Non-Negative Matrix Factorization (NMF)

(and Interpretable Topic Modeling)

Overview

- What is NMF?
  - Unbaking a cake
- Popular Applications
  - Document Clustering (i.e., Topic Modeling)
- Deeper Dive into Mechanics
  - Gradient Descent
  - Alternating Least Squares (ALS)
- NMF vs. PCA

Problem Motivation

- Soft clustering
  - Clustering where each observation can have partial membership in each cluster
- Identify latent features
  - Identify topics in a corpus of text
  - Identity parts of faces in facial recognition

Example: Unbaking a cake
(i.e., finding its recipe)

- Given a cake, we **cannot** directly observe its ingredients (flour ***(F)*** and sugar ***(S)***)
  - (the recipe is a closely guarded secret!)

$$\begin{matrix}\text{recipe} \\ \text{a. k. a.,} \\ \textbf{cake as ingredients}\end{matrix} = W = (w_F \quad w_S)$$

Example: Unbaking a cake
(i.e., finding its recipe)

- But we **can** directly measure its **macro nutrients (carbohydrates *(c)*, proteins *(p)*, fats *(f)*)**
  - E.g., the cake is made of 50g of carbohydrates, 10g of proteins, and 3g of fats

$$\textbf{cake} \text{ as } \textbf{macro nutrients} = (50 \quad 10 \quad 3)$$

$$V = (v_c \quad v_p \quad v_f)$$

Example: Unbaking a cake

(i.e., finding its recipe)

- Ingredients are themselves made of macro nutrients (for which we know the proportions):
  - 1 cup flour is made of 20g of carbohydrates, 5g of proteins, and 1g of fats
  - 1 cup sugar is made of 10g of carbohydrates, 0g of proteins, and 1g of fats

$$\textbf{ingredients}\ \text{as}\ \textbf{macro nutrients} = \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix}$$

$$H = \begin{pmatrix} h_{F,c} & h_{F,p} & h_{F,f} \\ h_{S,c} & h_{S,p} & h_{S,f} \end{pmatrix}$$

Example: Unbaking a cake
(i.e., finding its recipe)

- Figuring out the recipe is about solving the following equation:

cake as **ingredients** (recipe) · **ingredients** as **macro nutrients** ≈ **cake** as **macro nutrients**

$$W \cdot H \approx V$$

$$(w_F \quad w_S) \cdot \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix} \approx (50 \quad 10 \quad 3)$$

Generalization: Unbaking cakes
(i.e., finding recipes)

$$\begin{matrix} cake\boldsymbol{s} \\ as \\ ingredients \\ (recipe) \end{matrix} \cdot \begin{matrix} ingredients \\ as \\ \textit{macro nutrients} \end{matrix} \approx \begin{matrix} cake\boldsymbol{s} \\ as \\ \textit{macro nutrients} \end{matrix}$$

$$W \cdot H \approx V$$

$$\begin{pmatrix} w_{\mathbf{1},F} & w_{\mathbf{1},S} \\ w_{\mathbf{2},F} & w_{\mathbf{2},S} \\ w_{\mathbf{3},F} & w_{\mathbf{3},S} \\ \vdots & \vdots \end{pmatrix} \cdot \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \\ 30 & 5 & 2 \\ 25 & 3 & 3 \\ \vdots & \vdots & \vdots \end{pmatrix}$$

Generalization: Unbaking cakes
(i.e., finding recipes)

- Given H and V, can we estimate W?

$$\begin{pmatrix} w_{1,F} & w_{1,S} \\ w_{2,F} & w_{2,S} \\ w_{3,F} & w_{3,S} \\ \vdots & \vdots \end{pmatrix} \cdot \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \\ 30 & 5 & 2 \\ 25 & 3 & 3 \\ \vdots & \vdots & \vdots \end{pmatrix}$$

- Can perform OLS minimization to find each recipe. E.g,

$$\begin{pmatrix} w_{1,F} & w_{1,S} \end{pmatrix} \cdot \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \end{pmatrix}$$

Generalization: Unbaking cakes

(i.e., finding recipes)

$$\begin{pmatrix} w_{1,F} & w_{1,S} \end{pmatrix} \cdot \begin{pmatrix} 20 & 5 & 1 \\ 10 & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \end{pmatrix}$$

$$20w_{1,F} + 10w_{1,S} = \hat{v}_c \approx v_c = 50$$
$$5w_{1,F} + 0w_{1,S} = \hat{v}_p \approx v_p = 10$$
$$1w_{1,F} + 1w_{1,S} = \hat{v}_f \approx v_f = 3$$

$$\underset{w=\begin{pmatrix} w_{1,F} & w_{1,S} \end{pmatrix}}{\operatorname{argmin}} \|v - \hat{v}\|^2$$

Unbaking cakes and finding ingredients

- **Given W and V, can we estimate H?**

$$\begin{pmatrix} 2 & 1 \\ 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \end{pmatrix} \cdot \begin{pmatrix} h_{F,c} & h_{F,p} & h_{F,f} \\ h_{S,c} & h_{S,p} & h_{S,f} \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \\ 30 & 5 & 2 \\ 25 & 3 & 3 \\ \vdots & \vdots & \vdots \end{pmatrix}$$

$$2h_{F,c} + 1h_{F,c} = \hat{v}_{1,c} \approx v_{1,c} = 50$$
$$1w_{1,F} + 1w_{1,S} = \hat{v}_{1,p} \approx v_{1,p} = 10$$
$$1w_{1,F} + 2w_{1,S} = \hat{v}_{1,f} \approx v_{1,f} = 3$$
(…continued for all observations)

$$\underset{H=\begin{pmatrix} h_{F,c} & h_{F,p} & h_{F,f} \\ h_{S,c} & h_{S,p} & h_{S,f} \end{pmatrix}}{\operatorname{argmin}} \left\| V - \hat{V} \right\|^2$$

Unbaking cakes and finding ingredients

- **What is you don't know W and H?**

$$\begin{pmatrix} w_{1,F} & w_{1,S} \\ w_{2,F} & w_{2,S} \\ w_{3,F} & w_{3,S} \\ \vdots & \vdots \end{pmatrix} \cdot \begin{pmatrix} h_{F,c} & h_{F,p} & h_{F,f} \\ h_{S,c} & h_{S,p} & h_{S,f} \end{pmatrix} \approx \begin{pmatrix} 50 & 10 & 3 \\ 30 & 5 & 2 \\ 25 & 3 & 3 \\ \vdots & \vdots & \vdots \end{pmatrix}$$

- **NMF can learn what the ingredients and the proportions are from all these cakes**

What is Non-Negative Matrix Factorization (NMF)?

Matrix $V_{m \times n}$ where each entry $v_{i,j} \geq 0$

$$\underset{m \times \boldsymbol{r}}{W} \times \underset{\boldsymbol{r} \times n}{H} = \underset{m \times n}{V}$$

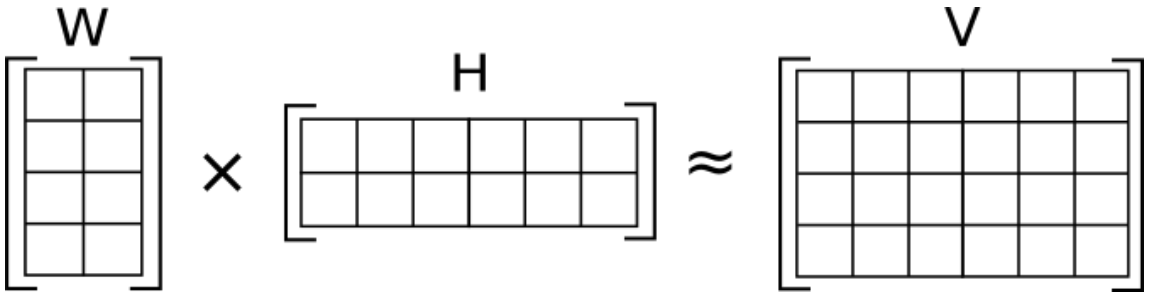
(also, $w_{i,j} \geq 0$ and $h_{i,j} \geq 0$)

(non-negativity constraint, unlike the decompositions we've looked at so far)

What is Non-Negative Matrix Factorization (NMF)?

$$W \quad \times \quad H \quad \approx \quad V$$

$$m \times \boldsymbol{r} \qquad \boldsymbol{r} \times n \qquad m \times n$$

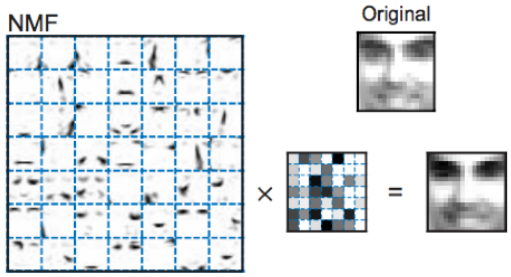


- $\boldsymbol{r}$ is set by you ($r < \min(m, n)$)
- Cannot be solved analytically, so approximated numerically

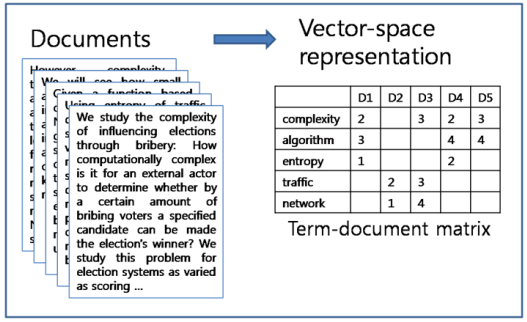
- When $r > \boldsymbol{min(m, n)}$, we can perfectly recreate $\boldsymbol{V}$ as $\boldsymbol{W \cdot H}$
- Otherwise, some sort of data compression is happening

Popular Applications

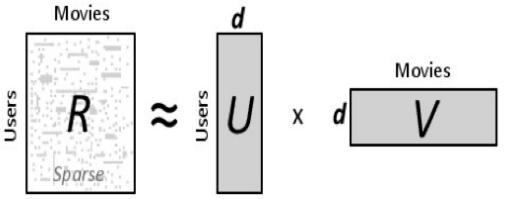
- Computer Visioning
  - Identify / classifying objects
  - Generally reducing feature space of images



- Topic Modeling
  - A.k.a., Document Clustering
  - Today!



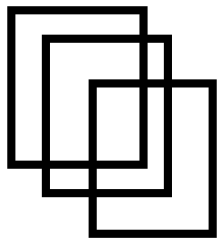
- Recommender systems
  - In just a few days!



Topic Modeling with NMF

500 documents

10,000 words



cakes ➡ documents, e.g., term frequencies

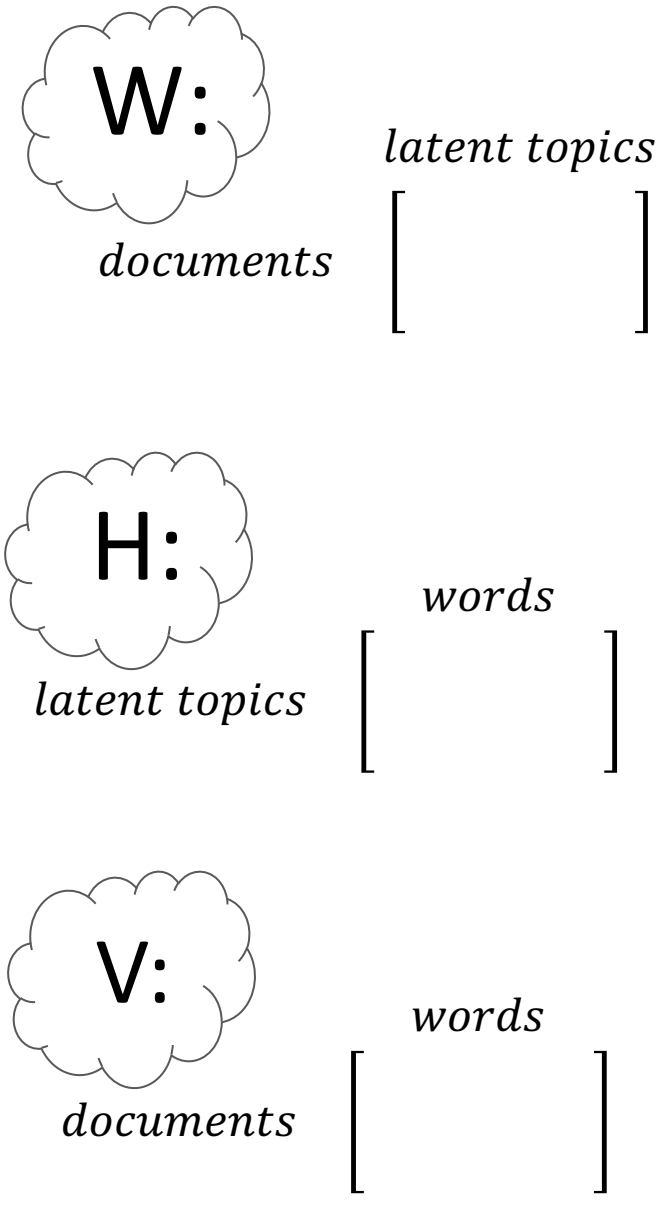
macro nutrients ➡ words

(latent) ingredients ➡ (latent) topics

$$V \approx W \cdot H$$

cake as **ingredients** · **ingredients** as **macro nutrients** ≈ **cake** as **macro nutrients**

document as **topics** · **topics** as **words** ≈ **document** as **words**



- In looking at the dimensions of W, we notice that the number of rows remain the same, as we would expect. Thus, each row of W must represent some information about the corresponding row in V.
- Think of the column W as latent topics where the higher the word's cell value, the higher the word's rank for that latent feature.
- The same can be said about H: the number of columns between V and H are the same. Thus, each column of H must represent some information about the corresponding column in V.

Choosing k?

- Choosing k is more of an art than a science. We can look at how "good" of an approximation WH is for V and try to find the smallest k that makes it suitably small.
- At the end of the day, though, k, is likely going to be chosen based on intuition that you derive from inspecting the topics and possibly from some domain knowledge.

Example H Matrix

H =

k x m

"president"	"coach"	…	"team"	
0.3	5.1	...	4.2	Topic 1
10.3	1.07	...	0.08	Topic 2
..	...	...	...	⋮
...	...	...	...	
2.03	0.3	...	0.001	Topic k

Topic 1?

Topic 2?

Example H Matrix

H =

k x m

“president”	“coach”	…	“team”	
0.3	5.1	...	4.2	Topic 1
10.3	1.07	...	0.08	Topic 2
..	...	...	...	⋮
...	...	...	...	
2.03	0.3	...	0.001	Topic k

Topic 1: Sports?

Topic 2: Politics?

Mechanics

Minimize $\|V - WH\|^2$

with respect to W and H subject to $W, H \geq 0$

Steps

(1) Start with some random W and H (random, small positive values)

(2) Repeatedly adjust W and H to make RMSE smaller

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^T V)_{a\mu}}{(W^T W H)_{a\mu}} \qquad W_{ia} \leftarrow W_{ia} \frac{(V H^T)_{ia}}{(W H H^T)_{ia}}$$

($H_{a\mu}$ and W_{ia} are single entries in W and H)

- This is gradient descent! It's simple but can be slow. The convergence is sensitive to choice of step size.

(3) Stop when some threshold is met

- Decrease in RMSE, # of iterations, etc.

Mechanics – Alternating Least Squares (ALS)

Minimize $||V - WH||^2$
with respect to W and H subject to $W, H \geq 0$

Steps

(1) Randomly initialize(*) W and H to the appropriate shapes

(*) Initialization methods vary. Random initialization and k-means are common. One way: Initialize W to small, positive, random values.

(2) Repeat following

- Holding W fixed, update H by minimizing sum of squared errors. Ensure all $H \geq 0$. (Clip negative values)
- Holding H fixed, update W by minimizing sum of squared errors. Ensure all $W \geq 0$. (Clip negative values)

(3) Stop until convergence (i.e., some threshold is met)

– Decrease in RMSE, # of iterations, etc.

Note: ALS is "biconvex", i.e., it is not globally convex, so it will not necessarily find local optima (hence the strategic initializations)

NMF vs. PCA

PCA	NMF
Unsupervised dimensionality reduction	Unsupervised dimensionality reduction
Orthogonal vectors with positive and negative coefficients	Non-negative coefficients
"Holistic"; difficult to interpret	"Parts-based"; easier to interpret
Non-iterative	Iterative (the presented algorithm)

http://www.cs.cmu.edu/~02317/slides/lec_7.pdf#page=17