Non-Negative Matrix Factorization

Interpretable Topic Modeling

Cary Goltermann

Galvanize

2016

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Thinking About Topic Modeling

As a form of unsupervised learning, topic modeling attempts to extract an underlying structure from our data.

Specifically, we will endeavor to discover an underlying set of "topics" that describe high level ideas about our data well.

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Thinking About Topics

Consider having the term-frequency matrix for a corpus of documents, all coming from some sort of related source; articles from a newspaper possibly?

In trying to discover topics, or latent features, in these data, we might expect to find such overarching ones as: Sports, International, Arts and Leisure, etc.

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Topic Analysis Assumptions

1. The observations are well described by underlying topics. E.g. All sports articles will have similar sportsy words. In math, each topic has a corresponding distribution of words:

$$tf(word|topic)$$

Topic Analysis Assumptions

1. The observations are well described by underlying topics. E.g. All sports articles will have similar sportsy words. In math, each topic has a corresponding distribution of words:

$$tf(word|topic)$$

2. The words in a document can be represented by an appropriate combination of topics. E.g. An article about FIFA could be represented by the topics International and Sports. Math:

$$tf(word|doc) = \sum_{t \in T} tf(word|topic) \times w(t|doc)$$

Where T is the set of topics and w is some positive weight.

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

NMFactorization

Another way to mathematically express the topic modeling assumptions is through the equation:

$$X \approx W \cdot H, \qquad w_{i,j} \,\&\, h_{i,j} \geq 0$$

Or, X can be approximated by the dot product of two matrices. Hence factorization.

Thinking in Dimensions

How can we think about the values in each of these matrices, W and H. Let's look at their dimensions!

$$\underset{n\,x\,m}{X} \approx \underset{n\,x\,k}{W} \cdot \underset{k\,x\,m}{H}$$

We can easily rationalize that when the internal dimension, k, is $\geq m$ we can perfectly recreate X.

Thinking in Dimensions

How can we think about the values in each of these matrices, W and H. Let's look at their dimensions!

$$\underset{n \times m}{X} \approx \underset{n \times k}{W} \cdot \underset{k \times m}{H}$$

We can easily rationalize that when the internal dimension, k, is $\geq m$ we can perfectly recreate X.

So what happens when this dimension, k, is smaller than m?

A Smaller Number of Decomposed Dimensions

- In looking at the dimensions of the W matrix from our decomposition, we notice that the number of rows remains the same, as we would expect.
- Thus, the rows of W must represent some information about each row in X.
- When we use a smaller number of dimensions to represent our data in W, a.k.a $k < m$, we will necessarily find that some sort of data compression is happening.
- Or, in math, we are projecting our data onto a lower dimensioned basis.

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Topics as Latent Feature Bases

It, somewhat magically, turns out that when we conduct a factorization of this nature each of bases in the lower dimensional representation of X, a.k.a. W, can be viewed of as a latent feature.

These latent features are discovered as somewhat of a side-effect of projecting into a smaller number of dimensions, of performing some sort of compression.

The Latent Features are Topics

The values of the each row in the latent feature space correspond to their strength with the associated topic.

Example W Matrix

$W =$

n x k

	Topic 1	Topic 2	…	Topic k
Document 1	0.3	5.1	...	1.2
Document 2	9.76	0.04	...	2.7
	..	...	...	...
	...	...	...	...
Document n	0.06	0.3	...	0.001

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Identifying Topics

How, then, do we figure out what topics these latent feature bases correspond to? Consider that this is an unsupervised approach, so it's not like we're telling it to look for sports words.

To put "labels" on our latent features and identify them as topics we can do one of to things.

Identifying Topics

How, then, do we figure out what topics these latent feature bases correspond to? Consider that this is an unsupervised approach, so it's not like we're telling it to look for sports words.

To put "labels" on our latent features and identify them as topics we can do one of to things.

1. Look at the observations that load heavily on each topic and manually inspect them, trying to identify some commonalities, a.k.a. the latent features.

Identifying Topics

How, then, do we figure out what topics these latent feature bases correspond to? Consider that this is an unsupervised approach, so it's not like we're telling it to look for sports words.

To put "labels" on our latent features and identify them as topics we can do one of to things.

1. Look at the observations that load heavily on each topic and manually inspect them, trying to identify some commonalities, a.k.a. the latent features.
2. Inspect the H matrix and see what features contribute to each topic.

Identifying Topics

How, then, do we figure out what topics these latent feature bases correspond to? Consider that this is an unsupervised approach, so it's not like we're telling it to look for sports words.

To put "labels" on our latent features and identify them as topics we can do one of to things.

1. Look at the observations that load heavily on each topic and manually inspect them, trying to identify some commonalities, a.k.a. the latent features.
2. Inspect the H matrix and see what features contribute to each topic.

Inspecting the H Matrix

How would we inspect the H matrix, and why would doing so make sense?

Inspecting the H Matrix

How would we inspect the H matrix, and why would doing so make sense? Again, let's look at the dimensions!

$$\underset{n \times m}{X} \approx \underset{n \times k}{W} \cdot \underset{k \times m}{H}$$

- We see that the number of columns in X, m, is the same as in H.
- From this we can reason that the columns in H represent some information about the columns in X.
- More specifically, we can view the features in X as being the basis for our latent topics.

Example H Matrix

$H =$

k x m

Feature 1	Feature 2	…	Feature m	
0.3	5.1	...	4.2	Topic 1
10.3	1.07	...	0.08	Topic 2
..	...	...	...	⋮
...	...	...	...	
2.03	0.3	...	0.001	Topic k

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

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 $\rightarrow$ Sports?

Topic 2 $\rightarrow$ Politics??

Choosing k

Unfortunately, choosing k is more of an art than a science. We can look at how "good" of an approximation WH is for X 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.

Reflection

Time: 2 minutes

Exercise: Discuss, in groups of 2-3, how we can think about the rows in W and the columns in H.

Thinking About Matrix Factorizations

We have already explored two other factorization techniques, PCA and SVD. What's the difference between those and NMF?

Thinking About Matrix Factorizations

We have already explored two other factorization techniques, PCA and SVD. What's the difference between those and NMF?

Apart from the obvious that PCA and SVD decompose into three matrices and NMF only two; the main difference is the non-negativity constraint.

> *Note: the other, oft-forgotten difference is that the bases in NMF are not orthogonal like in PCA/SVD.*

Why All the Non-Negativity?

Why do we care about having all the entries in the factorized matrices be positive?

Why All the Non-Negativity?

Why do we care about having all the entries in the factorized matrices be positive?

It comes down to the interpretability of the topics. Remember, in PCA/SVD we were allowed negative entries in our matrices.

Why All the Non-Negativity?

Why do we care about having all the entries in the factorized matrices be positive?

It comes down to the interpretability of the topics. Remember, in PCA/SVD we were allowed negative entries in our matrices.

However, when looking at our data, we frequently find that we have all positive entries in X. So, when it comes to interpreting the negative values in the decomposed matrices, how do we think about them?

Comparing PCA/SVD and NMF

Sports	Politics	…	Burritos
-0.8	9.1	...	-3.2
0.6	-0.26	...	15.3
..	...	...	...
...	...	...	...
3.5	4.6	...	-1.02

PCA / SVD

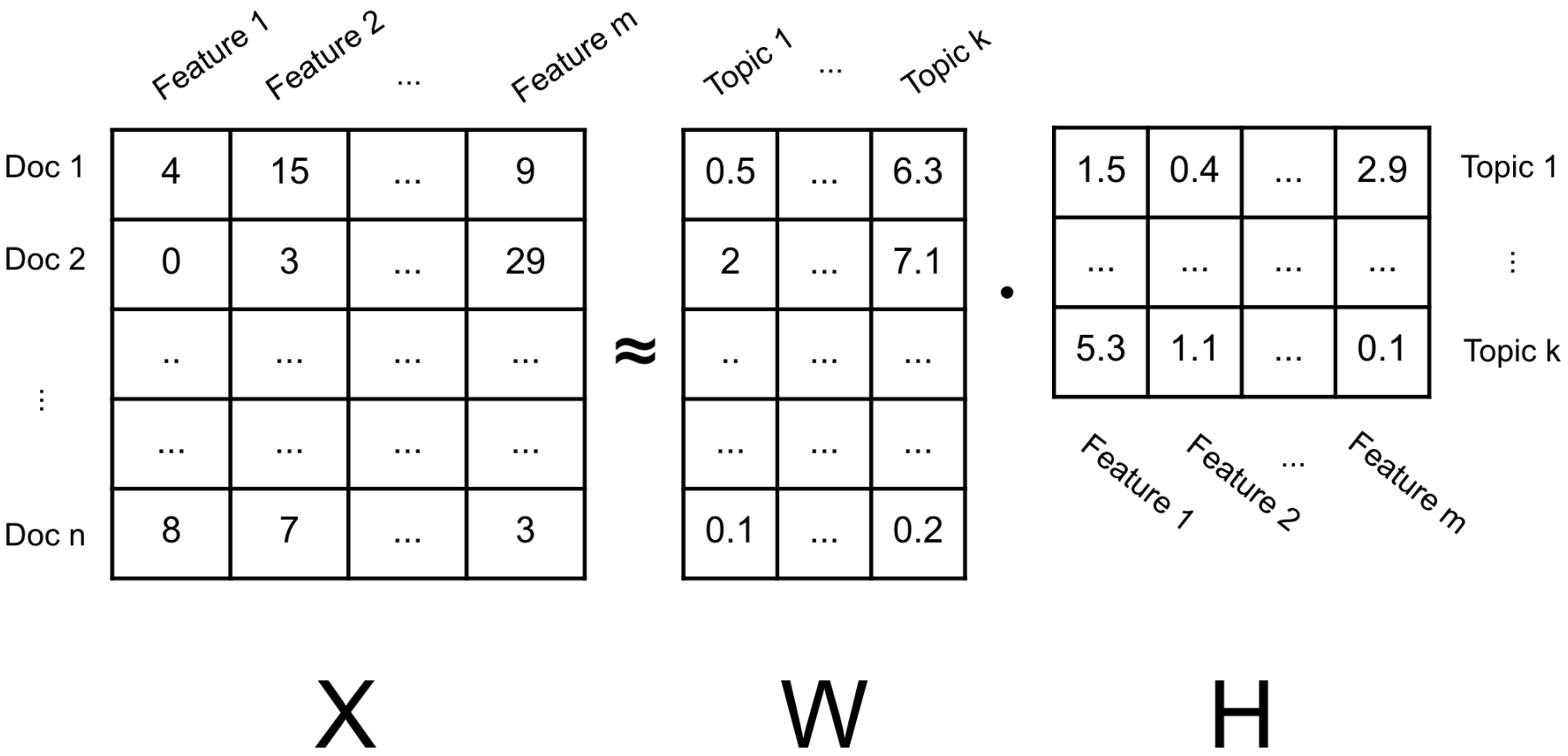
Sports	Politics	…	Burritos	
0.3	5.1	...	1.2	Doc 1
9.76	0.04	...	2.7	Doc 2
..	...	...	...	⋮
...	...	...	...	
0.06	0.3	...	0.001	Doc n

NMF

Have only additive components of a topic makes more sense!

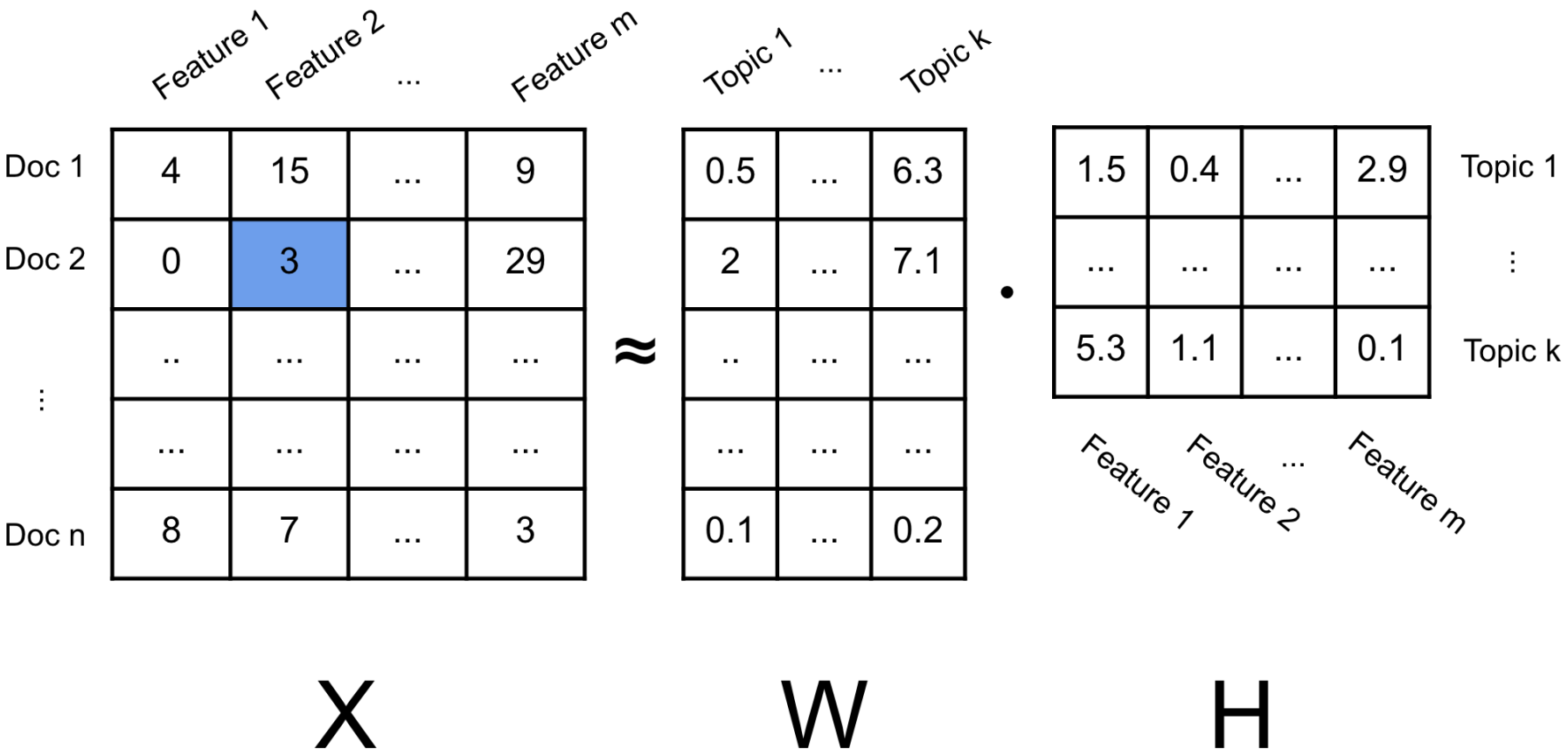
Reconstructing X

X is approximated by the inner product of W and H.



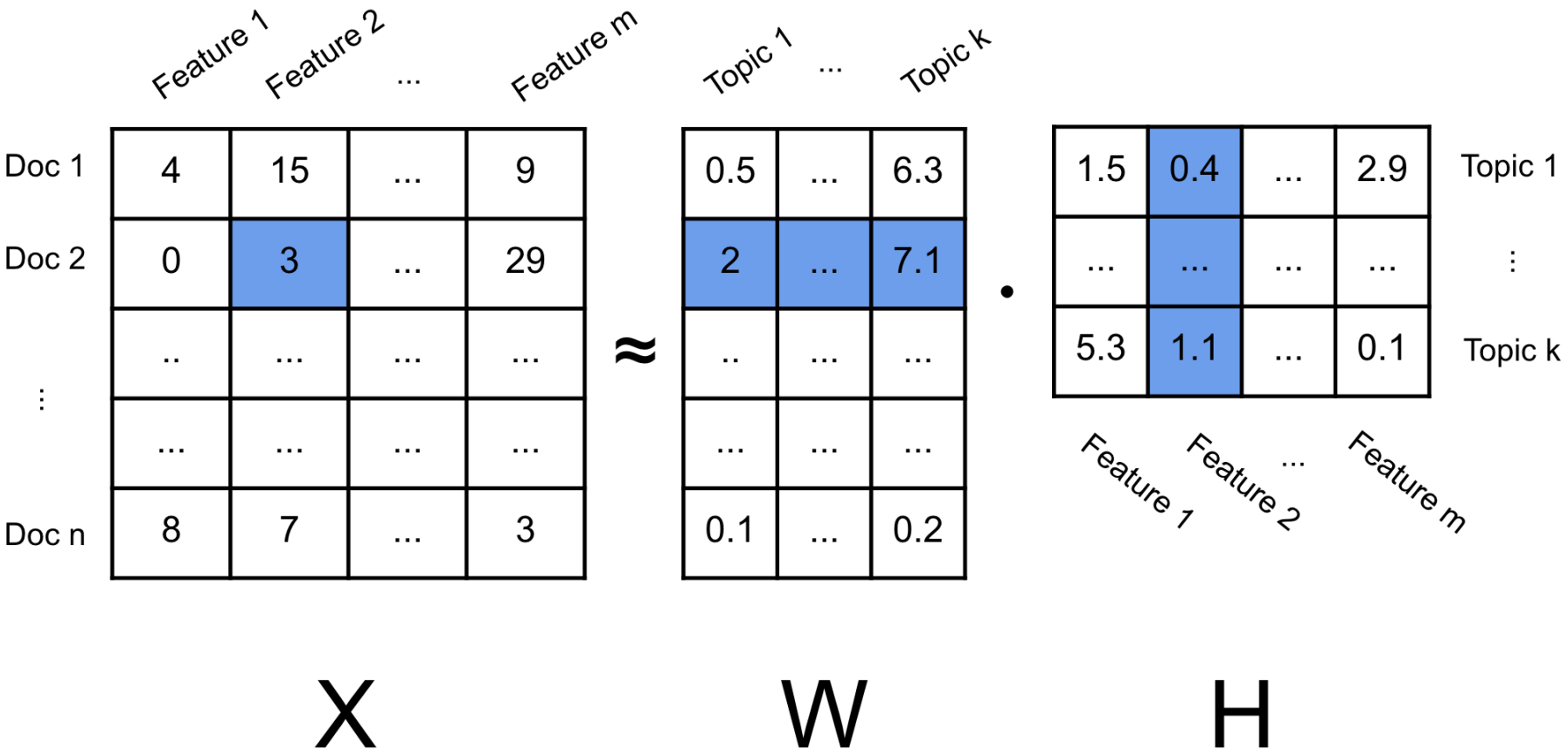
Reconstructing X

How do we reconstruct only one cell of X?



Reconstructing X

Inner product of W and H's correct column and row.



How Could We Find Such Matrices?

So, we're trying to find matrices, W and H such that:

$$X \approx W \cdot H \qquad w_{i,j} \,\&\, h_{i,j} \geq 0$$

With PCA and SVD we have a closed form solution for finding the those factorizations. Unfortunately, we don't have such a solution for NMF.

How Could We Find Such Matrices?

So, we're trying to find matrices, W and H such that:

$$X \approx W \cdot H \qquad w_{i,j} \,\&\, h_{i,j} \geq 0$$

With PCA and SVD we have a closed form solution for finding the those factorizations. Unfortunately, we don't have such a solution for NMF.

So how do we find W and H?

Overview

Topic Modeling

- Motivation
- Thinking About Topics
- Assumptions

Non-Negative Matrix Factorization

- The Math
- Latent Features
- Interpretation
- Algorithms

Finding W and H

We call this type of optimization problem biconvex, it's convex in either W **or** H but not both, and there's a straightforward way we can brute force an approximate solution in this case.

Finding W and H

We call this type of optimization problem biconvex, it's convex in either W **or** H but not both, and there's a straightforward way we can brute force an approximate solution in this case.

We can notice, that, while there is no closed form solution for W **and** H, if we hold one of these matrices constant there **is** a closed form optimum for the other.

This leads us to the **Alternating Least Squares** algorithm.

Alternating Least Squares (ALS)

We will take advantage of the biconvexivity by alternating which matrix, W or H, that we treat as stationary, solving for the other's optimal values, and then clipping all the negative values in that solution to 0.

Alternating Least Squares (ALS)

We will take advantage of the biconvexivity by alternating which matrix, W or H, that we treat as stationary, solving for the other's optimal values, and then clipping all the negative values in that solution to 0.

Pseudo-code:

1. Initialize W to small, positive, random values.
2. For max number of iterations:
    1. Find the least squares solution to $X = W \cdot H$ w.r.t. H.
    2. Clip negative values in H to 0: $H < 0 = 0$.
    3. Find the least squares solution to $X = W \cdot H$ w.r.t. W.
    4. Clip negative values in H to 0: $W < 0 = 0$.

ALS Pros and Cons

Pros

- Fast
- Works well in practice

Cons

- Non-negativity enforced in an ad hoc way
- Not guaranteed to find a local minimum much less global
- No convergence theory

Cost Function

Another way that we could solve this optimization problem is with gradient descent. What do we need in order to descend a gradient, though?

Cost Function

Another way that we could solve this optimization problem is with gradient descent. What do we need in order to descend a gradient, though?

A cost function! What should our cost function for this optimization problem be?

Cost Function

Another way that we could solve this optimization problem is with gradient descent. What do we need in order to descend a gradient, though?

A cost function! What should our cost function for this optimization problem be?

First we're going to define how much we're missing in our approximation of X, the matrix $X - WH$ as the reconstruction error.

Cost Function

Another way that we could solve this optimization problem is with gradient descent. What do we need in order to descend a gradient, though?

A cost function! What should our cost function for this optimization problem be?

First we're going to define how much we're missing in our approximation of X, the matrix $X - WH$ as the reconstruction error.

Then, we're going to use the generalization of euclidean distance on matrices, also known as the Frobenius norm on the reconstruction error to quantify how well we are approximating X.

Gradient Descent

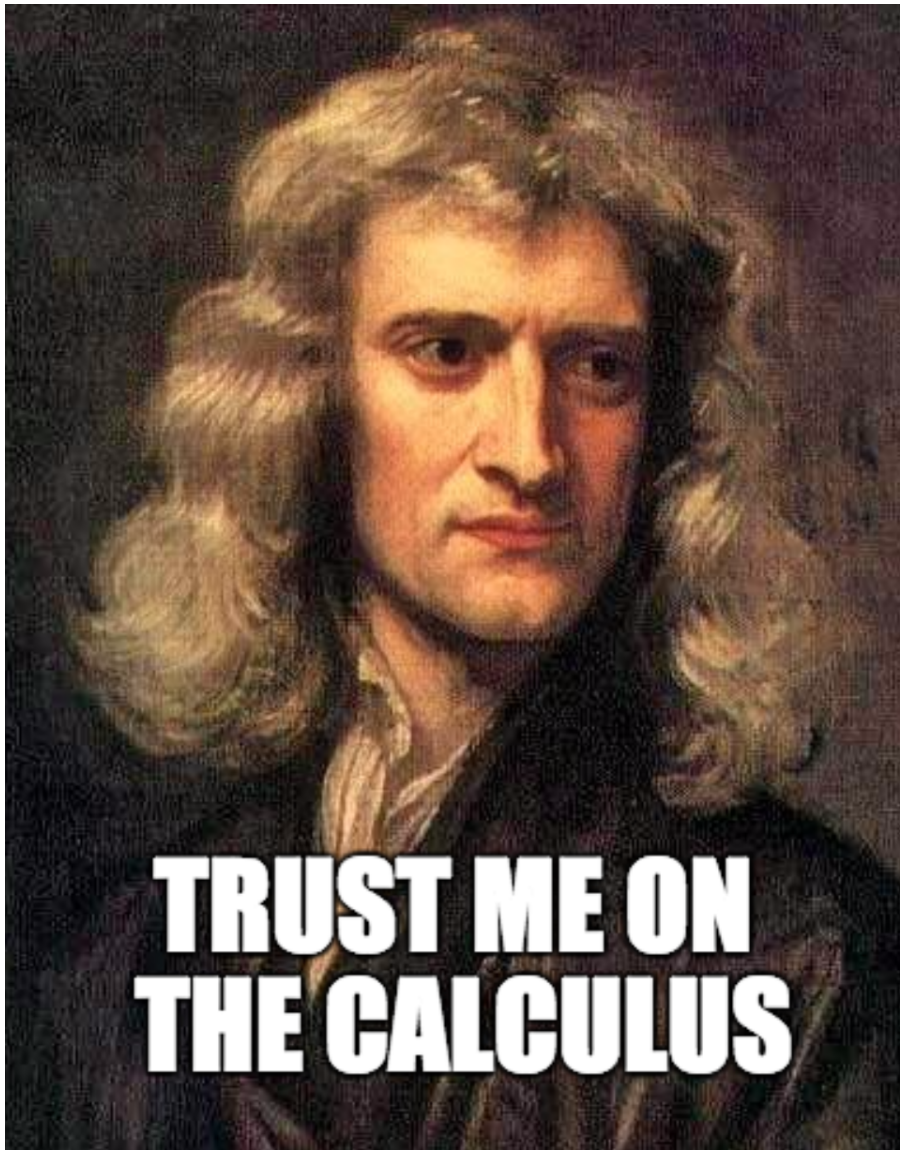
Let the Frobenius norm of the reconstruction error be the quantity that we are trying to minimize:

$$\min_{W,H} \|X - WH\|_F^2$$

From this, with a little be of matrix calculus we can determine that the update rules for a single entry in W and H are:

$$H_{a,\mu} \leftarrow H_{a,\mu} + \nu_{a,\mu}\left[(W^T X)_{a,\mu} - (W^T WH)_{a,\mu}\right]$$
$$W_{i,a} \leftarrow W_{i,a} + \nu_{i,a}\left[(XH^T)_{i,a} - (WHH^T)_{i,a}\right]$$

Are You Sure About that Calculus?



Multiplicative Update Rules

If we are clever about the value that we choose as our step size, ν we can reduce those gradient descent updates to what are known as the multiplicative update rules. Let:

$$\nu_{a,\mu} = \frac{H_{a,\mu}}{(W^T WH)_{a,\mu}} \qquad \nu_{i,a} = \frac{W_{i,a}}{(WHH^T)_{i,a}}$$

Then we can rewrite the gradient descent updates as:

$$H_{a,\mu} \leftarrow H_{a,\mu} \frac{(W^T X)_{a,\mu}}{(W^T WH)_{a,\mu}} \qquad W_{i,a} \leftarrow W_{i,a} \frac{(XH^T)_{i,a}}{(WHH^T)_{i,a}}$$

These updates will be iteratively performed from W and H initialized with random, small, positive values.

Lack of Unique Solution

It's important to know that NMF, as a topic modeling algorithm will almost never yield a unique solution due to the non-negativity constraint we're putting on the entries in W and H.

The ramifications of this are that the latent topics that you discover each time you run NMF can change quite a lot. This is somewhat unsatisfying, however, it's the price we pay for interpretable topics.

Transforming New Data

Once we have a factorization set up we might want to be able to project a new document into our latent feature space. How would we do this?

Transforming New Data

Once we have a factorization set up we might want to be able to project a new document into our latent feature space. How would we do this?

It's actually not particularly difficult. We can use the same OLS technique we learned in linear regression. Let's see how this works.

OLS for Transforming New Data

What we have is a new row in our X matrix, X_{new} and we're trying to find a representation in the space spanned by the columns of W, a.k.a. W_{new}. So what we're actually trying to solve is the equation:

$$\underset{1 \times m}{X_{new}} = \underset{1 \times k}{W_{new}} \cdot \underset{k \times m}{H_{same}}$$

for W_{new}.

Looks a lot like:

$$\underset{n \times 1}{y} = \underset{n \times m}{X} \; \underset{m \times 1}{\beta}$$

But it's worth noting that this functionality is available to you built into existing NMF algorithms, e.g. sklearn's (with the *transform*() method).

Side Notes

NMF is considered a soft clustering, so called because each of the latent features can be viewed as a cluster, and because each observation can be partially in more than one "cluster".

There are a number of penalties that have been devised to make the factorizations more friendly to interpretation.

Simple Ridge / Lasso regularization.

Complex Try to enforce sparsity in W and H.