MUSIC RECOMMENDATION AT SPOTIFY

- Manual Curation
- Manually Tag Attributes
- Audio Content, Metadata, Text Analysis
- Collaborative Filtering

https://www.slideshare.net/MrChrisJohnson/algorithmic-music-recommendations-at-spotify/8-8How_can_we_find_goodrecommendations

The "Music Genome Project" of Pandora

- Up to 450 "genes"
- Each gene corresponds to a characteristic of the music
- Human-powered: Each song is analyzed by a musicologist

https://en.wikipedia.org/wiki/Music_Genome_Project

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Alice = 95% Romance fan + 5% Action fan + …
Bob = 90% Romance fan + 10% Action fan + …
Carol = 5% Romance fan + 95% Action fan + …
…

Problem motivation

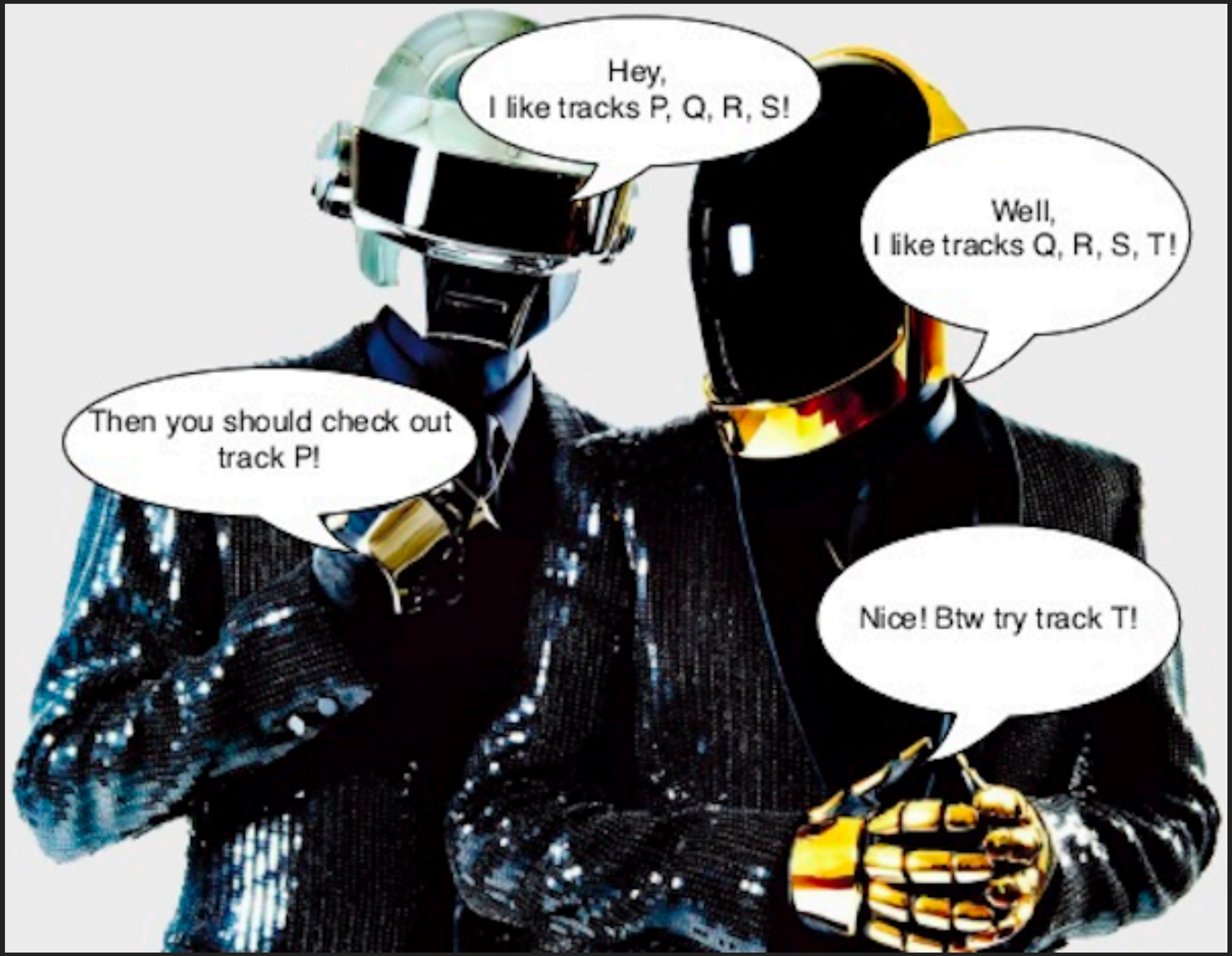
Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Optimization algorithm:

$$\min_{\theta^{(1)},\ldots,\theta^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$$

COLLABORATIVE FILTERING



https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

COLLABORATIVE FILTERING: NEIGHBORHOOD METHOD

AGGREGATING THE NEIGHBORS' RATINGS

$$
\begin{array}{cl}
\begin{pmatrix}
? & 2 & ? & 2 & 5 & ? & 4 \\
1 & 1 & ? & 3 & 5 & ? & ? \\
 & & & \vdots & & & \\
4 & 5 & ? & ? & 1 & 4 & 2
\end{pmatrix}
&
\begin{array}{l}
\text{Alice} \\
\text{Bob} \\
\vdots \\
\text{Zoe}
\end{array}
\end{array}
$$

Alice is close to Bob. Her rating for Titanic is probably close to Bob's (1). So $? \leftarrow 1$

Slides by Nicholas Hug

https://www.youtube.com/watch?v=z0dx-YckFko

How to define similarity?

- Number of common related items
- Average absolute difference between ratings
- Cosine angle between vectors u and v
- Pearson correlation coefficient
- And: millions of others!

https://www.youtube.com/watch?v=z0dx-YckFko

CONTENT-BASED

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Alice = 95% Romance fan + 5% Action fan + …
Bob = 90% Romance fan + 10% Action fan + …
Carol = 5% Romance fan + 95% Action fan + …
…

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0	?	?
Cute puppies of love	?	4	0	?	?	?
Nonstop car chases	0	0	5	4	?	?
Swords vs. karate	0	0	5	?	?	?

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Alice = ? % Romance fan + ? % Action fan + …

Bob = ? % Romance fan + ? % Action fan + …

Carol = ? % Romance fan + ? % Action fan + …

…

COLLABORATIVE FILTERING

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (r ? e)	x_2 (?)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0	?	?
Cute puppies of love	?	4	0	?	?	?
Nonstop car chases	0	0	5	4	?	?
Swords vs. karate	0	0	5	?	?	?

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Alice = ? % ? fan + ? % ? fan + …

Bob = ? % ? fan + ? % ? fan + …

Carol = ? % ? fan + ? % ? fan + …

…

MATRIX FACTORIZATION

- A lot of hype during the Netflix Prize (2006-2009: *improve our system, get rich*)
- Model the ratings in an insightful way
- Takes its root in dimensional reduction and SVD

Slides by Nicholas Hug

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

BEFORE SVD: PCA

- Here are 400 greyscale images (64 x 64):

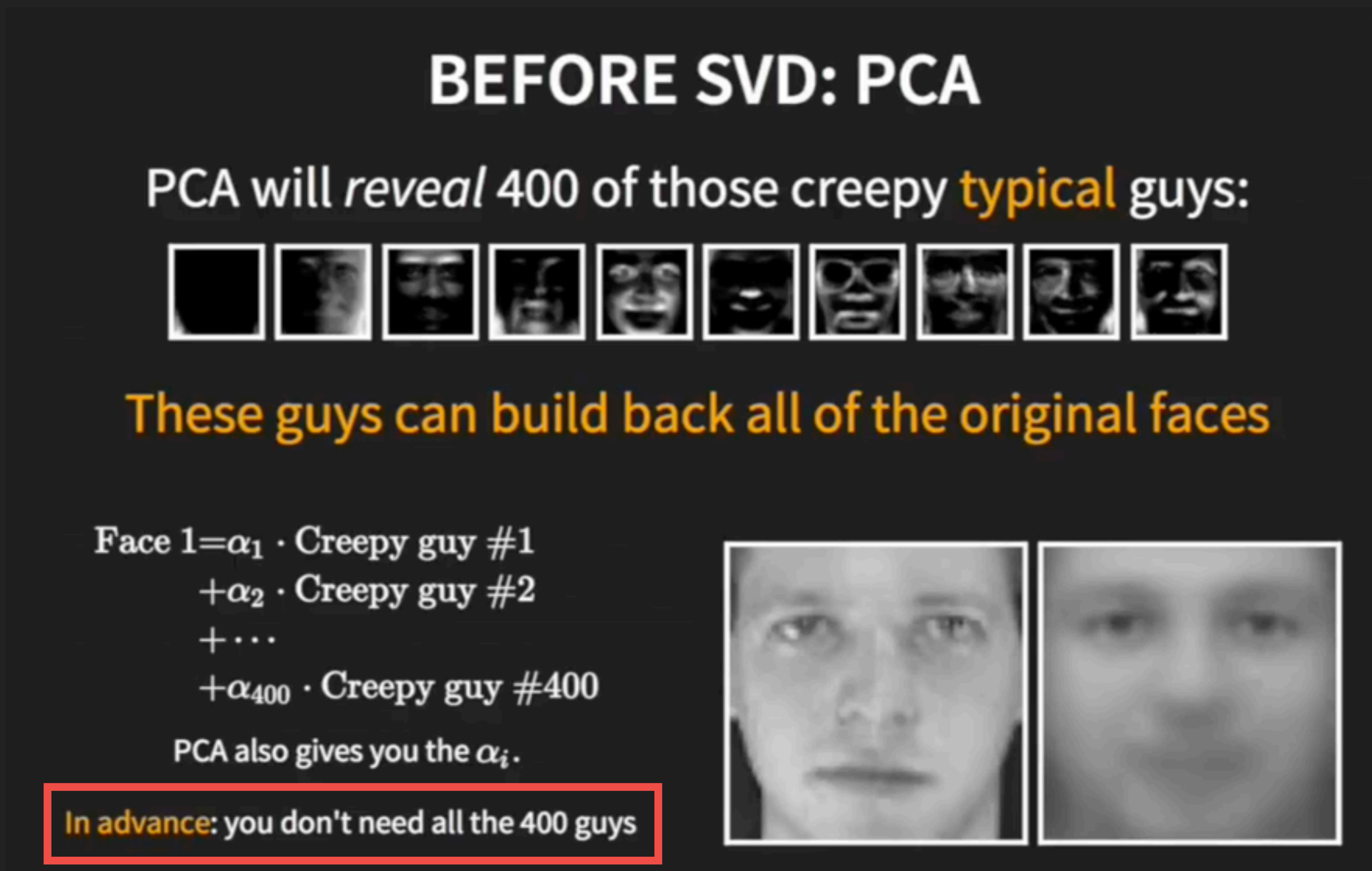
- Put them in a 400 x 4096 matrix X:

$$X = \begin{pmatrix} \text{---} & \text{Face 1} & \text{---} \\ \text{---} & \text{Face 2} & \text{---} \\ & \vdots & \\ \text{---} & \text{Face 400} & \text{---} \end{pmatrix}$$

Slides by Nicholas Hug

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

COLLABORATIVE FILTERING: MATRIX FACTORIZATION



Slides by Nicholas Hug

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

COLLABORATIVE FILTERING: MATRIX FACTORIZATION

PCA ON A RATING MATRIX? SURE!

Assume all ratings are known

$$X = \begin{pmatrix} — & \text{Face 1} & — \\ — & \text{Face 2} & — \\ & \vdots & \\ — & \text{Face 400} & — \end{pmatrix} \qquad R = \begin{pmatrix} — & \text{Alice} & — \\ — & \text{Bob} & — \\ & \vdots & \\ — & \text{Zoe} & — \end{pmatrix}$$

Exact same thing! We just have ratings instead of pixels.

PCA will reveal typical users.

$$\text{Alice} = 10\%\ \text{Action fan} + 10\%\ \text{Comedy fan} + 50\%\ \text{Romance fan} + \cdots$$
$$\text{Bob} = 50\%\ \text{Action fan} + 30\%\ \text{Comedy fan} + 10\%\ \text{Romance fan} + \cdots$$
$$\text{Zoe} = \cdots$$

Slides by Nicholas Hug

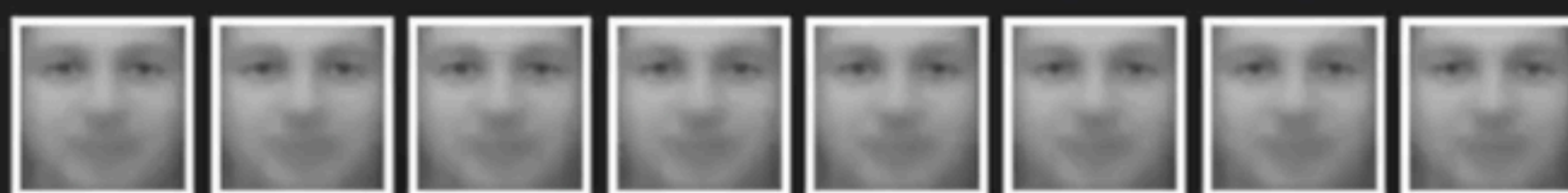
https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

PCA ON A RATING MATRIX? SURE!

Assume all ratings are known. Transpose the matrix

$$X = \begin{pmatrix} — & \text{Face 1} & — \\ — & \text{Face 2} & — \\ & \vdots & \\ — & \text{Face 400} & — \end{pmatrix} \qquad R^T = \begin{pmatrix} — & \text{Titanic} & — \\ — & \text{Toy Story} & — \\ & \vdots & \\ — & \text{Fargo} & — \end{pmatrix}$$

Exact same thing! PCA will reveal typical movies.



$$\text{Titanic} = 20\%\ \text{Action} + 0\%\ \text{Comedy} + 70\%\ \text{Romance} + \cdots$$
$$\text{Toy Story} = 30\%\ \text{Action} + 60\%\ \text{Comedy} + 0\%\ \text{Romance} + \cdots$$

Note: in practice, the factors semantic is not clearly defined.

Slides by Nicholas Hug

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

COLLABORATIVE FILTERING: MATRIX FACTORIZATION

SVD IS PCA²

- PCA on R gives you the typical users U
- PCA on R^T gives you the typical movies M
- SVD gives you both in one shot!

$$R = M\Sigma U^T$$

Σ is diagonal, it's *just* a scaler.

$$\boxed{R = MU^T}$$

This is our matrix factorization!

Slides by Nicholas Hug

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (r ? e)	x_2 (?)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0	?	?
Cute puppies of love	?	4	0	?	?	?
Nonstop car chases	0	0	5	4	?	?
Swords vs. karate	0	0	5	?	?	?

https://www.youtube.com/watch?v=9AP-DgFBNP4&t=22s

Alice = ? % ? fan + ? % ? fan + …

Bob = ? % ? fan + ? % ? fan + …

Carol = ? % ? fan + ? % ? fan + …

…

Implicit Matrix Factorization

- Aggregate all (**user**, **track**) streams into a large matrix
- **Goal**: Approximate binary preference matrix by inner product of 2 smaller matrices by minimizing the weighted **RMSE** (root mean squared error) using a function of plays, context, and recency as weight

$$\text{Users}\ \underbrace{\begin{pmatrix} 1\,0\,0\,0\,1\,0\,0\,1 \\ 0\,0\,1\,0\,0\,1\,0\,0 \\ 1\,0\,1\,0\,0\,0\,1\,1 \\ 0\,1\,0\,0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0\,0 \\ 1\,0\,0\,0\,1\,0\,0\,1 \end{pmatrix}}_{\text{Songs}} \approx \underbrace{\begin{pmatrix} \\ \mathrm{X} \\ \\ \end{pmatrix}}_{f} \left.\begin{pmatrix} \quad \mathrm{Y} \quad \end{pmatrix}\right\} f$$

$$\min_{x,y} \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i - \beta_u - \beta_i)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

- p_{ui} = 1 if user u streamed track i else 0
- $c_{ui} = 1 + \alpha r_{ui}$
- x_u = user $u's$ latent factor vector
- y_i = item $i's$ latent factor vector
- β_u = bias for user u
- β_i = bias for item i
- λ = regularization parameter

[1] Hu Y. & Koren Y. & Volinsky C. (2008) Collaborative Filtering for Implicit Feedback Datasets *8th IEEE Inter*

Slides by Chris Johnson

COLLABORATIVE FILTERING

- Benefits from large user bases

The more users (ratings) the better!

- Flexible across different domains

Seemingly unrelated items by features

- Produces more serendipitous recommendations

Lovely surprises

- Capture more nuance around items

Implicitly capture features that cannot be easily quantified

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

- Complexity and expense

" Millions of items, and millions of users… "

- Data sparsity

" You have listened to only 0.0001% of what we have here "

- The "cold-start" problem

" You are too new for us to know you well :("

https://hackernoon.com/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe

https://www.upwork.com/hiring/data/how-collaborative-filtering-works/

Raw audio: Convolutional Neural Networks

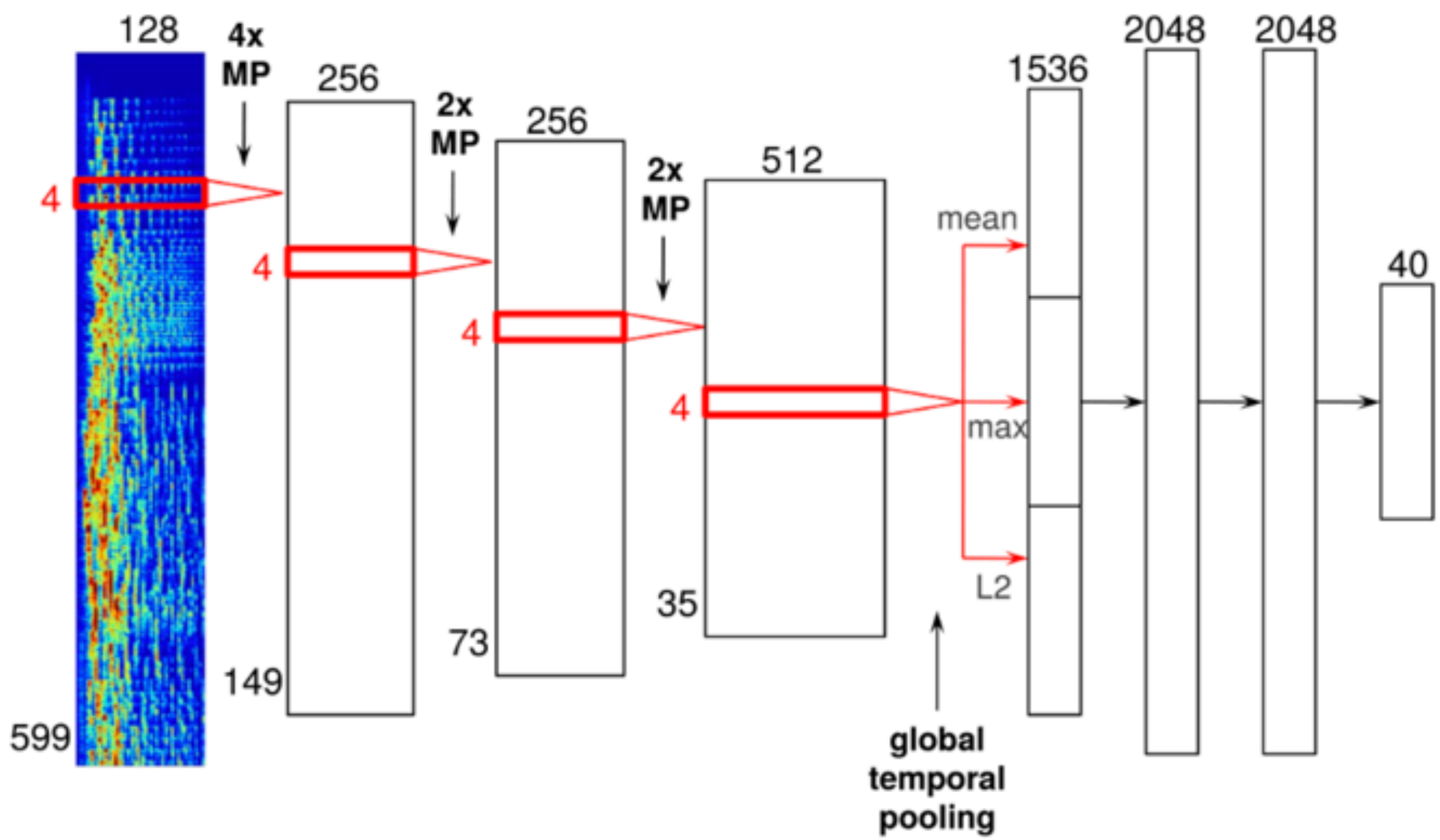


Image credit: Sander Dieleman

Slides by Chris Johnson

https://hackernoon.com/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe

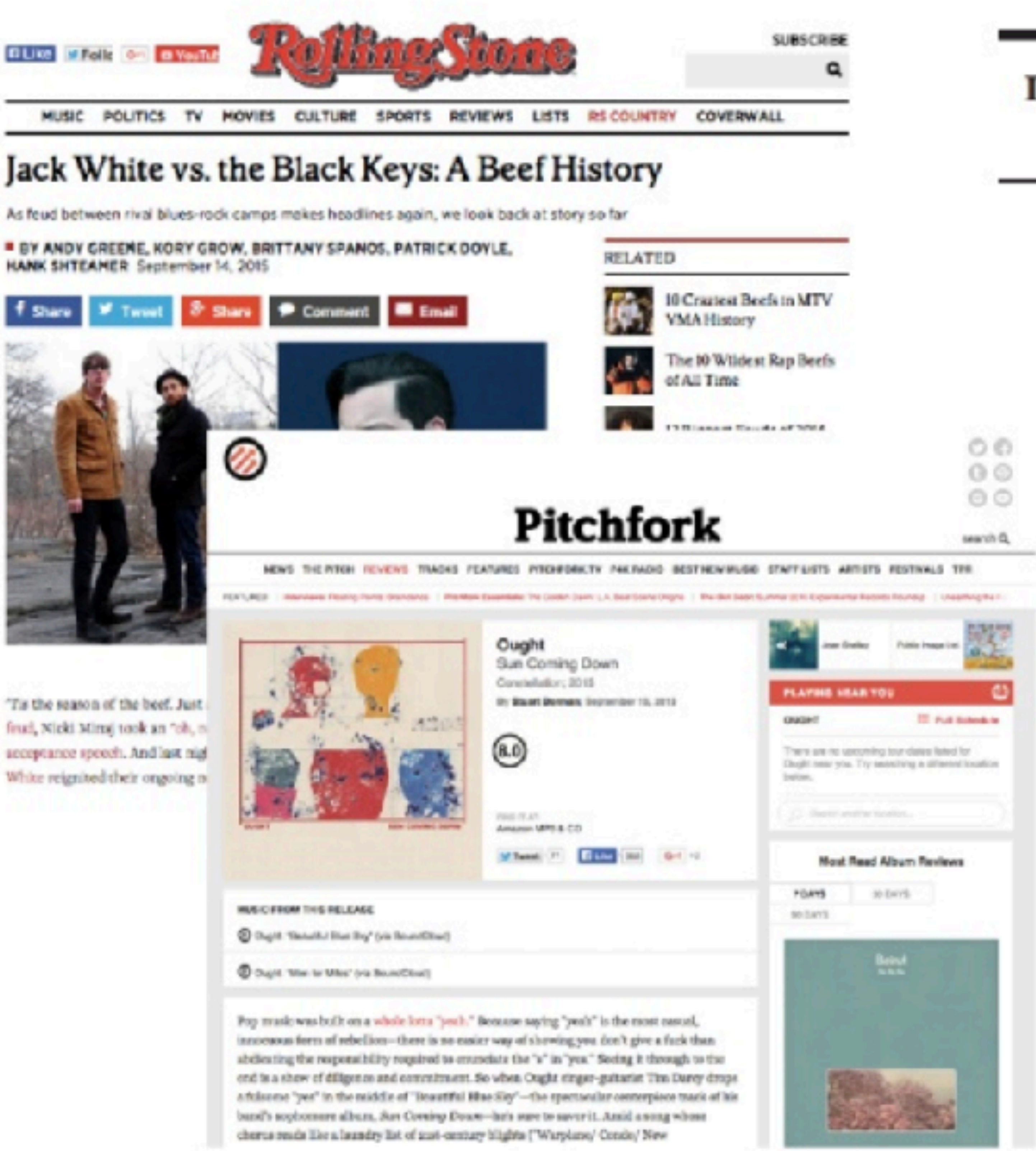
Metadata & internet sources: Natural Language Processing

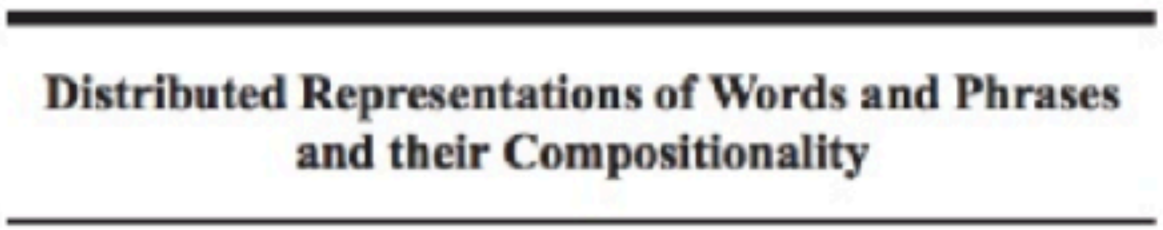
IN NATURAL LANGUAGE PROCESSING, THERE'S A TECHNIQUE CALLED **WORD2VEC**, WHICH TAKES WORDS AND ENCODES THEM INTO A MATHEMATICAL REPRESENTATION—A VECTOR. IN THESE MATHEMATICAL REPRESENTATIONS, VECTORS WITH A SIMILAR SHAPE WOULD EQUATE TO WORDS WITH A SIMILAR MEANING. BASICALLY, IT'S MATHEMATICAL REPRESENTATION OF THE IMPLICIT ASSOCIATIONS AND RELATIONSHIPS BETWEEN WORDS THAT WE KNOW TO BE TRUE IN EVERYDAY SPEECH.

WHAT SPOTIFY DOES IS VERY SIMILAR TO **WORD2VEC**. IT TAKES PLAYLISTS AND TREATS THEM AS A PARAGRAPH OR BIG BLOCK OF TEXT, AND TREATS EACH SONG IN THE PLAYLIST AS AN INDIVIDUAL WORD. THIS RESULTS IN VECTOR REPRESENTATIONS OF SONGS THAT CAN BE USED TO DETERMINE TWO PIECES OF MUSIC THAT ARE SIMILAR. AS SUCH, SPOTIFY IS ABLE TO DETERMINE WHICH SONGS ARE SIMILAR TO EACH OTHER, THUS ENABLING IT TO TACKLE THE COLD START PROBLEM AND RECOMMEND SONGS WITH VERY FEW PLAYS.

https://hackernoon.com/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe

NLP Models on News and Blogs

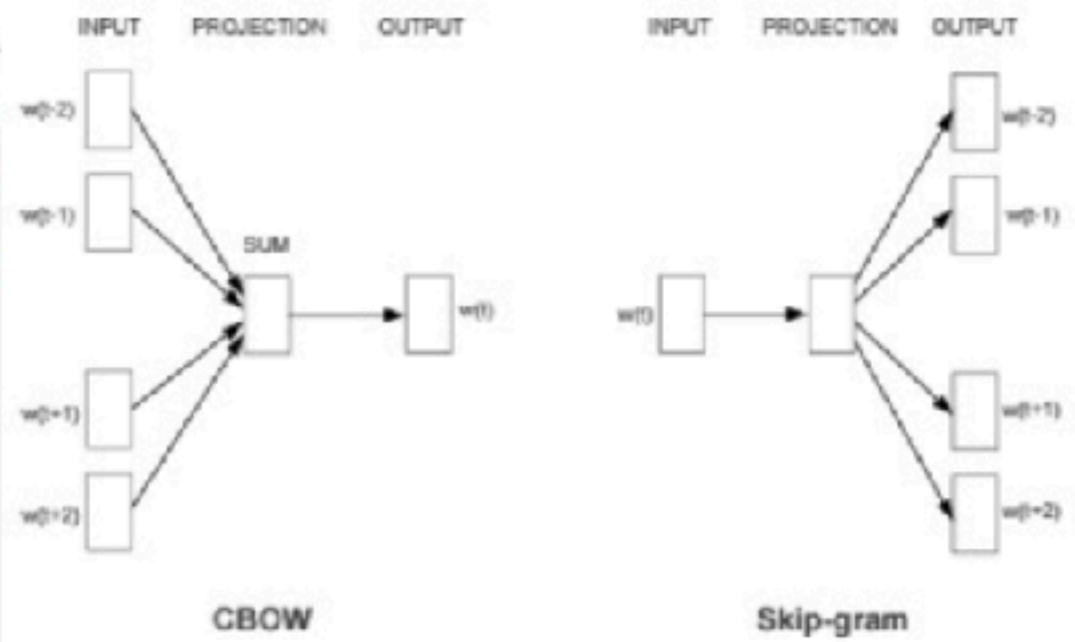




Distributed Representations of Words and Phrases and their Compositionality

Tomas Mikolov	Ilya Sutskever	Kai Chen
Google Inc.	Google Inc.	Google Inc.
Mountain View	Mountain View	Mountain View
mikolov@google.com	ilyasu@google.com	kai@google.com

Greg Corrado	Jeffrey Dean
Google Inc.	Google Inc.
Mountain View	Mountain View
gcorrado@google.com	jeff@google.com

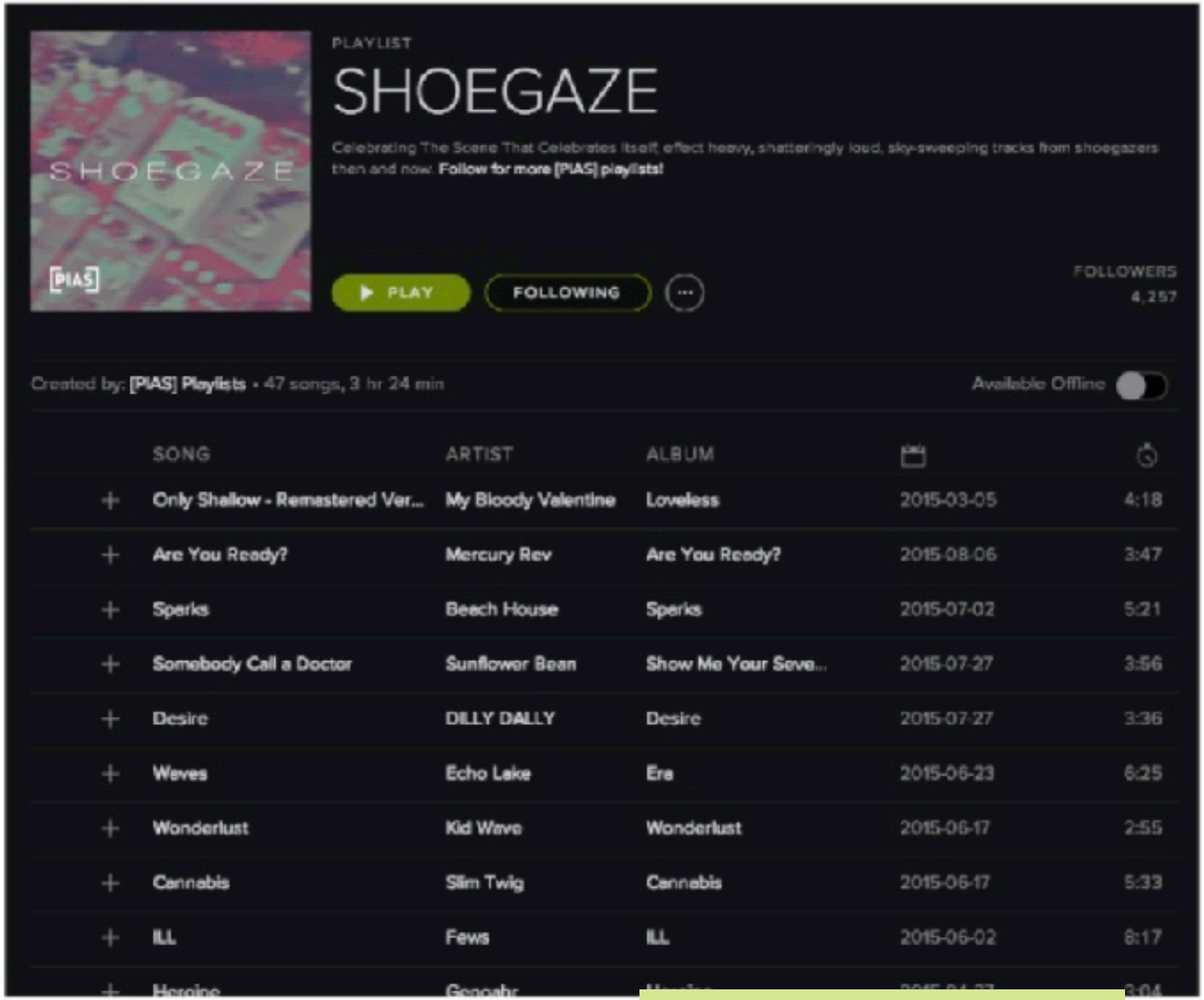


Slides by Chris Johnson

NLP Models work great on Playlists!

Playlist itself is a document

Songs in playlist are words



Slides by Chris Johnson

Q&A

- I'm new to Spotify! What do I do?
- My friend at the party played Maroon5 using my account. Does that ruin my awesome indie folk playlist?
- How do I find more music I like?

REFERENCES

Slides by Chris Johnson (worked at Spotify)

https://www.slideshare.net/MrChrisJohnson/from-idea-to-execution-spotifys-discover-weekly/34-Playlist_itself_is_adocumentSongs_inplaylist

https://www.slideshare.net/MrChrisJohnson/algorithmic-music-recommendations-at-spotify

Slides by Erik Bernhardsso (worked at Spotify)

https://www.slideshare.net/erikbern/collaborative-filtering-at-spotify-16182818/10-Supervised_collaborative_filtering_is_pretty

Youtube series on recommendation systems by Andrew Ng:
https://www.youtube.com/watch?v=z0dx-YckFko

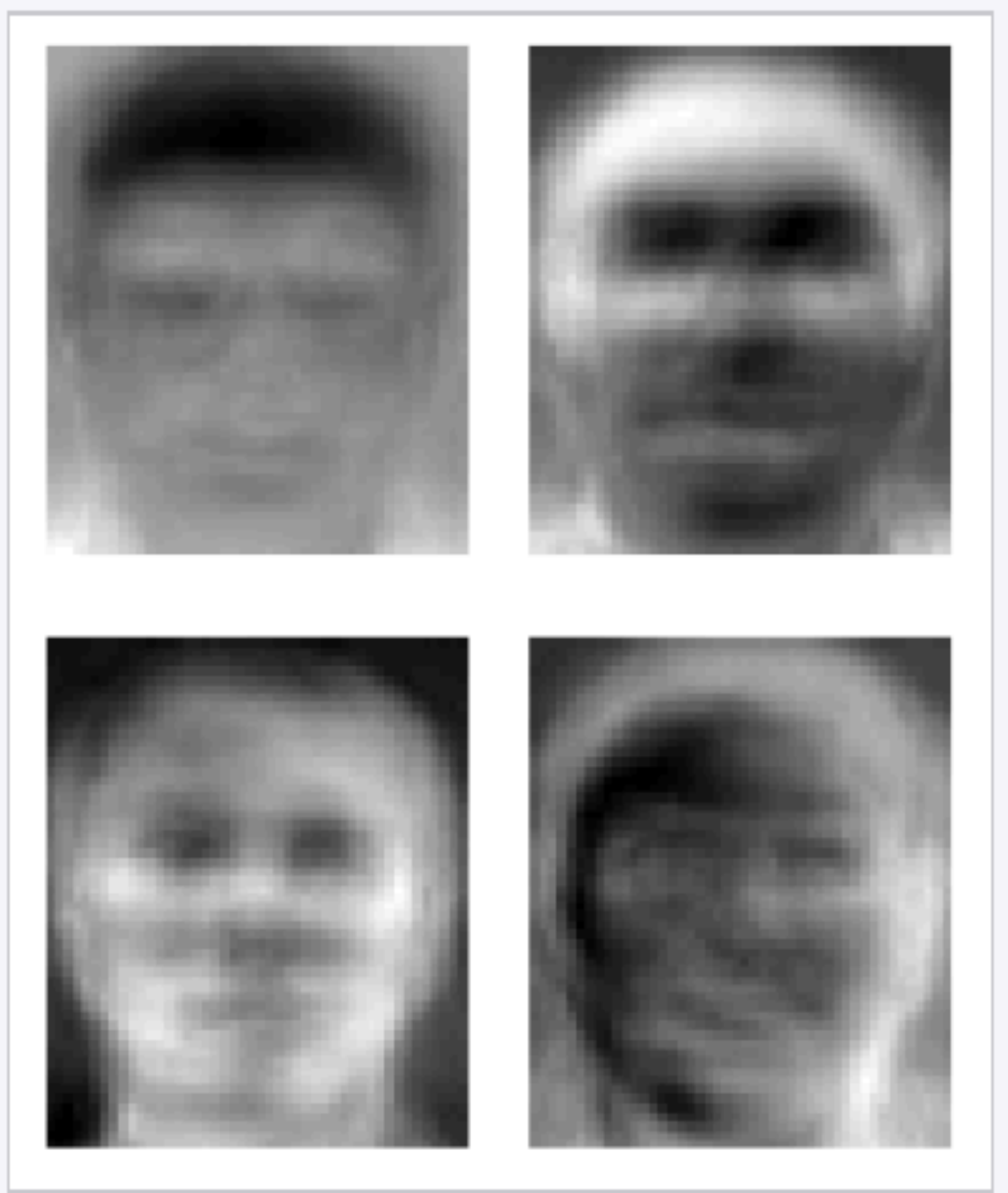
Excellent overview of Spotify's workflow:
https://hackernoon.com/spotifys-discover-weekly-how-machine-learning-finds-your-new-music-19a41ab76efe

Great talk on matrix factorization by Nicholas Hug:
https://www.youtube.com/watch?v=9AP-DgFBNP4&t=219s

REFERENCES

Eigenfaces

https://en.wikipedia.org/wiki/Eigenface



Some eigenfaces from AT&T Laboratories Cambridge