Dimension Reduction using PCA and SVD

Plan of Class

- Starting the machine Learning part of the course.
- Based on Linear Algebra.
- If your linear algebra is rusty, check out the pages on "Resources/Linear Algebra"
- This class will all be theory.
- Next class will be on doing PCA in Spark.
- HW3 will open on friday, be due the following friday.

Dimensionality reduction

Why reduce the number of features in a data set?

1. It reduces storage and computation time.
2. High-dimensional data often has a lot of redundancy.
3. Remove noisy or irrelevant features.

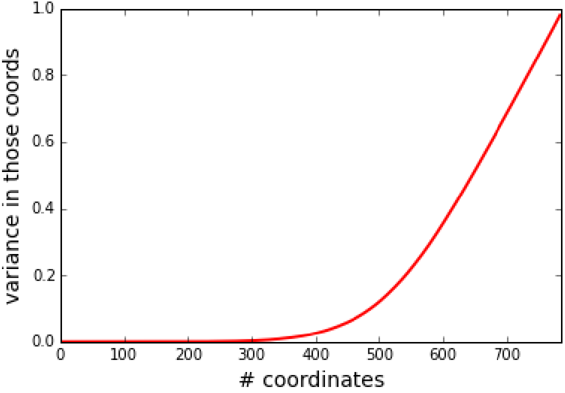
Example: are all the pixels in an image equally informative?
$28 \times 28 = 784$pixels. A vector $\vec{x} \in \mathbb{R}^{784}$

If we were to choose a few pixels to discard, which would be the prime candidates?

Those with lowest variance...

Eliminating low variance coordinates

Example: MNIST. What fraction of the total variance is contained in the 100 (or 200, or 300) coordinates with lowest variance?



We can easily drop 300-400 pixels...

Can we eliminate more?

Yes! By using features that are **combinations** of pixels instead of single pixels.

Covariance (a quick review)

Suppose X has mean μ_X and Y has mean μ_Y.

- Covariance

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

Maximized when $X = Y$, in which case it is $\mathrm{var}(X)$.
In general, it is at most $\mathrm{std}(X)\mathrm{std}(Y)$.

Covariance: example 1

$$\mathrm{cov}(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)] \;=\; \mathbb{E}[XY] - \mu_X\mu_Y$$

x	y	$\Pr(x, y)$
-1	-1	$1/3$
-1	1	$1/6$
1	-1	$1/3$
1	1	$1/6$

$$\begin{aligned}\mu_X &= 0\\ \mu_Y &= \;-1/3\\ \mathrm{var}(X) &= 1\\ \mathrm{var}(Y) &= 8/9\\ \mathrm{cov}(X, Y) &= 0\end{aligned}$$

In this case, X, Y are independent. Independent variables always have zero covariance.

Covariance: example 2

$$\mathrm{cov}(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)] \;=\; \mathbb{E}[XY] - \mu_X \mu_Y$$

x	y	$\Pr(x, y)$
-1	-10	$1/6$
-1	10	$1/3$
1	-10	$1/3$
1	10	$1/6$

$$\begin{aligned} \mu_X &= 0 \\ \mu_Y &= 0 \\ \mathrm{var}(X) &= 1 \\ \mathrm{var}(Y) &= 100 \\ \mathrm{cov}(X, Y) &= -10/3 \end{aligned}$$

In this case, X and Y are negatively correlated.

Example: MNIST

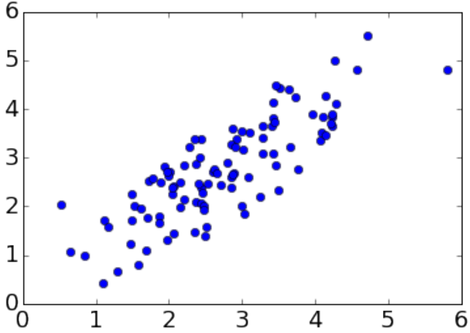
approximate a digit from class j as the class average plus k corrections:

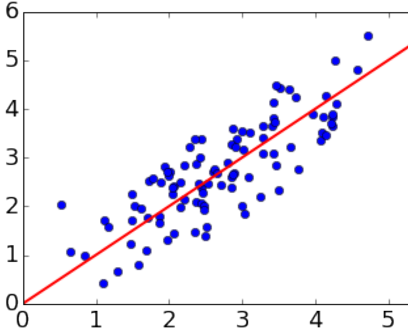
$$\vec{x} \approx \mu_j + \sum_{i=1}^{k} a_i \vec{v}_{j,i}$$

- $\mu_j \in \mathbb{R}^{784}$ class mean vector
- $\vec{v}_{j,1}, \ldots, \vec{v}_{j,k}$ are the **principal directions**.

The effect of correlation

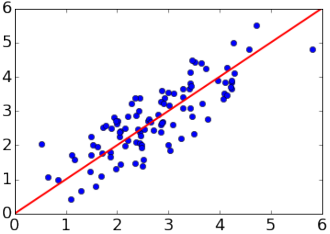
Suppose we wanted just one feature for the following data.



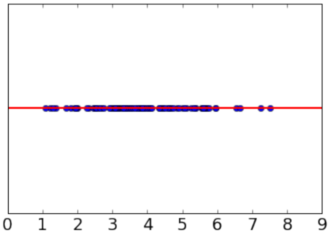


This is the **direction of maximum variance**.

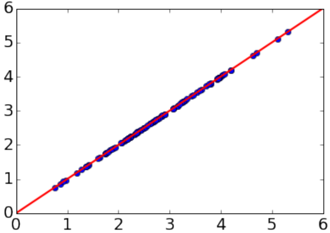
Two types of projection



Projection onto $\mathbb{R}$:

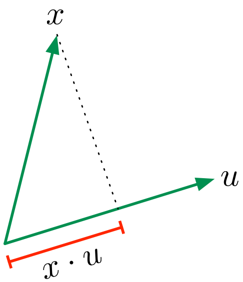


Projection onto a 1-d line in $\mathbb{R}^2$:



Projection: formally

What is the projection of $x \in \mathbb{R}^p$ onto direction $u \in \mathbb{R}^p$ (where $\|u\| = 1$)?



As a one-dimensional value:

$$x \cdot u = u \cdot x = u^T x = \sum_{i=1}^{p} u_i x_i.$$

As a p-dimensional vector:

$$(x \cdot u)u = uu^T x$$

"Move $x \cdot u$ units in direction u"

What is the projection of $x = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$ onto the following directions?

- The coordinate direction e_1? Answer: 2
- The direction $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$? Answer: $-1/\sqrt{2}$

matrix notation I

A notation that allows a simple representation of multiple projections

A vector $\vec{v} \in \mathbb{R}^d$ can be represented, in matrix notation, as

- A column vector:

$$v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_d \end{pmatrix}$$

- A row vector:

$$v^T = \begin{pmatrix} v_1 & v_2 & \cdots & v_d \end{pmatrix}$$

matrix notation II

By convention an inner product is represented by a **row** vector followed by a a **column** vector:

$$\begin{pmatrix} u_1 & u_2 & \cdots & u_d \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_d \end{pmatrix} = \sum_{i=1}^{d} u_i v_i$$

While a **column** vector followd by a **row** vector represents an outer product which is a matrix:

$$\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \begin{pmatrix} u_1 & u_2 & \cdots & u_m \end{pmatrix} = \begin{pmatrix} u_1 v_1 & u_2 v_1 & \cdots & u_m v_1 \\ \vdots & \ddots & \ddots & \vdots \\ u_1 v_n & u_2 v_n & \cdots & u_m v_n \end{pmatrix}$$

Projection onto multiple directions

Want to project $x \in \mathbb{R}^p$ into the k-dimensional subspace defined by vectors $u_1, \ldots, u_k \in \mathbb{R}^p$.

This is easiest when the u_i's are **orthonormal**:

- They each have length one.
- They are at right angles to each other: $u_i \cdot u_j = 0$ whenever $i \neq j$

Then the projection, as a k-dimensional vector, is

$$(x \cdot u_1, x \cdot u_2, \ldots, x \cdot u_k) \;=\; \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_k & \longrightarrow \end{pmatrix}}_{\text{call this } U^T} \begin{pmatrix} \uparrow \\ x \\ \downarrow \end{pmatrix}$$

As a p-dimensional vector, the projection is

$$(x \cdot u_1)u_1 + (x \cdot u_2)u_2 + \cdots + (x \cdot u_k)u_k = UU^T x.$$

Projection onto multiple directions: example

Suppose data are in $\mathbb{R}^4$ and we want to project onto the first two coordinates.

Take vectors $u_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad u_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ (notice: orthonormal)

Then write $U^T = \begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

The projection of $x \in \mathbb{R}^4$, as a 2-d vector, is

$$U^T x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

The projection of x as a 4-d vector is

$$U U^T x = \begin{pmatrix} x_1 \\ x_2 \\ 0 \\ 0 \end{pmatrix}$$

But we'll generally project along non-coordinate directions.

The best single direction

Suppose we need to map our data $x \in \mathbb{R}^p$ into just **one** dimension:

$$x \mapsto u \cdot x \quad \text{for some unit direction } u \in \mathbb{R}^p$$

What is the direction u of maximum variance?

Theorem: Let Σ be the $p \times p$ covariance matrix of X. The variance of X in direction u is given by $u^T \Sigma u$.

- Suppose the mean of X is $\mu \in \mathbb{R}^p$. The projection $u^T X$ has mean

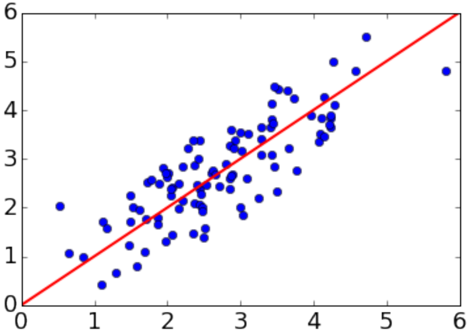
$$\mathbb{E}(u^T X) = u^T \mathbb{E} X = u^T \mu.$$

- The variance of $u^T X$ is

$$\begin{aligned} \text{var}(u^T X) &= \mathbb{E}(u^T X - u^T \mu)^2 = \mathbb{E}(u^T (X - \mu)(X - \mu)^T u) \\ &= u^T \mathbb{E}(X - \mu)(X - \mu)^T u = u^T \Sigma u. \end{aligned}$$

Another theorem: $u^T \Sigma u$ is maximized by setting u to the first **eigenvector** of Σ. The maximum value is the corresponding **eigenvalue**.

Best single direction: example



This direction is the **first eigenvector** of the 2×2 covariance matrix of the data.

The best k-dimensional projection

Let Σ be the $p \times p$ covariance matrix of X. Its **eigendecomposition** can be computed in $O(p^3)$ time and consists of:

- real **eigenvalues** $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$
- corresponding **eigenvectors** $u_1, \ldots, u_p \in \mathbb{R}^p$ that are orthonormal: that is, each u_i has unit length and $u_i \cdot u_j = 0$ whenever $i \neq j$.

Theorem: Suppose we want to map data $X \in \mathbb{R}^p$ to just k dimensions, while capturing as much of the variance of X as possible. The best choice of projection is:

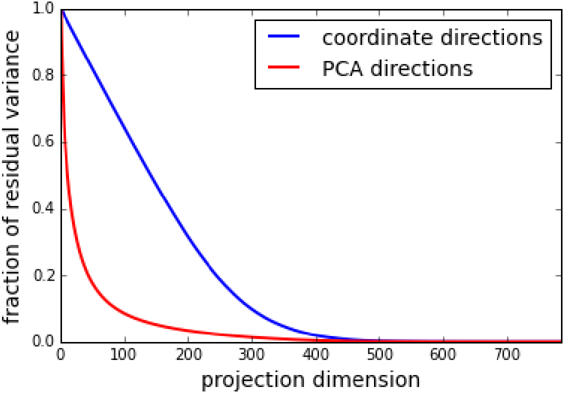
$$x \mapsto (u_1 \cdot x, u_2 \cdot x, \ldots, u_k \cdot x),$$

where u_i are the eigenvectors described above.

Projecting the data in this way is **principal component analysis** (PCA).

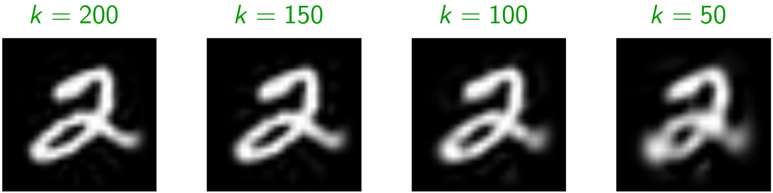
Example: MNIST

Contrast coordinate projections with PCA:



MNIST: image reconstruction

Reconstruct this original image from its PCA projection to k dimensions.



Q: What are these reconstructions exactly?
A: Image x is reconstructed as $UU^T x$, where U is a $p \times k$ matrix whose columns are the top k eigenvectors of Σ.

What are eigenvalues and eigenvectors?

There are several steps to understanding these.

1. Any matrix M defines a function (or **transformation**) $x \mapsto Mx$.
2. If M is a $p \times q$ matrix, then this transformation maps vector $x \in \mathbb{R}^q$ to vector $Mx \in \mathbb{R}^p$.
3. We call it a **linear transformation** because $M(x + x') = Mx + Mx'$.
4. We'd like to understand the nature of these transformations. The easiest case is when M is **diagonal**:

   $$\underbrace{\begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 10 \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} 2x_1 \\ -x_2 \\ 10x_3 \end{pmatrix}}_{Mx}$$

   In this case, M simply scales each coordinate separately.
5. What about more general matrices that are symmetric but not necessarily diagonal? They also just scale coordinates separately, but in a **different coordinate system**.

Eigenvalue and eigenvector: definition

Let M be a $p \times p$ matrix.
We say $u \in \mathbb{R}^p$ is an **eigenvector** if M maps u onto the same direction, that is,

$$Mu = \lambda u$$

for some scaling constant λ. This λ is the **eigenvalue** associated with u.

Question: What are the eigenvectors and eigenvalues of:

$$M = \begin{pmatrix} 2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 10 \end{pmatrix} ?$$

Answer: Eigenvectors $e_1, e_2 . e_3$, with corresponding eigenvalues $2, -1, 10$.

Notice that these eigenvectors form an orthonormal basis.

Eigenvectors of a real symmetric matrix

Theorem. Let M be any real symmetric $p \times p$ matrix. Then M has

- p eigenvalues $\lambda_1, \ldots, \lambda_p$
- corresponding eigenvectors $u_1, \ldots, u_p \in \mathbb{R}^p$ that are orthonormal

We can think of $u_1, \ldots, u_p$ as being the axes of the natural coordinate system for understanding M.

Example: consider the matrix

$$M = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$$

It has eigenvectors

$$u_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad u_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

and corresponding eigenvalues $\lambda_1 = 4$ and $\lambda_2 = 2$. (Check)

Spectral decomposition

Theorem. Let M be any real symmetric $p \times p$ matrix. Then M has

- p eigenvalues $\lambda_1, \ldots, \lambda_p$
- corresponding eigenvectors $u_1, \ldots, u_p \in \mathbb{R}^p$ that are orthonormal

Spectral decomposition: Here is another way to write M:

$$M = \underbrace{\begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ u_1 & u_2 & \cdots & u_p \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}}_{U\text{: columns are eigenvectors}} \underbrace{\begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_p \end{pmatrix}}_{\Lambda\text{: eigenvalues on diagonal}} \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_p & \longrightarrow \end{pmatrix}}_{U^T}$$

Thus $Mx = U\Lambda U^T x$, which can be interpreted as follows:

- U^T rewrites x in the $\{u_i\}$ coordinate system
- Λ is a simple coordinate scaling in that basis
- U then sends the scaled vector back into the usual coordinate basis

Spectral decomposition: example

Apply spectral decomposition to the matrix M we saw earlier:

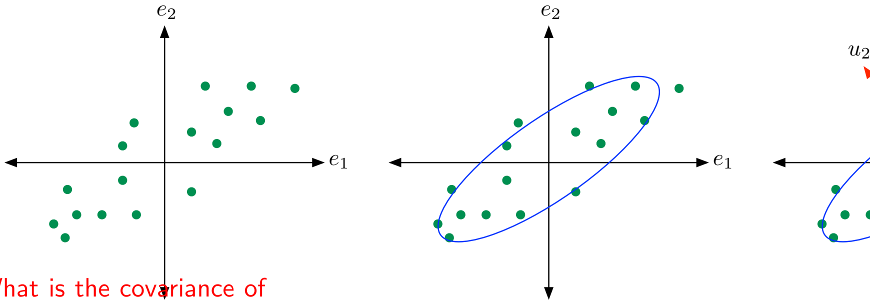
$$M = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix} = \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}}_{U} \underbrace{\begin{pmatrix} 4 & 0 \\ 0 & 2 \end{pmatrix}}_{\Lambda} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}}_{U^T}$$

$$\begin{aligned} M \begin{pmatrix} 1 \\ 2 \end{pmatrix} &= ??? \\ &= U \Lambda U^T \begin{pmatrix} 1 \\ 2 \end{pmatrix} \\ &= U \Lambda \frac{1}{\sqrt{2}} \begin{pmatrix} 3 \\ 1 \end{pmatrix} \\ &= U \frac{1}{\sqrt{2}} \begin{pmatrix} 12 \\ 2 \end{pmatrix} \\ &= \begin{pmatrix} 5 \\ 7 \end{pmatrix} \end{aligned}$$

Principal component analysis: recap

Consider data vectors $X \in \mathbb{R}^p$.

- The covariance matrix Σ is a $p \times p$ symmetric matrix.
- Get eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$, eigenvectors $u_1, \ldots, u_p$.
- $u_1, \ldots, u_p$ is an alternative basis in which to represent the data.
- The variance of X in direction u_i is λ_i.
- To project to k dimensions while losing as little as possible of the overall variance, use $x \mapsto (x \cdot u_1, \ldots, x \cdot u_k)$.



What is the covariance of the projected data?

Example: personality assessment

What are the dimensions along which personalities differ?

- *Lexical hypothesis*: most important personality characteristics have become encoded in natural language.
- Allport and Odbert (1936): sat down with the English dictionary and extracted all terms that could be used to distinguish one person's behavior from another's. Roughly 18000 words, of which 4500 could be described as personality traits.
- Step: group these words into (approximate) synonyms. This is done by manual clustering. e.g. Norman (1967):

Spirit	Jolly, merry, witty, lively, peppy
Talkativeness	Talkative, articulate, verbose, gossipy
Sociability	Companionable, social, outgoing
Spontaneity	Impulsive, carefree, playful, zany
Boisterousness	Mischievous, rowdy, loud, prankish
Adventure	Brave, venturous, fearless, reckless
Energy	Active, assertive, dominant, energetic
Conceit	Boastful, conceited, egotistical
Vanity	Affected, vain, chic, dapper, jaunty
Indiscretion	Nosey, snoopy, indiscreet, meddlesome
Sensuality	Sexy, passionate, sensual, flirtatious

- Data collection: Ask a variety of subjects to what extent each of these words describes them.

Personality assessment: the data

Matrix of data (1 = strongly disagree, 5 = strongly agree)

	shy	merry	tense	boastful	forgiving	quiet
Person 1	4	1	1	2	5	5
Person 2	1	4	4	5	2	1
Person 3	2	4	5	4	2	2
		⋮				

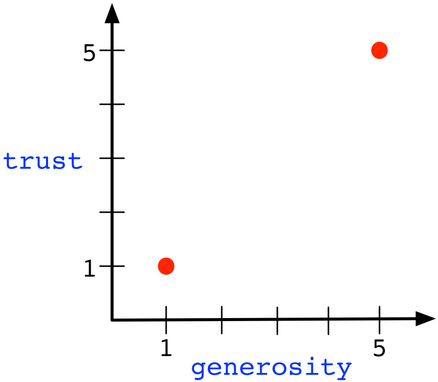
How to extract important directions?

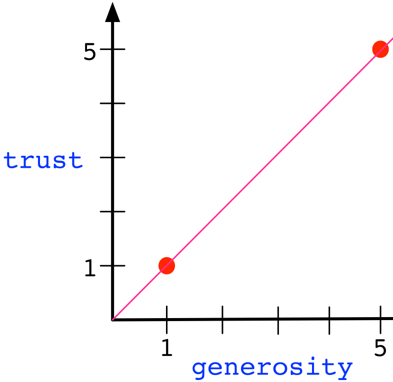
- Treat each column as a data point, find tight clusters
- Treat each row as a data point, apply PCA
- Other ideas: factor analysis, independent component analysis, ...

Many of these yield similar results

What does PCA accomplish?

Example: suppose two traits (generosity, trust) are highly correlated, to the point where each person either answers “1” to both or “5” to both.





This single PCA dimension entirely accounts for the two traits.

The “Big Five” taxonomy

Extraversion		Agreeableness		Conscientiousness		Neuroticism		Oppenness/Intellect	
Low	High	Low	High	Low	High	Low	High	Low	High
-.83 Quiet	.85 Talkative	-.52 Fault-finding	.87 Sympathetic	-.58 Careless	.80 Organized	-.39 Stable*	.73 Tense	-.74 Commonplace	.76 Wide interests
-.80 Reserved	.83 Assertive	-.48 Cold	.85 Kind	-.53 Disorderly	.80 Thorough	-.35 Calm*	.72 Anxious	-.73 Narrow interests	.76 Imaginative
-.75 Shy	.82 Active	-.45 Unfriendly	.85 Appreciative	-.50 Frivolous	.78 Planful	-.21 Contented*	.72 Nervous	-.67 Simple	.72 Intelligent
-.71 Silent	.82 Energetic	-.45 Quarrelsome	.84 Affectionate	-.49 Irresponsible	.78 Efficient	.14 Unemotional*	.71 Moody	-.55 Shallow	.73 Original
-.67 Withdrawn	.82 Outgoing	-.45 Hard-hearted	.84 Soft-hearted	-.40 Slipshot	.73 Responsible		.71 Worrying	-.47 Unintelligent	.68 Insightful
-.66 Retiring	.80 Outspoken	-.38 Unkind	.82 Warm	-.39 Undependable	.72 Reliable		.68 Touchy		.64 Curious
	.79 Dominant	-.33 Cruel	.81 Generous	-.37 Forgetful	.70 Dependable		.64 Fearful		.59 Sophisticated
	.73 Forceful	-.31 Stern*	.78 Trusting		.68 Conscientious		.63 High-strung		.59 Artistic
	.73 Enthusiastic	-.28 Thankless	.77 Helpful		.66 Precise		.63 Self-pitying		.59 Clever
	.68 Show-off	-.24 Stingy*	.77 Forgiving		.66 Practical		.60 Temperamental		.58 Inventive
	.68 Sociable		.74 Pleasant		.65 Deliberate		.59 Unstable		.56 Sharp-witted
	.64 Spunky		.73 Good-natured		.46 Painstaking		.58 Self-punishing		.55 Ingenious
	.64 Adventurous		.73 Friendly		.26 Cautious*		.54 Despondent		.45 Witty*
	.62 Noisy		.72 Cooperative				.51 Emotional		.45 Resourceful*
	.58 Bossy		.67 Gentle						.37 Wise
			.66 Unselfish						.33 Logical*
			.56 Praising						.29 Civilized*
			.51 Sensitive						.22 Foresighted*
									.21 Polished*
									.20 Dignified*

Many applications, such as online match-making.

Singular value decomposition (SVD)

For **symmetric** matrices, such as covariance matrices, we have seen:

- Results about existence of eigenvalues and eigenvectors
- The fact that the eigenvectors form an alternative basis
- The resulting spectral decomposition, which is used in PCA

But what about arbitrary matrices $M \in \mathbb{R}^{p \times q}$?

Any $p \times q$ matrix (say $p \leq q$) has a **singular value decomposition**:

$$M = \underbrace{\begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_p \\ \downarrow & & \downarrow \end{pmatrix}}_{p \times p \text{ matrix } U} \underbrace{\begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_p \end{pmatrix}}_{p \times p \text{ matrix } \Lambda} \underbrace{\begin{pmatrix} \longleftarrow & v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & v_p & \longrightarrow \end{pmatrix}}_{p \times q \text{ matrix } V^T}$$

- $u_1, \dots, u_p$ are orthonormal vectors in $\mathbb{R}^p$
- $v_1, \dots, v_p$ are orthonormal vectors in $\mathbb{R}^q$
- $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p$ are **singular values**

Matrix approximation

We can **factor** any $p \times q$ matrix as $M = UW^T$:

$$M = \begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_p \\ \downarrow & & \downarrow \end{pmatrix} \begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_p \end{pmatrix} \begin{pmatrix} \longleftarrow & v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & v_p & \longrightarrow \end{pmatrix}$$

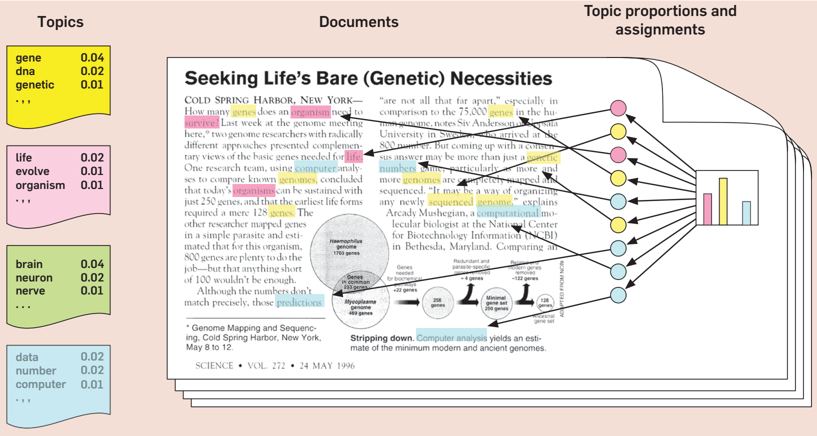
$$= \underbrace{\begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_p \\ \downarrow & & \downarrow \end{pmatrix}}_{p \times p \text{ matrix } U} \underbrace{\begin{pmatrix} \longleftarrow & \sigma_1 v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \sigma_p v_p & \longrightarrow \end{pmatrix}}_{p \times q \text{ matrix } W^T}$$

A concise approximation to M: just take the first k columns of U and the first k rows of W^T, for $k < p$:

$$\widehat{M} = \underbrace{\begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_k \\ \downarrow & & \downarrow \end{pmatrix}}_{p \times k} \underbrace{\begin{pmatrix} \longleftarrow & \sigma_1 v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \sigma_k v_k & \longrightarrow \end{pmatrix}}_{k \times q}$$

Example: topic modeling

Blei (2012):



Latent semantic indexing (LSI)

Given a large corpus of n documents:

- Fix a vocabulary, say of V words.
- Bag-of-words representation for documents: each document becomes a vector of length V, with one coordinate per word.
- The corpus is an $n \times V$ matrix, one row per document.

	cat	dog	house	boat	garden	…
Doc 1	4	1	1	0	2	
Doc 2	0	0	3	1	0	
Doc 3	0	1	3	0	0	
		⋮				

Let's find a concise approximation to this matrix M.

Latent semantic indexing, cont'd

Use SVD to get an approximation to M: for small k,

$$\underbrace{\begin{pmatrix} \longleftarrow & \text{doc } 1 & \longrightarrow \\ \longleftarrow & \text{doc } 2 & \longrightarrow \\ \longleftarrow & \text{doc } 3 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \text{doc } n & \longrightarrow \end{pmatrix}}_{n \times V \text{ matrix } M} \approx \underbrace{\begin{pmatrix} \longleftarrow & \theta_1 & \longrightarrow \\ \longleftarrow & \theta_2 & \longrightarrow \\ \longleftarrow & \theta_3 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \theta_n & \longrightarrow \end{pmatrix}}_{n \times k \text{ matrix } \Theta} \underbrace{\begin{pmatrix} \longleftarrow & \Psi_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \Psi_k & \longrightarrow \end{pmatrix}}_{k \times V \text{ matrix } \Psi}$$

Think of this as a *topic model* with k topics.

- Ψ_j is a vector of length V describing topic j: coefficient Ψ_{jw} is large if word w appears often in that topic.
- Each document is a combination of topics: θ_{ij} is the weight of topic j in document i.

Document i originally represented by ith row of M, a vector in $\mathbb{R}^V$.
Can instead use $\theta_i \in \mathbb{R}^k$, a more concise "semantic" representation.

The rank of a matrix

Suppose we want to approximate a matrix M by a simpler matrix $\widehat{M}$. What is a suitable notion of "simple"?

- Let's say M and $\widehat{M}$ are $p \times q$, where $p \leq q$.
- Treat each row of $\widehat{M}$ as a data point in $\mathbb{R}^q$.
- We can think of the data as "simple" if it actually lies in a low-dimensional subspace.
- If the rows lie in k-dimensional subspace, we say that $\widehat{M}$ has **rank** k.

The **rank** of a matrix is the number of linearly independent rows.

Low-rank approximation: given $M \in \mathbb{R}^{p \times q}$ and an integer k, find the matrix $\widehat{M} \in \mathbb{R}^{p \times q}$ that is the best rank-k approximation to M.

That is, find $\widehat{M}$ so that

- $\widehat{M}$ has rank $\leq k$
- The approximation error $\sum_{i,j}(M_{ij} - \widehat{M}_{ij})^2$ is minimized.

We can get $\widehat{M}$ directly from the singular value decomposition of M.

Low-rank approximation

Recall: Singular value decomposition of $p \times q$ matrix M (with $p \leq q$):

$$M = \begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_p \\ \downarrow & & \downarrow \end{pmatrix} \begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_p \end{pmatrix} \begin{pmatrix} \longleftarrow & v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & v_p & \longrightarrow \end{pmatrix}$$

- $u_1, \ldots, u_p$ is an orthonormal basis of $\mathbb{R}^p$
- $v_1, \ldots, v_q$ is an orthonormal basis of $\mathbb{R}^q$
- $\sigma_1 \geq \cdots \geq \sigma_p$ are **singular values**

The **best rank-k approximation** to M, for any $k \leq p$, is then

$$\widehat{M} = \underbrace{\begin{pmatrix} \uparrow & & \uparrow \\ u_1 & \cdots & u_k \\ \downarrow & & \downarrow \end{pmatrix}}_{p \times k} \underbrace{\begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k \end{pmatrix}}_{k \times k} \underbrace{\begin{pmatrix} \longleftarrow & v_1 & \longrightarrow \\ & \vdots & \\ \longleftarrow & v_k & \longrightarrow \end{pmatrix}}_{k \times q}$$

Example: Collaborative filtering

Details and images from Koren, Bell, Volinksy (2009).

Recommender systems: matching customers with products.

- Given: data on prior purchases/interests of users
- Recommend: further products of interest

Prototypical example: Netflix.

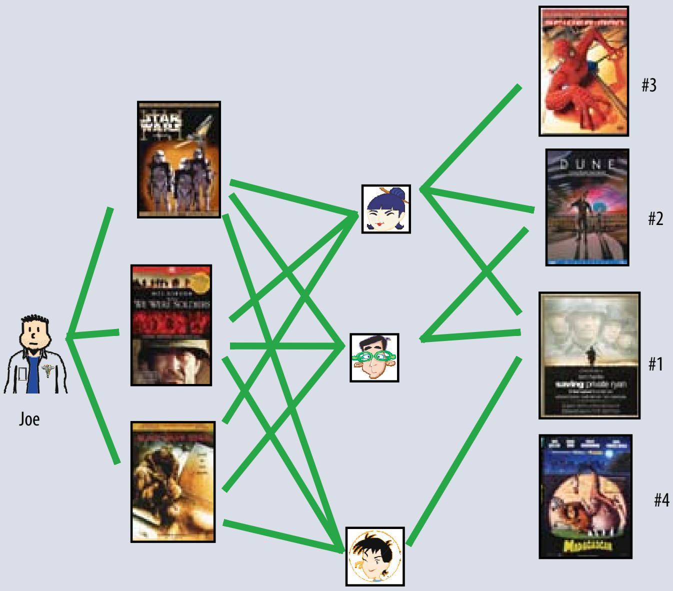
A successful approach: **collaborative filtering**.

- Model dependencies between different products, and between different users.
- Can give reasonable recommendations to a relatively new user.

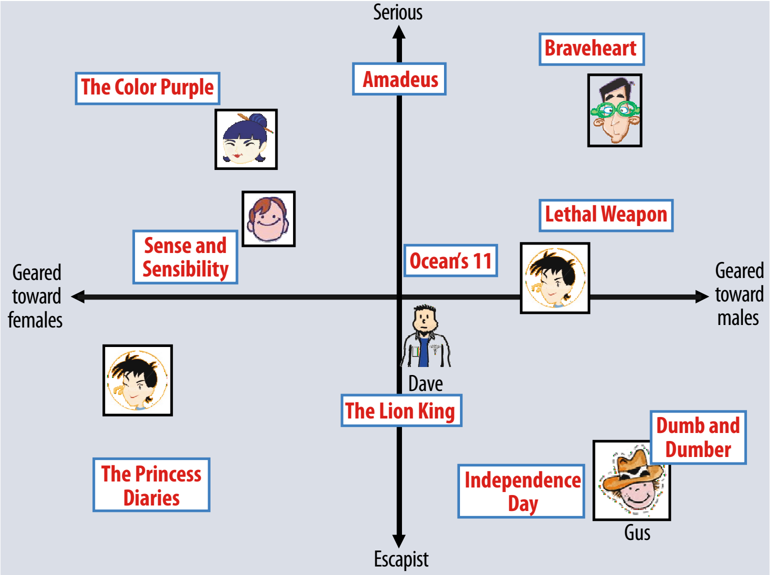
Two strategies for collaborative filtering:

- Neighborhood methods
- Latent factor methods

Neighborhood methods



Latent factor methods



The matrix factorization approach

User ratings are assembled in a large matrix M:

	Star Wars	Matrix	Casablanca	Camelot	Godfather	...
User 1	5	5	2	0	0	
User 2	0	0	3	4	5	
User 3	0	0	5	0	0	
		⋮				

- Not rated = 0, otherwise scores 1-5.
- For n users and p movies, this has size $n \times p$.
- Most of the entries are unavailable, and we'd like to predict these.

Idea: Find the best low-rank approximation of M, and use it to fill in the missing entries.

User and movie factors

Best rank-k approximation is of the form $M \approx UW^T$:

$$\underbrace{\begin{pmatrix} \longleftarrow & \text{user } 1 & \longrightarrow \\ \longleftarrow & \text{user } 2 & \longrightarrow \\ \longleftarrow & \text{user } 3 & \longrightarrow \\ & \vdots & \\ \longleftarrow & \text{user } n & \longrightarrow \end{pmatrix}}_{n \times p \text{ matrix } M} \approx \underbrace{\begin{pmatrix} \longleftarrow & u_1 & \longrightarrow \\ \longleftarrow & u_2 & \longrightarrow \\ \longleftarrow & u_3 & \longrightarrow \\ & \vdots & \\ \longleftarrow & u_n & \longrightarrow \end{pmatrix}}_{n \times k \text{ matrix } U} \underbrace{\begin{pmatrix} \uparrow & \uparrow & & \uparrow \\ w_1 & w_2 & \cdots & w_p \\ \downarrow & \downarrow & & \downarrow \end{pmatrix}}_{k \times p \text{ matrix } W^T}$$

Thus user i's rating of movie j is approximated as

$$M_{ij} \approx u_i \cdot w_j$$

This "latent" representation embeds users and movies within the same k-dimensional space:

- Represent ith user by $u_i \in \mathbb{R}^k$
- Represent jth movie by $w_j \in \mathbb{R}^k$

Top two Netflix factors