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & \lambda_3\end{bmatrix}}\ \underset{a}{\begin{bmatrix} a_1\\ a_2\\ a_3\end{bmatrix}}}$$

$$\underset{a^T}{\boxed{a_1 a_2 a_3}}\ \underset{\Lambda a}{\begin{bmatrix}\lambda_1 a_1\\ \lambda_2 a_2\\ \lambda_3 a_3\end{bmatrix}} = \sum_{j=1}^{D} \lambda_j a_j^2$$

• we know

a is unitary

$a^T a = u^T \underline{Q Q^T} u$
$\quad\quad\quad\quad I$

$= u^T u = 1$

$\Rightarrow \sum a_i^2 = 1$

• So we need to design vector a which is unitary to maximize LHS

• Strategy, set $\vec{a}$ as a one-hot vector.

BUT WHERE TO PUT THE ONE?

We know $\lambda_1 \geq \lambda_2 \geq \lambda_3 \dots \lambda_D$

so to maximize

$\sum_{i=1}^{D} \lambda_i^2 a_i^2$ we need to set $a_1 = 1$ & $a_{j \neq 1} = 0$

$\therefore \quad u = Qa = Q \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ = picks the first row of Q

= first eigenvalues eigenvector $= q_1$

Deriving PCA

- Maximize $\mathbf{a}^\top \mathbf{\Lambda} \mathbf{a} = \sum_{j=1}^{D} \lambda_j a_j^2$ for $\mathbf{a} = \mathbf{Q}^\top \mathbf{u}$.
    - This is a change-of-basis to the eigenbasis of $\mathbf{\Sigma}$.
- Assume the λ_i are in sorted order, $\lambda_1 \geq \lambda_2, \geq \ldots$
- Observation: since $\mathbf{u}$ is a unit vector, then by unitarity, $\mathbf{a}$ is also a unit vector: $\mathbf{a}^\top \mathbf{a} = \mathbf{u}^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{u} = \mathbf{u}^\top \mathbf{u}$, i.e., $\sum_j a_j^2 = 1$.
- By inspection, set $a_1 = \pm 1$ and $a_j = 0$ for $j \neq 1$.
- Hence, $\mathbf{u} = \mathbf{Q}\mathbf{a} = \mathbf{q}_1$ (the top eigenvector).

- A similar argument shows that the kth principal component is the kth eigenvector of $\mathbf{\Sigma}$.

Recap

Recap:

- Dimensionality reduction aims to find a low-dimensional representation of the data.
- PCA projects the data onto a subspace which maximizes the projected variance, or equivalently, minimizes the reconstruction error.
- The optimal subspace is given by the top eigenvectors of the empirical covariance matrix.
- PCA gives a set of decorrelated features.

- PCA has decorrelated features $\Rightarrow$ $z^{(i)}$'s $\forall i \in \{1 \dots n\}$ have features that are all independent of one another.

To show this try writing out what the covariance matrix looks like!

$$\mathbb{E}[zz^T] = \mathbb{E}[U^T(x-\mu)(x-\mu)^T U]$$

Expectation is over x due to change of variables $\therefore$ indep. of U, move inside.

$$= U^T \underbrace{\mathbb{E}[(x-\mu)(x-\mu)^T]}_{\text{covariance matrix } \hat{\Sigma}} U$$

$$= U^T \hat{\Sigma} U$$

simplify via spectral decomposition & analyze the result!

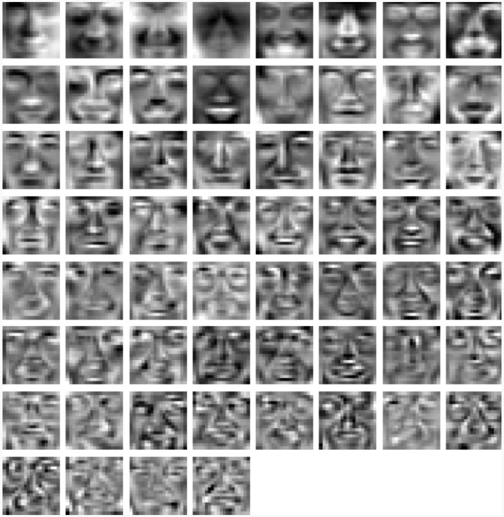
Applying PCA to faces

- Consider running PCA on 2429 19x19 grayscale images (CBCL data)
- Can get good reconstructions with only 3 components

- PCA for pre-processing: can apply classifier to latent representation
  - Original data is 361 dimensional
  - For face recognition PCA with 3 components obtains 79% accuracy on face/non-face discrimination on test data vs. 76.8% for a Gaussian mixture model (GMM) with 84 states. (We'll cover GMMs later in the course.)
- Can also be good for visualization

Applying PCA to faces: Learned basis

Principal components of face images (“eigenfaces”)



Lets apply PCA to the following dataset

$$X = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ -4 & -4 \end{bmatrix}$$

- This is a very simple data in 2D but for which we know points lie on a line.
- So the optimal projection is a 1D PCA plot

$$\hat{\mu} = \begin{bmatrix} -1 & -1 \end{bmatrix}$$

$$\hat{X} = X - \vec{1}\mu = \begin{bmatrix} 0 & 0 \\ -1 & -1 \\ -3 & -3 \end{bmatrix}$$

- center the dataset.

$$\hat{\Sigma} = \frac{1}{N-1}\tilde{X}^T\tilde{X} = \frac{1}{2}\begin{bmatrix} 0 & -1 & -3 \\ 0 & -1 & -3 \end{bmatrix}\begin{bmatrix} 0 & 0 \\ -1 & -1 \\ -3 & -3 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

For a 2x2 PSD matrix, the eigenvalues are

$$A = \begin{bmatrix} a & b \\ b & a \end{bmatrix} \quad \lambda_1 = a+b \quad \lambda_2 = a-b$$

$$\underbrace{\lambda_1 = 4}, \underbrace{\lambda_2 = 0}$$

→ This zero is telling us there is no stretching in one direction !

↓ This (larger) eigenvalue will tell us what the principal component is

For $\lambda_i = 4$ solve for the eigenvector using $(A - \lambda I)v = 0$

$$\Rightarrow \left(\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} - \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} \right) \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} -2 & 2 \\ 2 & -2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{array}{l} -2q_1 + 2q_2 = 0 \\ 2q_1 - 2q_2 = 0 \end{array}$$

$$\Rightarrow q_1 = q_2$$

For $\lambda = 0$

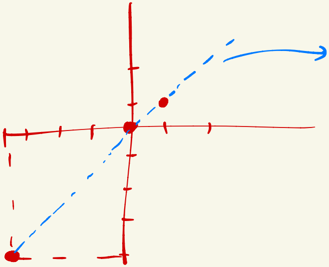
$$\left(\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \right) \begin{bmatrix} \hat{q}_1 \\ \hat{q}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow 2\hat{q}_1 = -2\hat{q}_2 \Rightarrow \hat{q}_1 = -\hat{q}_2$$

Lets pick unit-length eigenvectors!

$$v_1 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} \quad \& \quad v_2 = \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix}$$

$$\lambda_1 = 4 \qquad\qquad \lambda_2 = 0$$

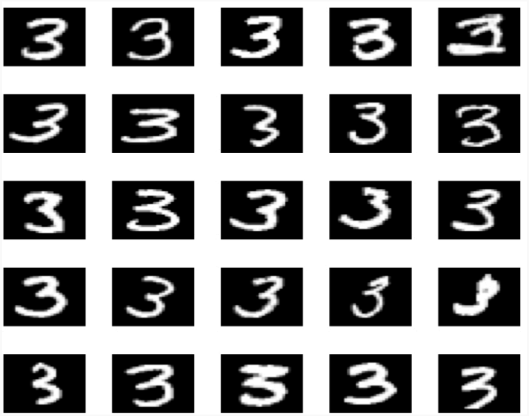


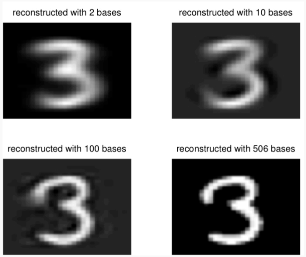
principal component

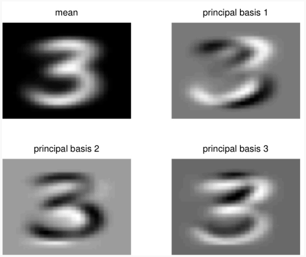
in 1D

with basis $\left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$

Applying PCA to digits




reconstructed with 2 bases
reconstructed with 10 bases
reconstructed with 100 bases
reconstructed with 506 bases

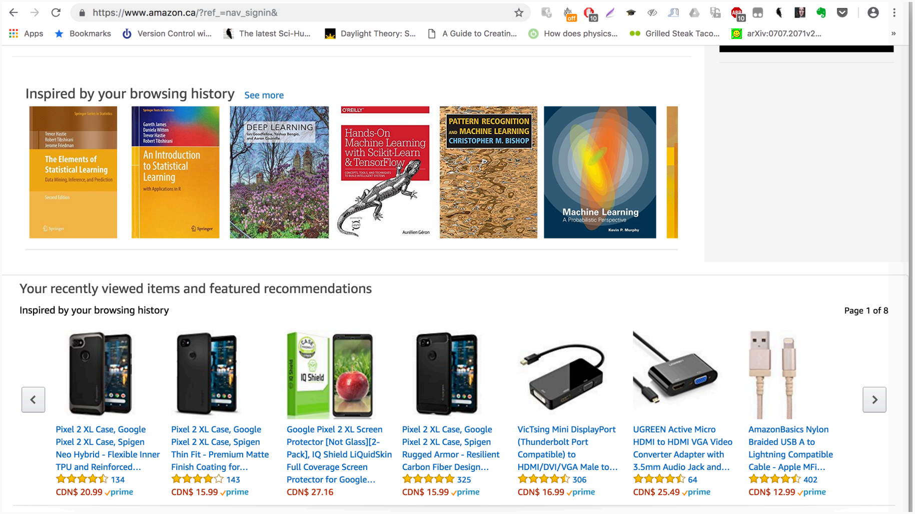


Next

Two more interpretations of PCA, which have interesting generalizations.

1. **Matrix factorization**
2. Autoencoder

Some recommender systems in action



https://www.amazon.ca/?ref_=nav_signin&

Apps | Bookmarks | Version Control wi... | The latest Sci-Hu... | Daylight Theory: S... | A Guide to Creatin... | How does physics... | Grilled Steak Taco... | arXiv:0707.2071v2...

Inspired by your browsing history See more

Your recently viewed items and featured recommendations

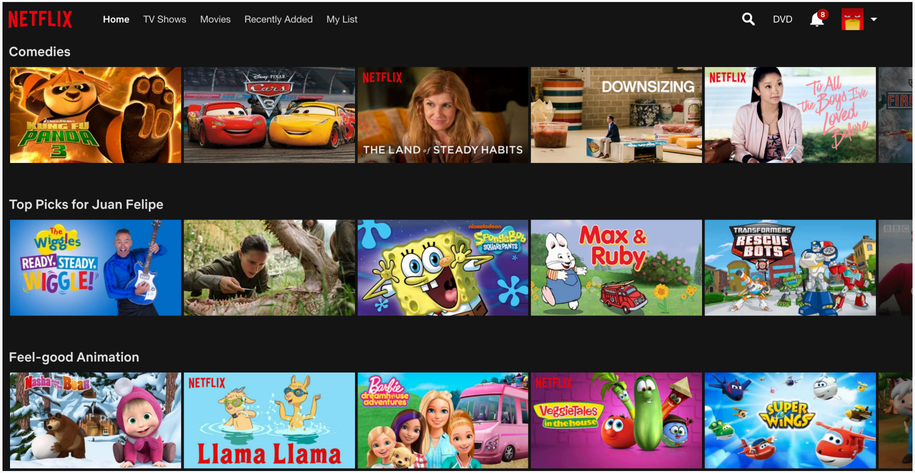
Inspired by your browsing history

Page 1 of 8

Item	Rating count	Price
Pixel 2 XL Case, Google Pixel 2 XL Case, Spigen Neo Hybrid - Flexible Inner TPU and Reinforced...	134	CDN$ 20.99 ✓prime
Pixel 2 XL Case, Google Pixel 2 XL Case, Spigen Thin Fit - Premium Matte Finish Coating for...	143	CDN$ 15.99 ✓prime
Google Pixel 2 XL Screen Protector [Not Glass][2-Pack], IQ Shield LiQuidSkin Full Coverage Screen Protector for Google...		CDN$ 27.16
Pixel 2 XL Case, Google Pixel 2 XL Case, Spigen Rugged Armor - Resilient Carbon Fiber Design...	325	CDN$ 15.99 ✓prime
VicTsing Mini DisplayPort (Thunderbolt Port Compatible) to HDMI/DVI/VGA Male to...	306	CDN$ 16.99 ✓prime
UGREEN Active Micro HDMI to HDMI VGA Video Converter Adapter with 3.5mm Audio Jack and...	64	CDN$ 25.49 ✓prime
AmazonBasics Nylon Braided USB A to Lightning Compatible Cable - Apple MFi...	402	CDN$ 12.99 ✓prime

Ideally recommendations should combine global and seasonal interests, look at your history if available, should adapt with time, be coherent and diverse, etc.

Some recommender systems in action



The Netflix problem

Movie recommendation: Users watch movies and rate them out of 5★.

User	Movie	Rating
🥷	Thor	★ ☆ ☆ ☆ ☆
🥷	Chained	★ ★ ☆ ☆ ☆
🥷	Frozen	★ ★ ★ ☆ ☆
🐱	Chained	★ ★ ★ ★ ☆
🐱	Bambi	★ ★ ★ ★ ★
😇	Titanic	★ ★ ★ ☆ ☆
😇	Goodfellas	★ ★ ★ ★ ★
😇	Dumbo	★ ★ ★ ★ ★
🙂	Twilight	★ ★ ☆ ☆ ☆
😛	Frozen	★ ★ ★ ★ ★
😐	Tangled	★ ☆ ☆ ☆ ☆

Because users only rate a few items, one would like to infer their preference for unrated items

Netflix Prize

PCA as Matrix Factorization

- Recall PCA: each input vector $\mathbf{x}^{(i)} \in \mathbb{R}^D$ is approximated as $\hat{\boldsymbol{\mu}} + \mathbf{U}\mathbf{z}^{(i)}$,

$$\mathbf{x}^{(i)} \approx \tilde{\mathbf{x}}^{(i)} = \hat{\boldsymbol{\mu}} + \mathbf{U}\mathbf{z}^{(i)}$$

  where $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_i \mathbf{x}^{(i)}$ is the data mean, $\mathbf{U} \in \mathbb{R}^{D\times K}$ is the orthogonal basis for the principal subspace, and $\mathbf{z}^{(i)} \in \mathbb{R}^K$ is the code vector, and $\tilde{\mathbf{x}}^{(i)} \in \mathbb{R}^D$ is $\mathbf{x}^{(i)}$'s reconstruction or approximation.
- Assume for simplicity that the data is centered: $\hat{\boldsymbol{\mu}} = 0$. Then, the approximation looks like

$$\mathbf{x}^{(i)} \approx \tilde{\mathbf{x}}^{(i)} = \mathbf{U}\mathbf{z}^{(i)}.$$

PCA as Matrix Factorization

- PCA(on centered data): input vector $\mathbf{x}^{(i)}$ is approximated as $\mathbf{U}\mathbf{z}^{(i)}$

$$\mathbf{x}^{(i)} \approx \mathbf{U}\mathbf{z}^{(i)}$$

- Write this in matrix form, we have $\mathbf{X} \approx \mathbf{Z}\mathbf{U}^\top$ where $\mathbf{X}$ and $\mathbf{Z}$ are matrices with one *row* per data point

$$\mathbf{X} = \begin{bmatrix} [\mathbf{x}^{(1)}]^\top \\ [\mathbf{x}^{(2)}]^\top \\ \vdots \\ [\mathbf{x}^{(N)}]^\top \end{bmatrix} \in \mathbb{R}^{N\times D} \quad \text{and} \quad \mathbf{Z} = \begin{bmatrix} [\mathbf{z}^{(1)}]^\top \\ [\mathbf{z}^{(2)}]^\top \\ \vdots \\ [\mathbf{z}^{(N)}]^\top \end{bmatrix} \in \mathbb{R}^{N\times K}$$

- Can write the squared reconstruction error as

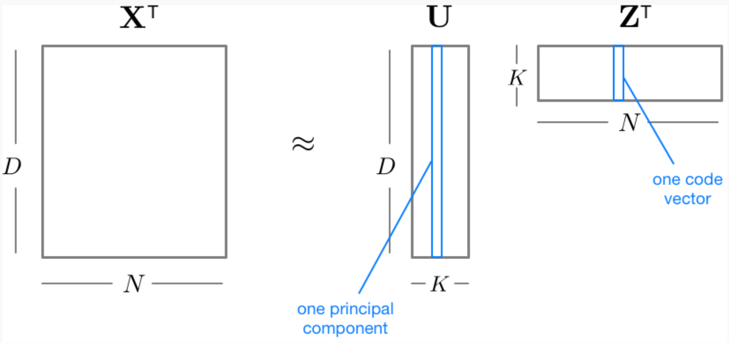
$$\sum_{i=1}^{N} \|\mathbf{x}^{(i)} - \mathbf{U}\mathbf{z}^{(i)}\|^2 = \|\mathbf{X} - \mathbf{Z}\mathbf{U}^\top\|_F^2,$$

- $\|\cdot\|_F$ denotes the Frobenius norm:

$$\|\mathbf{Y}\|_F^2 = \|\mathbf{Y}^\top\|_F^2 = \sum_{i,j} y_{ij}^2 = \sum_i \|\mathbf{y}^{(i)}\|^2.$$

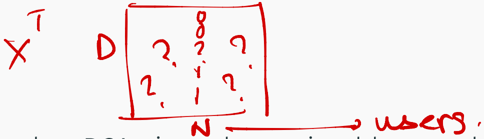
PCA as Matrix Factorization

- So PCA is approximating $\mathbf{X} \approx \mathbf{Z}\mathbf{U}^\top$, or equivalently $\mathbf{X}^\top \approx \mathbf{U}\mathbf{Z}^\top$.



- Based on the sizes of the matrices, this is a rank-K approximation.
- Since $\mathbf{U}$ was chosen to minimize reconstruction error, this is the *optimal* rank-K approximation, in terms of error $||\mathbf{X}^\top - \mathbf{U}\mathbf{Z}^\top||_F^2$.

Matrix Completion



- We just saw that PCA gives the optimal low-rank matrix factorization to a matrix $\mathbf{X}$.
- Can we generalize this to the case where $\mathbf{X}$ is only partially observed?
  - A sparse 1000×1000 matrix with 50,000 observations (only 5% observed).
  - A rank 5 approximation requires only 10,000 parameters, so it's reasonable to fit this.
  - Unfortunately, no closed form solution.

The Netflix problem

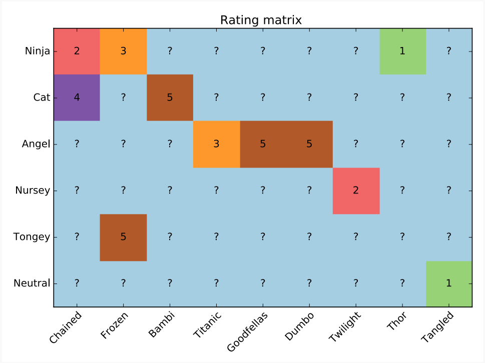
Movie recommendation: Users watch movies and rate them as good or bad.

User	Movie	Rating
🥷	Thor	★ ☆ ☆ ☆ ☆
🥷	Chained	★ ★ ☆ ☆ ☆
🥷	Frozen	★ ★ ★ ☆ ☆
🐱	Chained	★ ★ ★ ★ ☆
🐱	Bambi	★ ★ ★ ★ ★
😇	Titanic	★ ★ ★ ☆ ☆
😇	Goodfellas	★ ★ ★ ★ ★
😇	Dumbo	★ ★ ★ ★ ★
🙂	Twilight	★ ★ ☆ ☆ ☆
😋	Frozen	★ ★ ★ ★ ★
😐	Tangled	★ ☆ ☆ ☆ ☆

Because users only rate a few items, one would like to infer their

Matrix Completion

Matrix completion problem: Transform the table into a N users by M movies matrix $\mathbf{R}$



Rating matrix

	Chained	Frozen	Bambi	Titanic	Goodfellas	Dumbo	Twilight	Thor	Tangled
Ninja	2	3	?	?	?	?	?	1	?
Cat	4	?	5	?	?	?	?	?	?
Angel	?	?	?	3	5	5	?	?	?
Nursey	?	?	?	?	?	?	2	?	?
Tongey	?	5	?	?	?	?	?	?	?
Neutral	?	?	?	?	?	?	?	?	1

- **Data**: Users rate some movies. $\mathbf{R}_{\text{user,movie}}$. Very sparse
- **Task**: Predict missing entries, i.e. how a user would rate a movie they haven't previously rated
- **Evaluation Metric**: Squared error (used by Netflix Competition). Is this a reasonable metric?

Matrix Completion

- In our current setting, latent factor models attempt to explain the ratings by characterizing both movies and users on a number of factors K inferred from the ratings patterns.
- That is, we seek representations for movies and users as vectors in $\mathbb{R}^K$ that can ultimately be translated to ratings.
- For simplicity, we can associate these factors (i.e. the dimensions of the vectors) with idealized concepts like
  - comedy
  - drama
  - action
  - But also uninterpretable dimensions

Can we use the sparse ratings matrix $\mathbf{R}$ to find these latent factors automatically?

Matrix Completion

- Let the representation of user i in the K-dimensional space be $\mathbf{u}_i$ and the representation of movie j be $\mathbf{z}_j$
  - Intuition: maybe the first entry of $\mathbf{u}_i$ says how much the user likes horror films, and the first entry of $\mathbf{z}_j$ says how much movie j is a horror film.
- Assume the rating user i gives to movie j is given by a dot product: $R_{ij} \approx \mathbf{u}_i^\top \mathbf{z}_j$
- In matrix form, if:

$$\mathbf{U} = \begin{bmatrix} - & \mathbf{u}_1^\top & - \\ & \vdots & \\ - & \mathbf{u}_N^\top & - \end{bmatrix} \text{ and } \mathbf{Z}^\top = \begin{bmatrix} | & & | \\ \mathbf{z}_1 & \dots & \mathbf{z}_M \\ | & & | \end{bmatrix}$$

  then: $\mathbf{R} \approx \mathbf{U}\mathbf{Z}^\top$
- This is a matrix factorization problem!

Matrix Completion

- Recall PCA: To enforce $\mathbf{X}^\top \approx \mathbf{U}\mathbf{Z}^\top$, we minimized

$$\min_{\mathbf{U},\mathbf{Z}} \|\mathbf{X}^\top - \mathbf{U}\mathbf{Z}^\top\|_\mathsf{F}^2 = \sum_{i,j} (x_{ji} - \mathbf{u}_i^\top \mathbf{z}_j)^2$$

  where $\mathbf{u}_i$ and $\mathbf{z}_i$ are the i-th rows of matrices $\mathbf{U}$ and $\mathbf{Z}$, respectively.

- What's different about the Netflix problem?
  - Most entries are missing!
  - We only want to count the error for the observed entries.

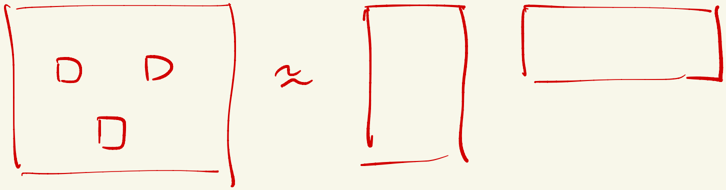
Matrix Completion

- Let $O = \{(n,m) : \text{ entry } (n,m) \text{ of matrix } \mathbf{R} \text{ is observed}\}$
- Using the squared error loss, matrix completion requires solving

$$\min_{\mathbf{U},\mathbf{Z}} \frac{1}{2} \sum_{(i,j)\in O} \left(R_{ij} - \mathbf{u}_i^\top \mathbf{z}_j\right)^2$$

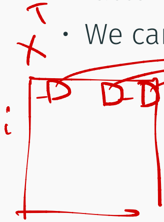
- The objective is non-convex in $\mathbf{U}$ and $\mathbf{Z}$ jointly, and in fact it's generally NP-hard to minimize the above cost function exactly.
- As a function of either $\mathbf{U}$ or $\mathbf{Z}$ individually, the problem is convex and easy to optimize. We can use coordinate descent, just like with K-means and mixture models!

Alternating Least Squares (ALS): fix $\mathbf{Z}$ and optimize $\mathbf{U}$, followed by fix $\mathbf{U}$ and optimize $\mathbf{Z}$, and so on until convergence.



Alternating Least Squares

- Want to minimize the squared error cost with respect to the factor $\mathbf{U}$. (The case of $\mathbf{Z}$ is exactly symmetric.)
- We can decompose the cost into a sum of independent terms:



$$\sum_{(i,j)\in O}\left(R_{ij} - \mathbf{u}_i^\top \mathbf{z}_j\right)^2 = \sum_i \underbrace{\sum_{j:(i,j)\in O}\left(R_{ij} - \mathbf{u}_i^\top \mathbf{z}_j\right)^2}_{\text{only depends on } \mathbf{u}_i}$$

This can be minimized independently for each $\mathbf{u}_i$.
- This is a linear regression problem in disguise. Its optimal solution is:

$$\mathbf{u}_i = \left(\sum_{j:(i,j)\in O} \mathbf{z}_j \mathbf{z}_j^\top\right)^{-1} \sum_{j:(i,j)\in O} R_{ij}\mathbf{z}_j$$

Alternating Least Squares

ALS for Matrix Completion problem

```
1. Initialize $\mathbf{U}$ and $\mathbf{Z}$ randomly
2. repeat until convergence
3.     for $i = 1, .., N$ do
4.         $\mathbf{u}_i = \left(\sum_{j:(i,j)\in O} \mathbf{z}_j \mathbf{z}_j^\top\right)^{-1} \sum_{j:(i,j)\in O} R_{ij}\mathbf{z}_j$
5.     for $j = 1, .., M$ do
6.         $\mathbf{z}_j = \left(\sum_{i:(i,j)\in O} \mathbf{u}_i \mathbf{u}_i^\top\right)^{-1} \sum_{i:(i,j)\in O} R_{ij}\mathbf{u}_i$
```

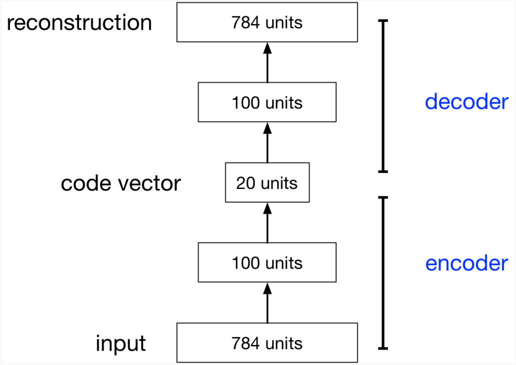
Next

Two more interpretations of PCA, which have interesting generalizations.

1. Matrix factorization
2. **Autoencoder**

Autoencoders

- An **autoencoder** is a feed-forward neural net whose job is to take an input $\mathbf{x}$ and predict $\mathbf{x}$.
- To make this non-trivial, we need to add a **bottleneck layer** whose dimension is much smaller than the input.



Linear Autoencoders

Why autoencoders?

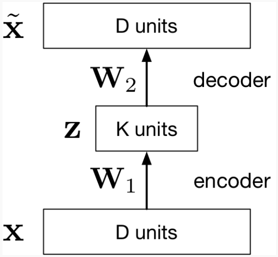
- Map high-dimensional data to two dimensions for visualization
- Learn abstract features in an unsupervised way so you can apply them to a supervised task
    - Unlabled data can be much more plentiful than labeled data

Linear Autoencoders

- The simplest kind of autoencoder has one hidden layer, linear activations, and squared error loss.

$$\mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}}) = ||\mathbf{x} - \tilde{\mathbf{x}}||^2$$

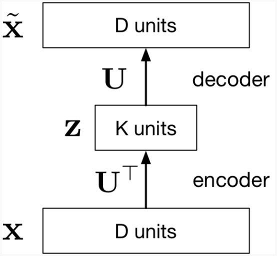
- This network computes $\tilde{\mathbf{x}} = \mathbf{W}_2\mathbf{W}_1\mathbf{x}$, which is a linear function.
- If $K \geq D$, we can choose $\mathbf{W}_2$ and $\mathbf{W}_1$ such that $\mathbf{W}_2\mathbf{W}_1$ is the identity matrix. This isn't very interesting.



- But suppose $K < D$:
  - $\mathbf{W}_1$ maps $\mathbf{x}$ to a K-dimensional space, so it's doing dimensionality reduction.

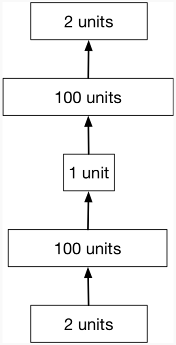
Linear Autoencoders

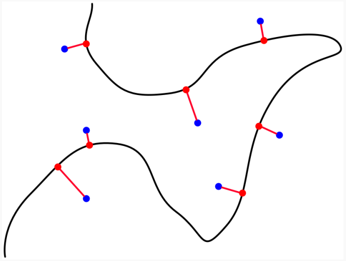
- Observe that the output of the autoencoder must lie in a K-dimensional subspace spanned by the columns of $\mathbf{W}_2$. This is because $\tilde{\mathbf{x}} = \mathbf{W}_2\mathbf{z}$
- We saw that the best possible (min error) K-dimensional linear subspace in terms of reconstruction error is the PCA subspace.
- The autoencoder can achieve this by setting $\mathbf{W}_1 = \mathbf{U}^\top$ and $\mathbf{W}_2 = \mathbf{U}$.
- Therefore, the optimal weights for a linear autoencoder are just the principal components!



Nonlinear Autoencoders

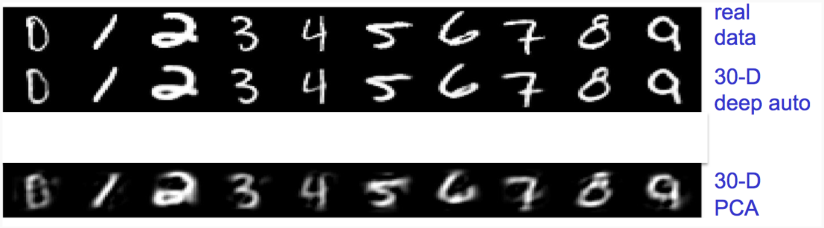
- Deep nonlinear autoencoders learn to project the data, not onto a subspace, but onto a nonlinear manifold
- This manifold is the image of the decoder.
- This is a kind of nonlinear dimensionality reduction.





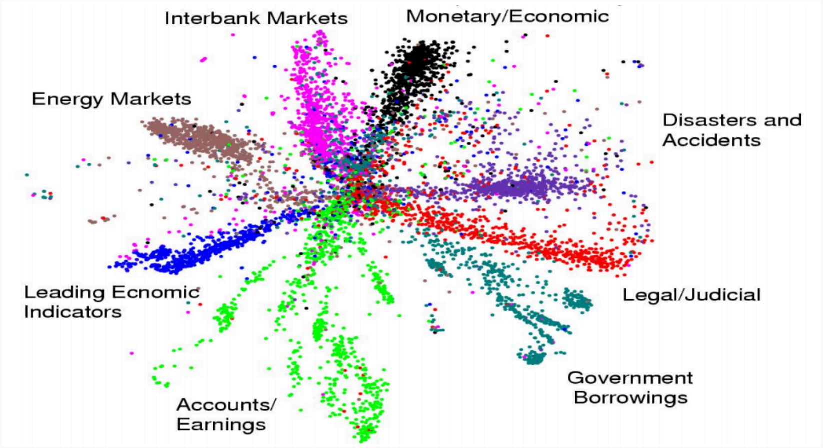
Nonlinear Autoencoders

- Nonlinear autoencoders can learn more powerful codes for a given dimensionality, compared with linear autoencoders (PCA)



Nonlinear Autoencoders

Here’s a 2-dimensional autoencoder representation of newsgroup articles. They’re color-coded by topic, but the algorithm wasn’t given the labels.


Interbank Markets
Monetary/Economic
Energy Markets
Disasters and Accidents
Leading Ecnomic Indicators
Legal/Judicial
Accounts/ Earnings
Government Borrowings