

ColumbiaX: Machine Learning

Lecture 17

Prof. John Paisley

Department of Electrical Engineering
& Data Science Institute

Columbia University

COLLABORATIVE FILTERING

OBJECT RECOMMENDATION

Matching consumers to products is an important practical problem.

We can often make these connections using user feedback about subsets of products. To give some prominent examples:

- ▶ Netflix lets users to rate movies
- ▶ Amazon lets users to rate products and write reviews about them
- ▶ Yelp lets users to rate businesses, write reviews, upload pictures
- ▶ YouTube lets users like/dislike a videos and write comments

Recommendation systems use this information to help recommend new things to customers that they may like.

CONTENT FILTERING

One strategy for object recommendation is:

Content filtering: Use known information about the products and users to make recommendations. Create profiles based on

- ▶ Products: movie information, price information, product descriptions
- ▶ Users: demographic information, questionnaire information

Example: A fairly well known example is the online radio Pandora, which uses the “Music Genome Project.”

- ▶ An expert scores a song based on hundreds of characteristics
- ▶ A user also provides information about his/her music preferences
- ▶ Recommendations are made based on pairing these two sources

COLLABORATIVE FILTERING

Content filtering requires a lot of information that can be difficult and expensive to collect. Another strategy for object recommendation is:

Collaborative filtering (CF): Use previous users' input/behavior to make future recommendations. Ignore any *a priori* user or object information.

- ▶ CF uses the ratings of similar users to predict my rating.
- ▶ CF is a domain-free approach. It doesn't need to know what is being rated, just who rated what, and what the rating was.

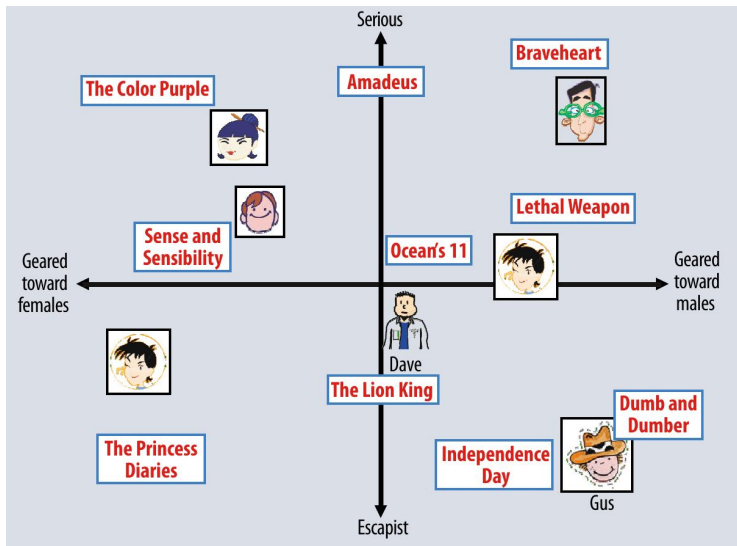
One CF method uses a *neighborhood-based* approach. For example,

1. define a similarity score between me and other users based on how much our overlapping ratings agree, then
2. based on these scores, let others "vote" on what I would like.

These filtering approaches are not mutually exclusive. Content information can be built into a collaborative filtering system to improve performance.

LOCATION-BASED CF METHODS (INTUITION)

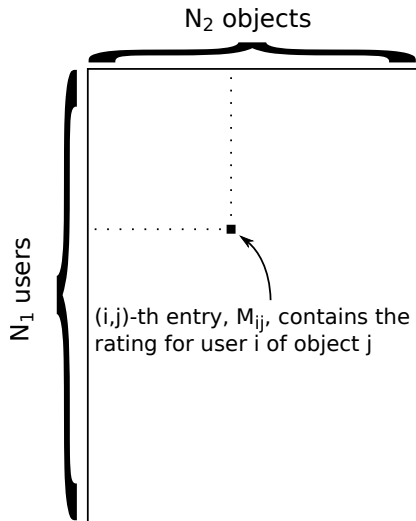
Location-based approaches embed users and objects into points in \mathbb{R}^d .



¹ Koren, Y., Robert B., and Volinsky, C.. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.

MATRIX FACTORIZATION

MATRIX FACTORIZATION



Matrix factorization (MF) gives a way to learn user and object locations.

First, form the rating matrix M :

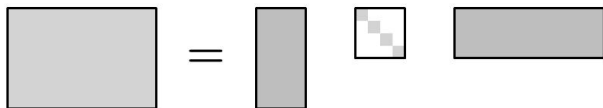
- ▶ Contains every user/object pair.
- ▶ Will have many missing values.
- ▶ The goal is to fill in these missing values.

MF and recommendation systems:

- ▶ We have prediction of every missing rating for user i .
- ▶ Recommend the highly rated objects among the predictions.

SINGULAR VALUE DECOMPOSITION

Our goal is to factorize the matrix M . We've discussed one method already.

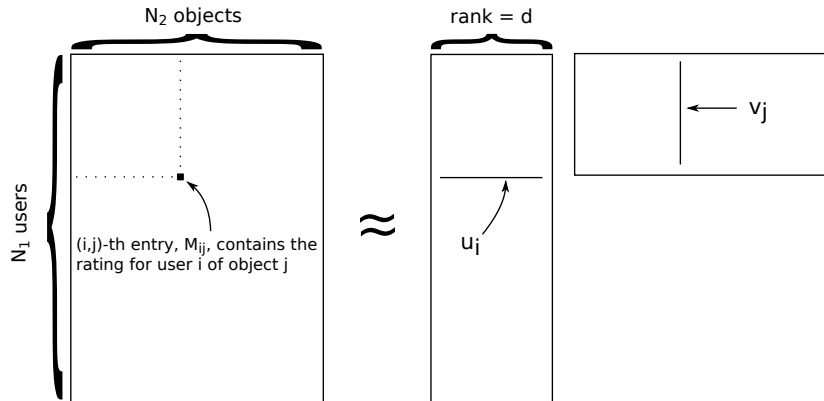

$$\begin{matrix} \mathbf{M} & = & \mathbf{U} & \mathbf{S} & \mathbf{V}^T \\ (n \times d) & & (n \times r) & (r \times r) & (r \times d) \end{matrix}$$

Singular value decomposition: Every matrix M can be written as $M = USV^T$, where $U^T U = I$, $V^T V = I$ and S is diagonal with $S_{ii} \geq 0$.

$r = \text{rank}(M)$. When it's small, M has fewer “degrees of freedom.”

Collaborative filtering with matrix factorization is intuitively similar.

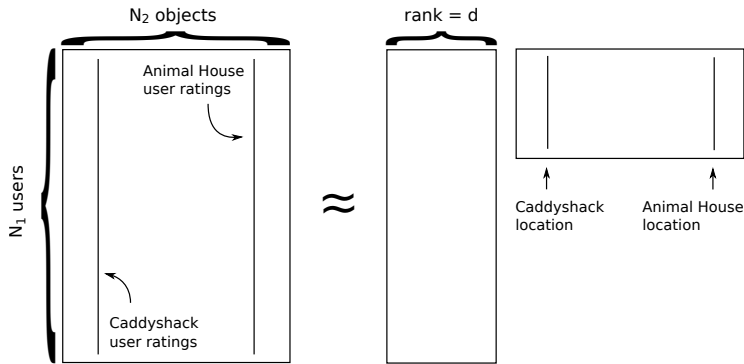
MATRIX FACTORIZATION



We will define a model for learning a low-rank factorization of M . It should:

1. Account for the fact that most values in M are missing
2. Be low-rank, where $d \ll \min\{N_1, N_2\}$ (e.g., $d \approx 10$)
3. Learn a location $u_i \in \mathbb{R}^d$ for user i and $v_j \in \mathbb{R}^d$ for object j

LOW-RANK MATRIX FACTORIZATION

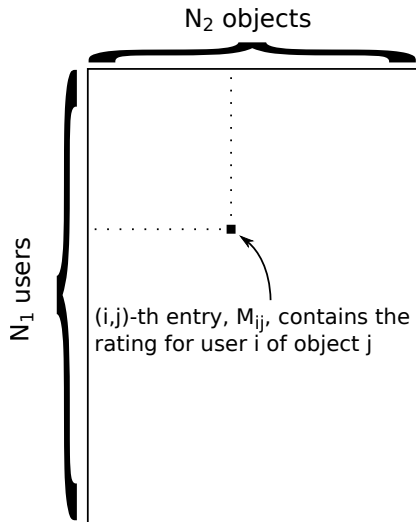


Why learn a low-rank matrix?

- ▶ We think that many columns should look similar. For example, movies like *Caddyshack* and *Animal House* should have **correlated** ratings.
- ▶ Low-rank means that the N_1 -dimensional columns don't "fill up" \mathbb{R}^{N_1} .
- ▶ Since $> 95\%$ of values may be missing, a low-rank restriction gives hope for filling in missing data because it models correlations.

PROBABILISTIC MATRIX FACTORIZATION

SOME NOTATION



- Let the set Ω contain the pairs (i,j) that are observed. In other words,

$$\Omega = \{(i,j) : M_{ij} \text{ is measured}\}.$$

So $(i,j) \in \Omega$ if user i rated object j .

- Let Ω_{u_i} be the index set of objects rated by user i .
- Let Ω_{v_j} be the index set of users who rated object j .

PROBABILISTIC MATRIX FACTORIZATION

Generative model

For N_1 users and N_2 objects, generate

User locations: $u_i \sim N(0, \lambda^{-1}I), \quad i = 1, \dots, N_1$

Object locations: $v_j \sim N(0, \lambda^{-1}I), \quad j = 1, \dots, N_2$

Given these locations the distribution on the data is

$$M_{ij} \sim N(u_i^T v_j, \sigma^2), \quad \text{for each } (i, j) \in \Omega.$$

Comments:

- ▶ Since M_{ij} is a rating, the Gaussian assumption is clearly wrong.
- ▶ However, the Gaussian is a convenient assumption. The algorithm will be easy to implement, and the model works well.

MODEL INFERENCE

Q: There are many missing values in the matrix M . Do we need some sort of EM algorithm to learn all the u 's and v 's?

- ▶ Let M_o be the part of M that is observed and M_m the missing part. Then

$$p(M_o|U, V) = \int p(M_o, M_m|U, V) dM_m.$$

- ▶ Recall that EM is a **tool** for maximizing $p(M_o|U, V)$ over U and V .
- ▶ Therefore, it is only needed when
 1. $p(M_o|U, V)$ is hard to maximize,
 2. $p(M_o, M_m|U, V)$ is easy to work with, and
 3. the posterior $p(M_m|M_o, U, V)$ is known.

A: If $p(M_o|U, V)$ doesn't present any problems for inference, then no.

(Similar conclusion in our MAP scenario, maximizing $p(M_o, U, V)$.)

MODEL INFERENCE

To test how hard it is to maximize $p(M_o, U, V)$ over U and V , we have to

1. Write out the joint likelihood
2. Take its natural logarithm
3. Take derivatives with respect to u_i and v_j and see if we can solve

The joint likelihood of $p(M_o, U, V)$ can be factorized as follows:

$$p(M_o, U, V) = \underbrace{\left[\prod_{(i,j) \in \Omega} p(M_{ij} | u_i, v_j) \right]}_{\text{conditionally independent likelihood}} \times \underbrace{\left[\prod_{i=1}^{N_1} p(u_i) \right] \left[\prod_{j=1}^{N_2} p(v_j) \right]}_{\text{independent priors}}.$$

By definition of the model, we can write out each of these distributions.

MAXIMUM A POSTERIORI

Log joint likelihood and MAP

The MAP solution for U and V is the maximum of the log joint likelihood

$$U_{\text{MAP}}, V_{\text{MAP}} = \arg \max_{U, V} \sum_{(i,j) \in \Omega} \ln p(M_{ij} | u_i, v_j) + \sum_{i=1}^{N_1} \ln p(u_i) + \sum_{j=1}^{N_2} \ln p(v_j)$$

Calling the MAP objective function \mathcal{L} , we want to maximize

$$\mathcal{L} = - \sum_{(i,j) \in \Omega} \frac{1}{2\sigma^2} \|M_{ij} - u_i^T v_j\|^2 - \sum_{i=1}^{N_1} \frac{\lambda}{2} \|u_i\|^2 - \sum_{j=1}^{N_2} \frac{\lambda}{2} \|v_j\|^2 + \text{constant}$$

The squared terms appear because all distributions are Gaussian.

MAXIMUM A POSTERIORI

To update each u_i and v_j , we take the derivative of \mathcal{L} and set to zero.

$$\nabla_{u_i} \mathcal{L} = \sum_{j \in \Omega_{u_i}} \frac{1}{\sigma^2} (M_{ij} - u_i^T v_j) v_j - \lambda u_i = 0$$

$$\nabla_{v_j} \mathcal{L} = \sum_{i \in \Omega_{v_j}} \frac{1}{\sigma^2} (M_{ij} - v_j^T u_i) u_i - \lambda v_j = 0$$

We can solve for each u_i and v_j individually (therefore EM isn't required),

$$u_i = \left(\lambda \sigma^2 I + \sum_{j \in \Omega_{u_i}} v_j v_j^T \right)^{-1} \left(\sum_{j \in \Omega_{u_i}} M_{ij} v_j \right)$$

$$v_j = \left(\lambda \sigma^2 I + \sum_{i \in \Omega_{v_j}} u_i u_i^T \right)^{-1} \left(\sum_{i \in \Omega_{v_j}} M_{ij} u_i \right)$$

However, we can't solve for all u_i and v_j at once to find the MAP solution. Thus, as with K-means and the GMM, we use a coordinate ascent algorithm.

PROBABILISTIC MATRIX FACTORIZATION

MAP inference coordinate ascent algorithm

Input: An incomplete ratings matrix M , as indexed by the set Ω . Rank d .

Output: N_1 user locations, $u_i \in \mathbb{R}^d$, and N_2 object locations, $v_j \in \mathbb{R}^d$.

Initialize each v_j . For example, generate $v_j \sim N(0, \lambda^{-1}I)$.

for each iteration **do**

for $i = 1, \dots, N_1$ **update user location**

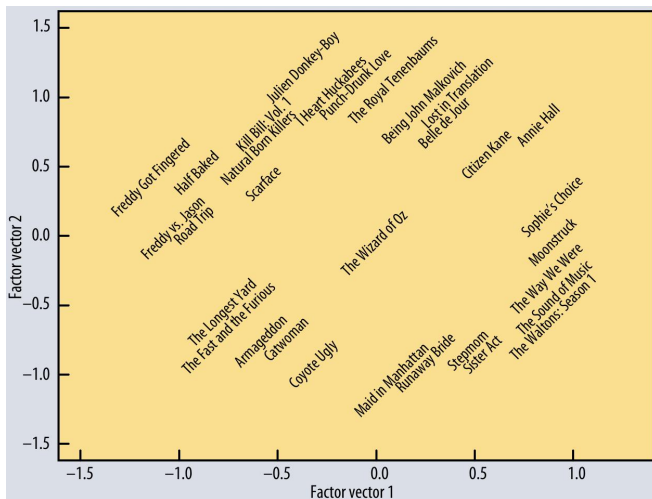
$$u_i = \left(\lambda \sigma^2 I + \sum_{j \in \Omega_{u_i}} v_j v_j^T \right)^{-1} \left(\sum_{j \in \Omega_{u_i}} M_{ij} v_j \right)$$

for $j = 1, \dots, N_2$ **update object location**

$$v_j = \left(\lambda \sigma^2 I + \sum_{i \in \Omega_{v_j}} u_i u_i^T \right)^{-1} \left(\sum_{i \in \Omega_{v_j}} M_{ij} u_i \right)$$

Predict that user i rates object j as $u_i^T v_j$ rounded to closest rating option

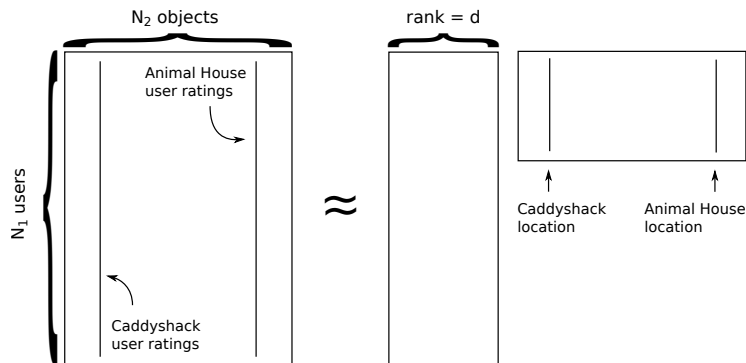
ALGORITHM OUTPUT FOR MOVIES



Hard to show in \mathbb{R}^2 , but we get locations for movies and users. Their relative locations captures relationships (that can be hard to explicitly decipher).

¹ Koren, Y., Robert B., and Volinsky, C.. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.

ALGORITHM OUTPUT FOR MOVIES

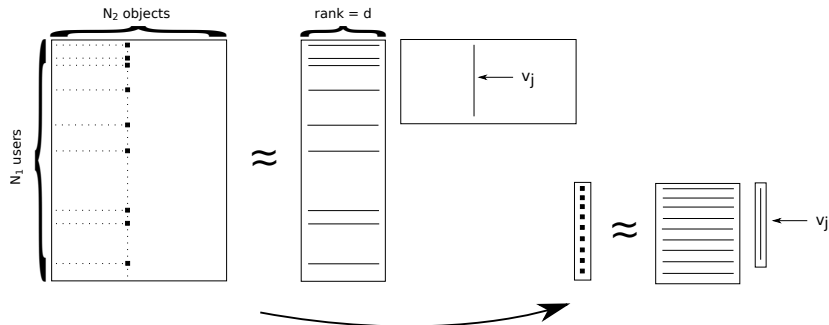


Returning to *Animal House* (j) and *Caddyshack* (j'), it's easy to understand the relationship between their locations v_j and $v_{j'}$:

- ▶ For these two movies to have similar rating patterns, their respective v 's must be similar (i.e., close to each other in \mathbb{R}^d).
- ▶ The same holds for users who have similar tastes across movies.

MATRIX FACTORIZATION AND RIDGE REGRESSION

MATRIX FACTORIZATION AND RIDGE REGRESSION



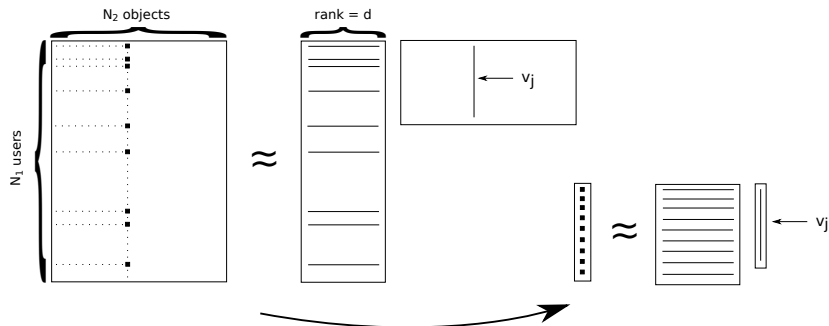
There is a close relationship between this algorithm and ridge regression.

- ▶ Think from the perspective of object location v_j .
- ▶ Minimize the sum squared error $\frac{1}{\sigma^2} (M_{ij} - u_i^T v_j)^2$ with penalty $\lambda \|v_j\|^2$.
- ▶ This is ridge regression for v_j , as the update also shows:

$$v_j = \left(\lambda \sigma^2 I + \sum_{i \in \Omega_{v_j}} u_i u_i^T \right)^{-1} \left(\sum_{i \in \Omega_{v_j}} M_{ij} u_i \right)$$

- ▶ So this model is a set of $N_1 + N_2$ coupled ridge regression problems.

MATRIX FACTORIZATION AND LEAST SQUARES



We can also connect it to least squares.

- ▶ Remove the Gaussian priors on u_i and v_j . The update for, e.g., v_j is then

$$v_j = \left(\sum_{i \in \Omega_{v_j}} u_i u_i^T \right)^{-1} \left(\sum_{i \in \Omega_{v_j}} M_{ij} u_i \right)$$

- ▶ This is the least squares solution. It requires that every user has rated at least d objects and every object is rated by at least d users.
- ▶ This probably isn't the case, so we see why a prior is *necessary* here.

ColumbiaX: Machine Learning

Lecture 18

Prof. John Paisley

Department of Electrical Engineering
& Data Science Institute

Columbia University

TOPIC MODELING

MODELS FOR TEXT DATA



Given text data we want to:

- ▶ Organize
- ▶ Visualize
- ▶ Summarize
- ▶ Search
- ▶ Predict
- ▶ Understand

Topic models allow us to

1. Discover themes in text
2. Annotate documents
3. Organize, summarize, etc.

BUSINESS DAY

A Digital Shift on Health Data Swells Profits in an Industry

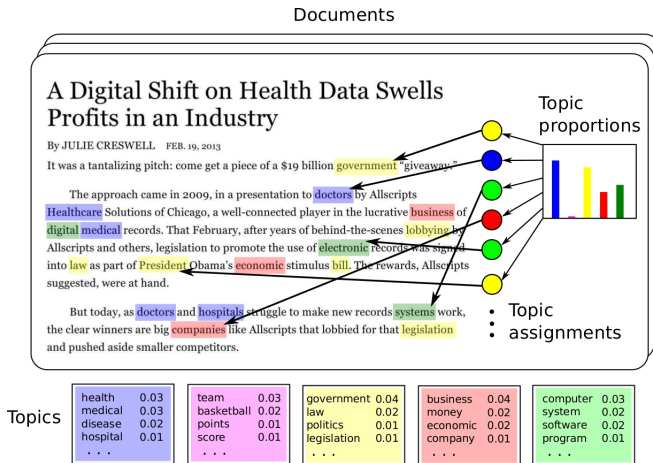
By JULIE CRESWELL FEB. 19, 2013

It was a tantalizing pitch: come get a piece of a \$19 billion government “giveaway.”

The approach came in 2009, in a presentation to doctors by Allscripts Healthcare Solutions of Chicago, a well-connected player in the lucrative business of digital medical records. That February, after years of behind-the-scenes lobbying by Allscripts and others, legislation to promote the use of electronic records was signed into law as part of President Obama’s economic stimulus bill. The rewards, Allscripts suggested, were at hand.

But today, as doctors and hospitals struggle to make new records systems work, the clear winners are big companies like Allscripts that lobbied for that legislation and pushed aside smaller competitors.

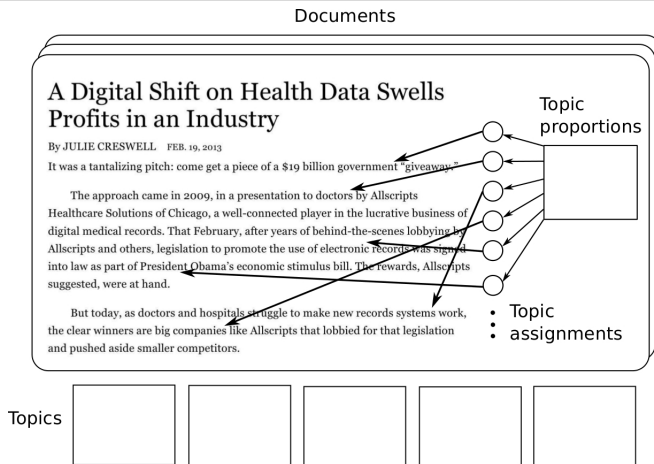
TOPIC MODELING



A probabilistic topic model

- ▶ Learns distributions on words called “topics” shared by documents
- ▶ Learns a distribution on topics for each document
- ▶ Assigns every word in a document to a topic

TOPIC MODELING



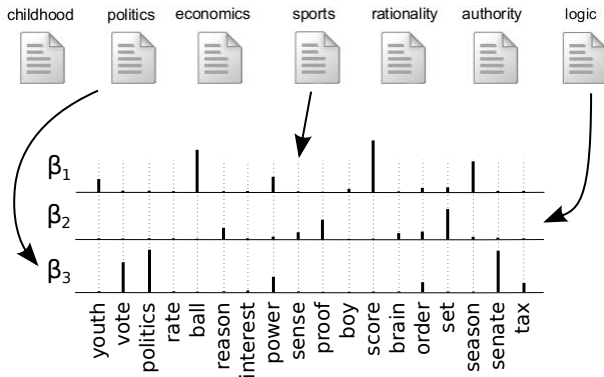
However, none of these things are known in advance and must be learned

- ▶ Each document is treated as a “bag of words”
- ▶ Need to define (1) a model, and (2) an algorithm to learn it
- ▶ We will review the standard topic model, but won’t cover inference

LATENT DIRICHLET ALLOCATION

There are two essential ingredients to latent Dirichlet allocation (LDA).

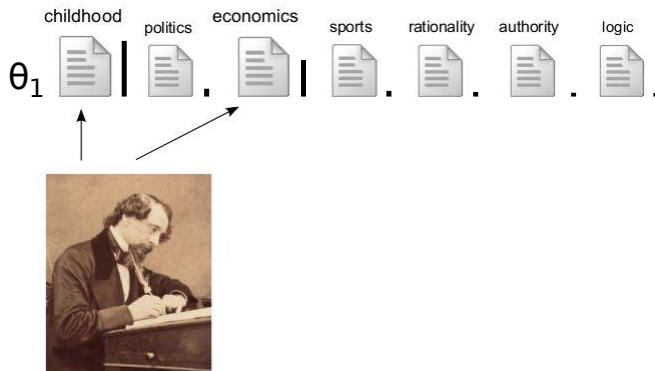
1. A collection of distributions on words (topics).
2. A distribution on topics for each document.



LATENT DIRICHLET ALLOCATION

There are two essential ingredients to latent Dirichlet allocation (LDA).

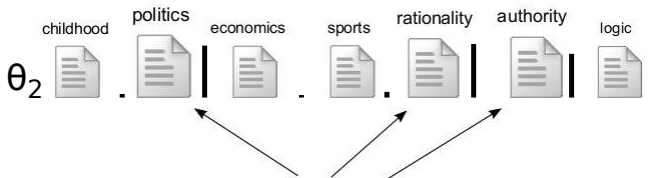
1. A collection of distributions on words (topics).
2. A distribution on topics for each document.



LATENT DIRICHLET ALLOCATION

There are two essential ingredients to latent Dirichlet allocation (LDA).

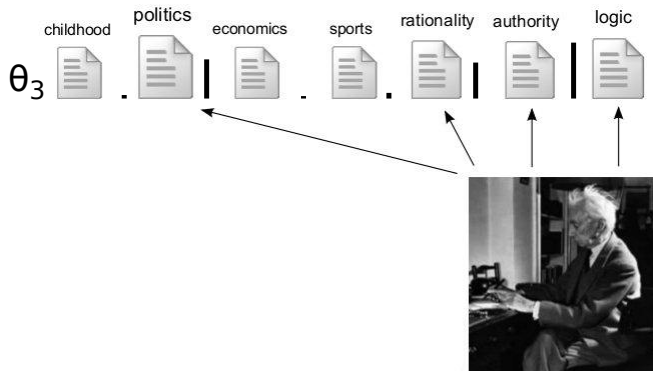
1. A collection of distributions on words (topics).
2. A distribution on topics for each document.



LATENT DIRICHLET ALLOCATION

There are two essential ingredients to latent Dirichlet allocation (LDA).

1. A collection of distributions on words (topics).
2. A distribution on topics for each document.



LATENT DIRICHLET ALLOCATION

There are two essential ingredients to latent Dirichlet allocation (LDA).

1. A collection of distributions on words (topics).
2. A distribution on topics for each document.

The generative process for LDA is:

1. Generate each topic, which is a distribution on words

$$\beta_k \sim \text{Dirichlet}(\gamma), \quad k = 1, \dots, K$$

2. For each document, generate a distribution on topics

$$\theta_d \sim \text{Dirichlet}(\alpha), \quad d = 1, \dots, D$$

3. For the n th word in the d th document,

- a) Allocate the word to a topic, $c_{dn} \sim \text{Discrete}(\theta_d)$
- b) Generate the word from the selected topic, $x_{dn} \sim \text{Discrete}(\beta_{c_{dn}})$

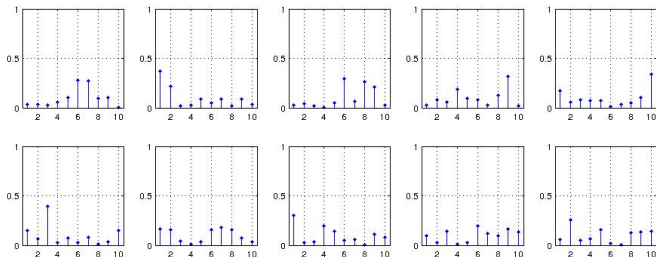
DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.

$\gamma = 1$

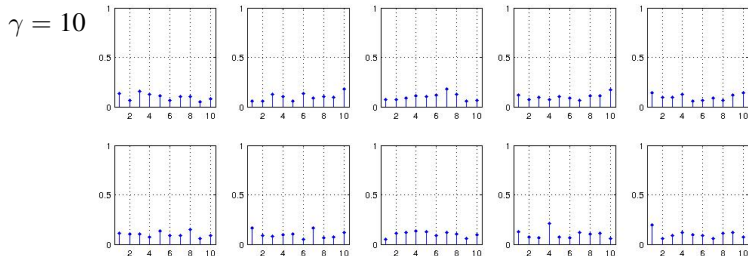


DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.

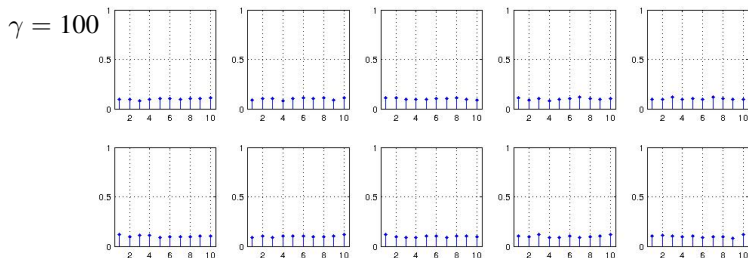


DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.



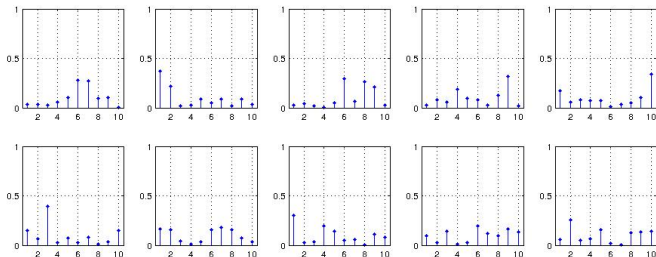
DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.

$\gamma = 1$

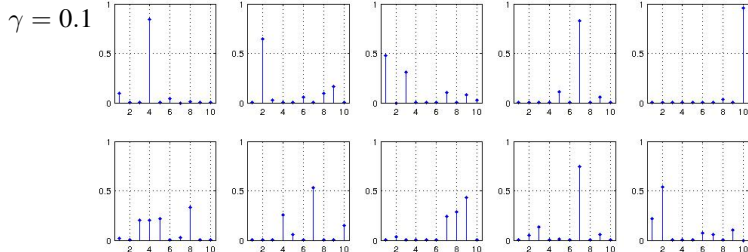


DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.

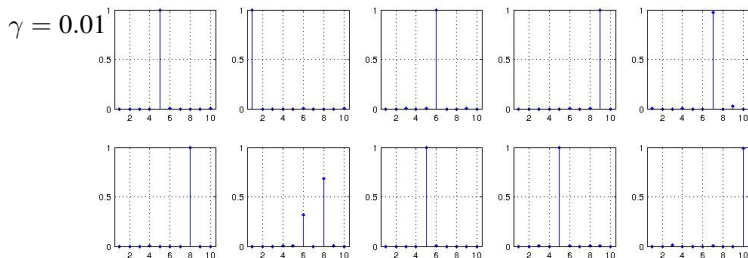


DIRICHLET DISTRIBUTION

A continuous distribution on discrete probability vectors. Let β_k be a probability vector and γ a positive parameter vector,

$$p(\beta_k|\gamma) = \frac{\Gamma(\sum_v \gamma_v)}{\prod_{v=1}^V \Gamma(\gamma_v)} \prod_{v=1}^V \beta_{k,v}^{\gamma_v-1}$$

This defines the Dirichlet distribution. Some examples of β_k generated from this distribution for a constant value of γ and $V = 10$ are given below.



LDA OUTPUT

The New York Times

music band songs rock album jazz pop song singer night	book life novel story books man stories love children family	art museum show exhibition artist artists paintings painting century works	game knicks nets points team season play games night coach	show film television movie series says life man character know
theater play production show stage street broadway director musical directed	clinton bush campaign gore political republican dole presidential senator house	stock market percent fund investors funds companies stocks investment trading	restaurant sauce menu food dishes street dining dinner chicken served	budget tax governor county mayor billion taxes plan legislature fiscal

LDA outputs two main things:

1. A set of distributions on words (topics). Shown above are ten topics from NYT data. We list the ten words with the highest probability.
2. A distribution on topics for each document (not shown). This indicates its thematic breakdown and provides a compact representation.

LDA AND MATRIX FACTORIZATION

Q: For a particular document, what is $P(x_{dn} = i | \beta, \theta_d)$?

A: Find this by integrating out the cluster assignment,

$$\begin{aligned} P(x_{dn} = i | \beta, \theta) &= \sum_{k=1}^K P(x_{dn} = i, c_{dn} = k | \beta, \theta_d) \\ &= \sum_{k=1}^K \underbrace{P(x_{dn} = i, | \beta, c_{dn} = k)}_{= \beta_{ki}} \underbrace{P(c_{dn} = k | \theta_d)}_{= \theta_{dk}} \end{aligned}$$

Let $B = [\beta_1, \dots, \beta_K]$ and $\Theta = [\theta_1, \dots, \theta_D]$, then $P(x_{dn} = i | \beta, \theta) = (B\Theta)_{id}$

In other words, we can read the probabilities from a matrix formed by taking the product of two matrices that have nonnegative entries.

NONNEGATIVE MATRIX FACTORIZATION

NONNEGATIVE MATRIX FACTORIZATION

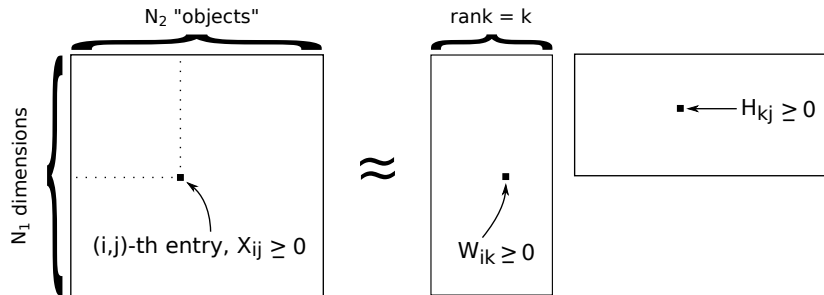
LDA can be thought of as an instance of nonnegative matrix factorization.

- ▶ It is a probabilistic model.
- ▶ Inference involves techniques not taught in this course.

We will discuss two other related models and their algorithms. These two models are called *nonnegative matrix factorization* (NMF)

- ▶ They can be used for the same tasks as LDA
- ▶ Though “nonnegative matrix factorization” is a general technique, “NMF” usually just refers to the following two methods.

NONNEGATIVE MATRIX FACTORIZATION



We use notation and think about the problem slightly differently from PMF

- ▶ Data X has nonnegative entries. None missing, but likely many zeros.
- ▶ The learned factorization W and H also have nonnegative entries.
- ▶ The value $X_{ij} \approx \sum_k W_{ik}H_{kj}$, but we won't write this with vector notation
- ▶ Later we interpret the output in terms of columns of W and H .

NONNEGATIVE MATRIX FACTORIZATION

What are some data modeling problems that can constitute X ?

- ▶ Text data:

- ▶ Word term frequencies
- ▶ X_{ij} contains the number of times word i appears in document j .

- ▶ Image data:

- ▶ Face identification data sets
- ▶ Put each *vectorized* $N \times M$ image of a face on a *column* of X .

- ▶ Other discrete grouped data:

- ▶ Quantize *continuous* sets of features using K-means
- ▶ X_{ij} counts how many times group j uses cluster i .
- ▶ For example: group = song, features = $d \times n$ spectral information matrix

TWO OBJECTIVE FUNCTIONS

NMF minimizes one of the following two objective functions over W and H .

Choice 1: Squared error objective

$$\|X - WH\|^2 = \sum_i \sum_j (X_{ij} - (WH)_{ij})^2$$

Choice 2: Divergence objective

$$D(X \| WH) = - \sum_i \sum_j [X_{ij} \ln(WH)_{ij} - (WH)_{ij}]$$

- ▶ Both have the constraint that W and H contain nonnegative values.
- ▶ NMF uses a fast, simple algorithm for optimizing these two objectives.

MINIMIZATION AND MULTIPLICATIVE ALGORITHMS¹

Recall what we should look for in minimizing an objective “ $\min_h F(h)$ ”:

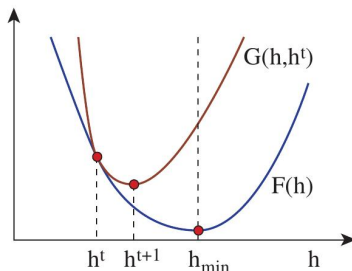
1. A way to generate a sequence of values h^1, h^2, \dots , such that

$$F(h^1) \geq F(h^2) \geq F(h^3) \geq \dots$$

2. Convergence of the sequence to a local minimum of F

The following algorithms fulfill these requirements. In this case:

- ▶ Minimization is done via an “auxiliary function.”
- ▶ Leads to a “multiplicative algorithm” for W and H .
- ▶ We’ll skip details (see reference).



¹For details, see D.D. Lee and H.S. Seung (2001). “Algorithms for non-negative matrix factorization.” *Advances in Neural Information Processing Systems*.

MULTIPLICATIVE UPDATE FOR $\|X - WH\|^2$

Problem

$$\min \sum_{ij} (X_{ij} - (WH)_{ij})^2 \quad \text{subject to } W_{ik} \geq 0, H_{kj} \geq 0.$$

Algorithm

- ▶ Randomly initialize H and W with nonnegative values.
- ▶ Iterate the following, first for all values in H , then all in W :

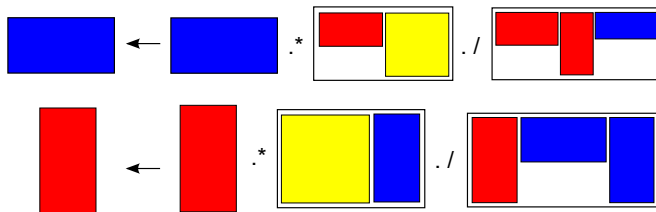
$$H_{kj} \leftarrow H_{kj} \frac{(W^T X)_{kj}}{(W^T W H)_{kj}},$$

$$W_{ik} \leftarrow W_{ik} \frac{(X H^T)_{ik}}{(W H H^T)_{ik}},$$

until the change in $\|X - WH\|^2$ is “small.”

A visualization that may be helpful. Use the color-coded definition above.

- ▶ Use element-wise multiplication/division across three columns below.
- ▶ Use matrix multiplication within each outlined box.



Probabilistically, the squared error penalty implies a Gaussian distribution,

$$X_{ij} \sim N(\sum_k W_{ik} H_{kj}, \sigma^2)$$

Since $X_{ij} \geq 0$ (and often isn't continuous), we are making an incorrect modeling assumption. Nevertheless, as with PMF it still works well.

MULTIPLICATIVE UPDATE FOR $D(X\|WH)$

Problem

$$\min \sum_{ij} \left[X_{ij} \ln \frac{1}{(WH)_{ij}} + (WH)_{ij} \right] \quad \text{subject to } W_{ik} \geq 0, H_{kj} \geq 0.$$

Algorithm

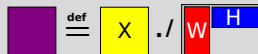
- ▶ Randomly initialize H and W with nonnegative values.
- ▶ Iterate the following, first for all values in H , then all in W :

$$H_{kj} \leftarrow H_{kj} \frac{\sum_i W_{ik} X_{ij} / (WH)_{ij}}{\sum_i W_{ik}},$$

$$W_{ik} \leftarrow W_{ik} \frac{\sum_j H_{kj} X_{ij} / (WH)_{ij}}{\sum_j H_{kj}},$$

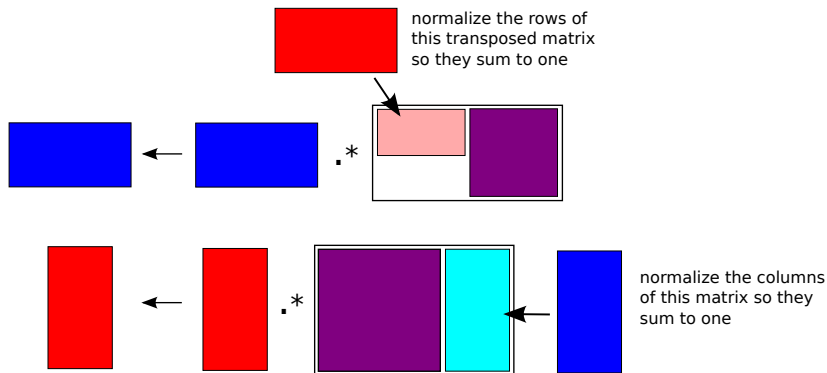
until the change in $D(X\|WH)$ is “small.”

VISUALIZATION



Visualizing the update for the divergence penalty is more complicated.

- ▶ Use the color-coded definition above.
- ▶ “Purple” is the data matrix “dot-divided” by the approximation of it.



MAXIMUM LIKELIHOOD

The maximum likelihood interpretation of the divergence penalty is more interesting than for the squared error penalty.

If we model the data as independent Poisson random variables

$$X_{ij} \sim \text{Pois}((WH)_{ij}), \quad \text{Pois}(x|\lambda) = \frac{\lambda^x}{x!} e^{-\lambda}, \quad x \in \{0, 1, 2, \dots\},$$

then the negative divergence penalty is maximum likelihood for W and H .

$$\begin{aligned} -D(X||WH) &= \sum_{ij} [X_{ij} \ln(WH)_{ij} - (WH)_{ij}] \\ &= \sum_{ij} \ln P(X_{ij}|W, H) + \text{constant} \end{aligned}$$

We use: $P(X|W, H) = \prod_{ij} P(X_{ij}|W, H) = \prod_{ij} \text{Pois}(X_{ij}|(WH)_{ij})$.

NMF AND TOPIC MODELING

As discussed, NMF can be used for topic modeling. In fact, one can show that the divergence penalty is closely related mathematically to LDA.

Step 1. Form the term-frequency matrix X . (X_{ij} = # times word i in doc j)

Step 2. Run NMF to learn W and H using $D(X||WH)$ penalty

Step 3. As an added step, after Step 2 is complete, for $k = 1, \dots, K$

1. Set $a_k = \sum_i W_{ik}$
2. Divide W_{ik} by a_k for all i
3. Multiply H_{kj} by a_k for all j

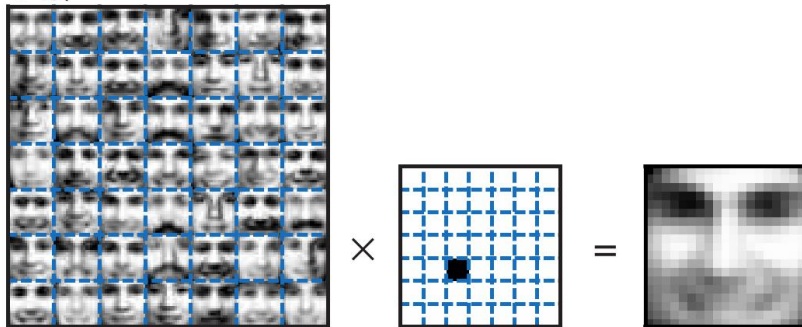
Notice that this does not change the matrix multiplication WH .

Interpretation: The k th *column* of W can be interpreted as the k th *topic*. The j th *column* of H can be interpreted as how much document j uses each topic.

NMF AND FACE MODELING

For face modeling, put the face images along the columns of X and factorize. Show columns of W as image. Compare this with K-means and SVD.

VQ

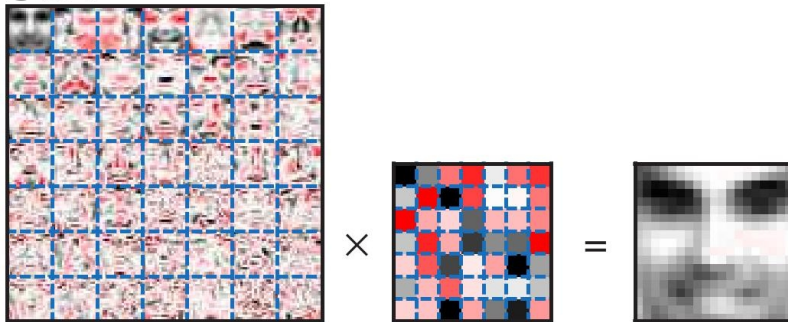


K-means (i.e., VQ): Equivalent to each column of H having a single 1.
K-means learns averages of full faces.

NMF AND FACE MODELING

For face modeling, put the face images along the columns of X and factorize. Show columns of W as image. Compare this with K-means and SVD.

SVD



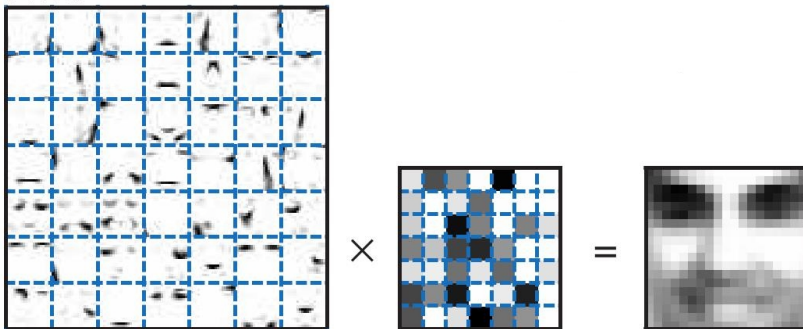
SVD: Finds the singular value decomposition of X .

Results not interpretable because of \pm values and orthogonality constraint

NMF AND FACE MODELING

For face modeling, put the face images along the columns of X and factorize. Show columns of W as image. Compare this with K-means and SVD.

NMF



NMF learns a “parts-based” representation. Each column captures something interpretable. This is a result of the nonnegativity constraint.