Spotify: How We Kept the Party Rolling

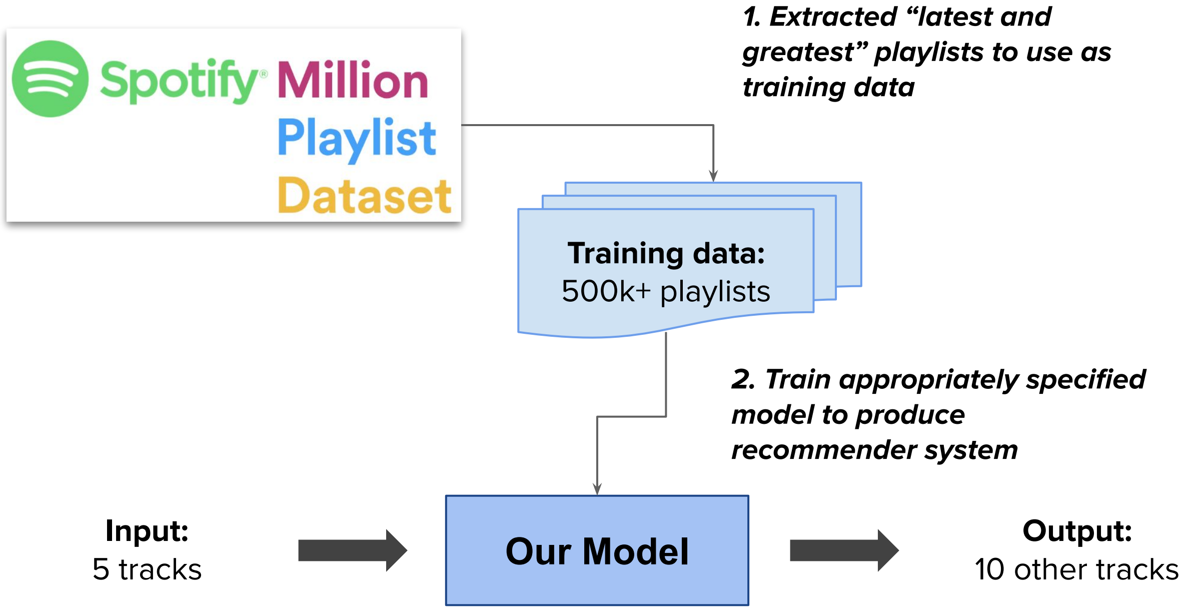
Bronte Baer, Jean-Luc Jackson, Lawis Koh, Christian Montecillo
3 August 2022

Recap: we want to make it easy for users to build their own playlists

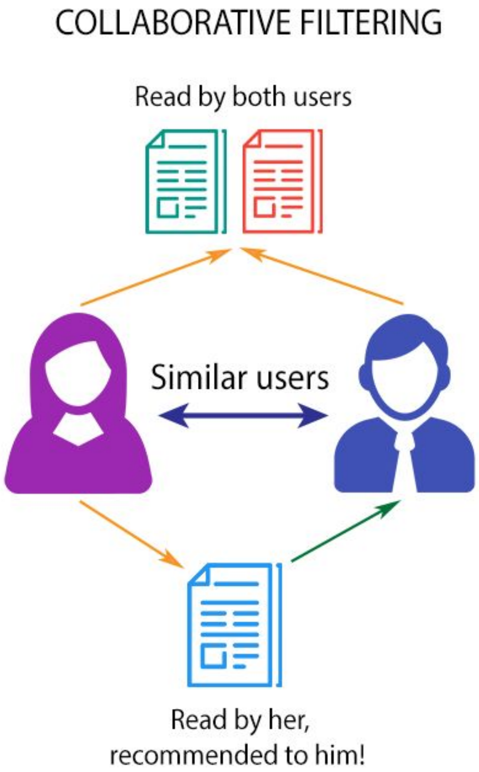
Pain points

- Hard to recall more than a few songs at a time
- Impossible to search through entire universe of songs

Our solution

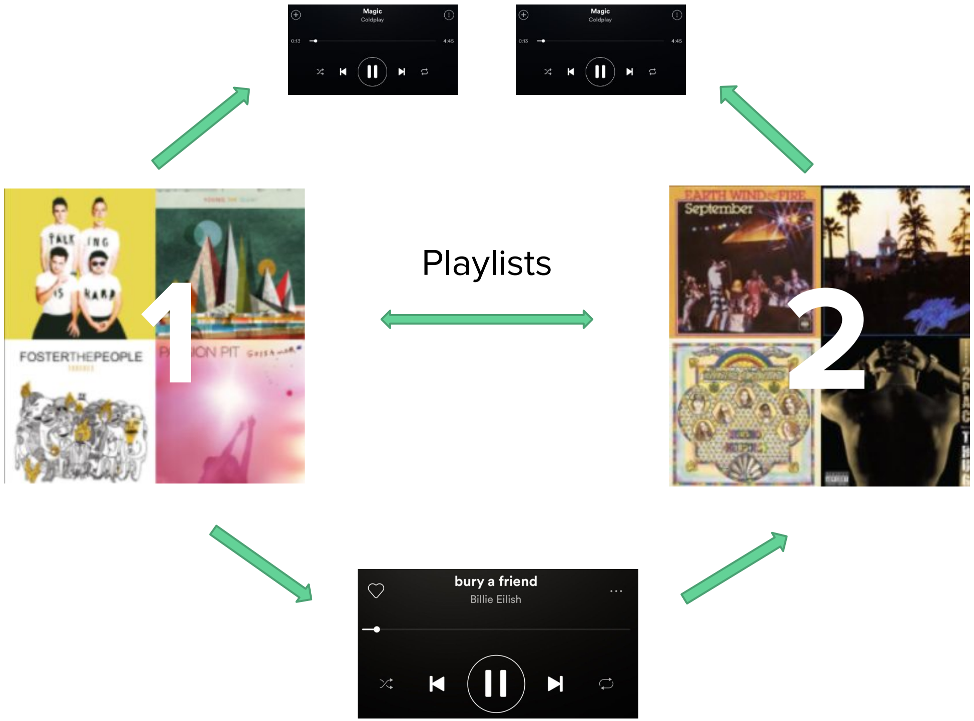


What did we want to do?



What we did...

Track on both playlists



Track on playlist 1, suggested for playlist 2

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

Clarification of key concepts:

Collaborative filtering

Matrix factorization

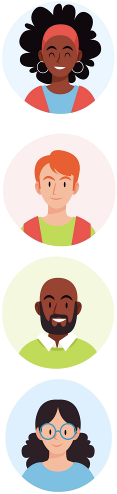
					
	???	2	???	2	0
	4	2	1	???	???
	???	4	2	2	???
	???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

Clarification of key concepts:

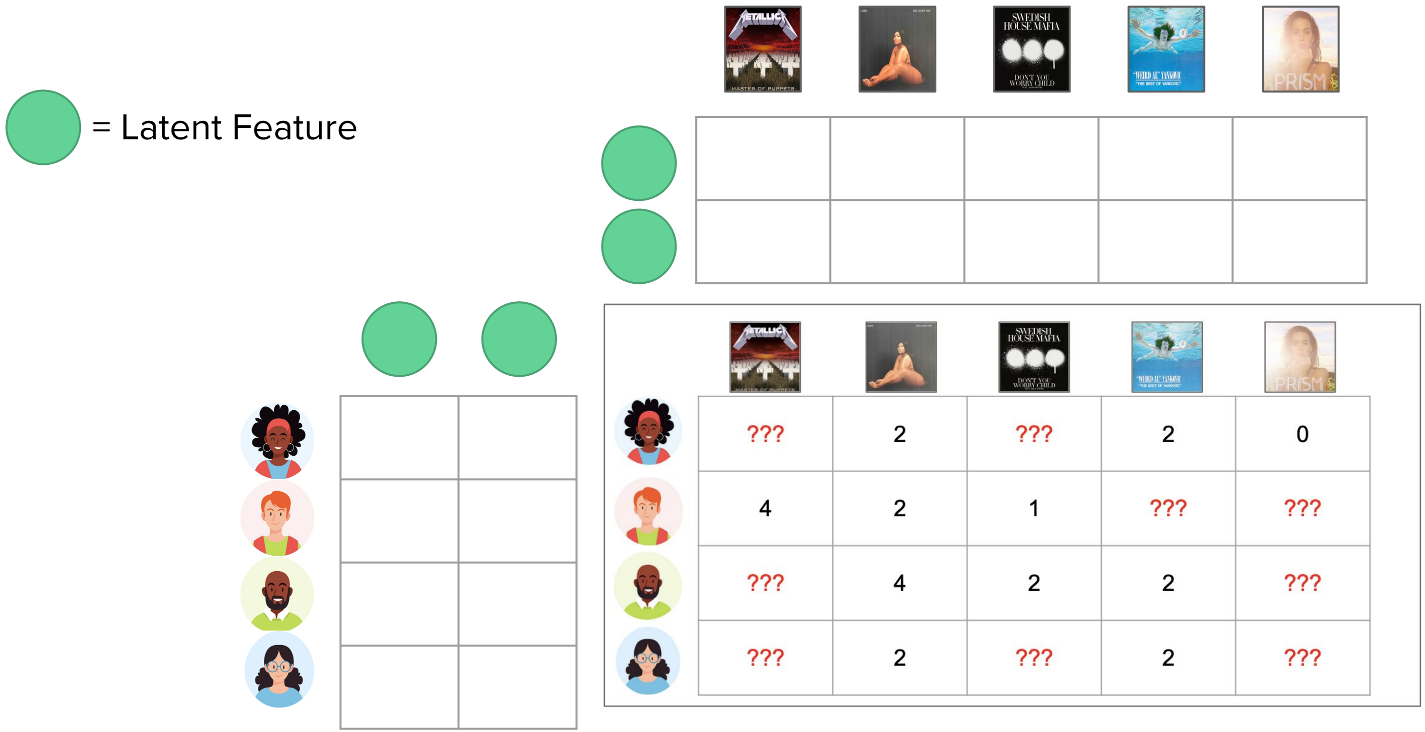
Collaborative filtering

Matrix factorization



???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

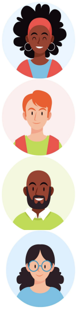


???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

0	2	1	3	1
0	1	1	1	0



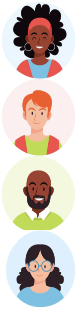
1	2
2	3
0	1
0	2

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

2	0	1	0	1
0	1	2	1	0



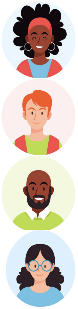
3	1
1	1
0	3
3	0

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

1	0	2	2	3
1	3	2	0	2



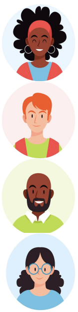
1	0
1	2
2	0
1	0

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

1	3	0	2	1
1	3	1	0	1



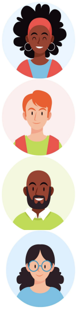
0	2
2	0
2	1
0	0

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

0	2	1	3	1
0	2	2	1	0



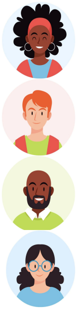
1	2
2	3
0	1
0	2

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

3	2	1	1	2
0	1	2	1	0



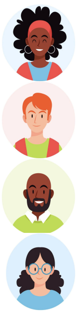
0	2
1	0
2	0
0	2

???	2	???	2	0
4	2	1	???	???
???	4	2	2	???
???	2	???	2	???

We chose to build a neural recommender system using a matrix factorization-based approach to collaborative filtering

= Latent Feature

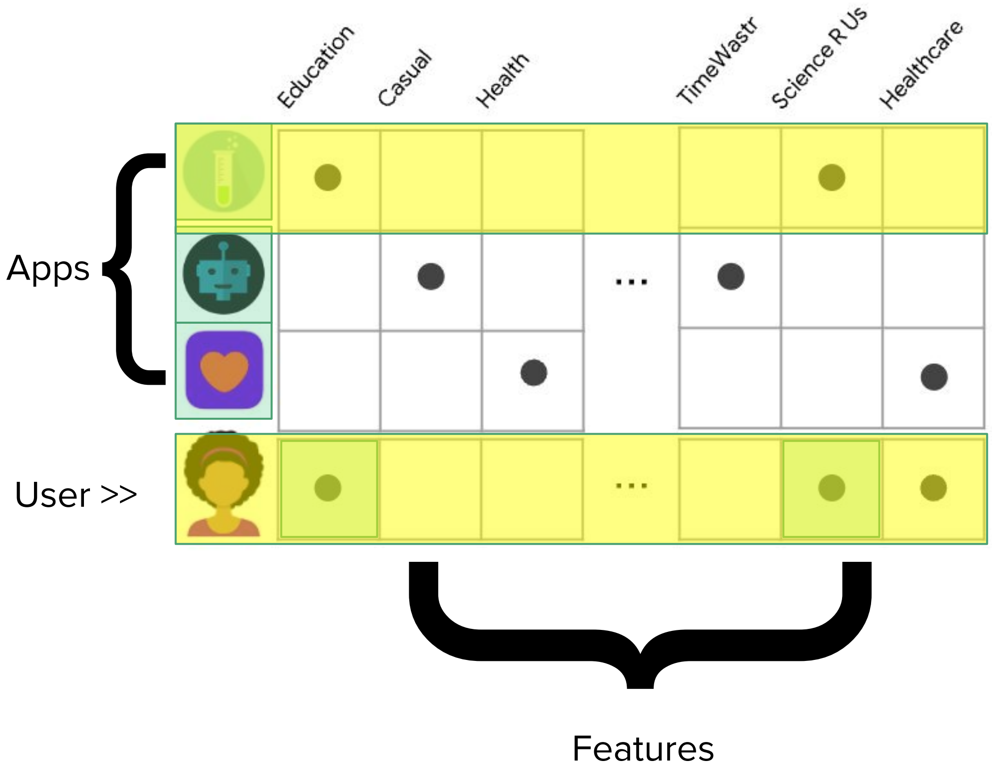
3	2	1	1	2
0	1	2	1	0



0	2
1	0
2	0
0	2

0	2	4	2	0
4	2	1	1	2
8	4	2	2	4
0	2	4	2	0

Content-Based Filtering



$$
\text{Apps}\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{array}{c} \text{User} \\ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \end{array} = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}
$$

The structure of our model is as follows:

```
Model: "binary_recommender_net_7"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
playlist_embeddings (Embeddi multiple                  15000

_________________________________________________________________
playlist_bias (Embedding)    multiple                  5000

_________________________________________________________________
track_embeddings (Embedding) multiple                  1937406

_________________________________________________________________
track_bias (Embedding)       multiple                  645802
=================================================================
Total params: 2,603,208
Trainable params: 2,603,208
Non-trainable params: 0

_________________________________________________________________
```

Matrix of latent factors for playlists and tracks

The structure of our model is as follows:

```
Model: "binary_recommender_net_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
playlist_embeddings (Embeddi multiple                  15000
_________________________________________________________________
playlist_bias (Embedding)    multiple                  5000
_________________________________________________________________
track_embeddings (Embedding) multiple                  1937406
_________________________________________________________________
track_bias (Embedding)       multiple                  645802
=================================================================
Total params: 2,603,208
Trainable params: 2,603,208
Non-trainable params: 0
_________________________________________________________________
```

Matrix of latent factors for playlists and tracks

Adding bias layers to capture effects that happen independently of user-item interactions.

The structure of our model is as follows:

```
Model: "binary_recommender_net_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
playlist_embeddings (Embeddi multiple                  15000
_________________________________________________________________
playlist_bias (Embedding)    multiple                  5000
_________________________________________________________________
track_embeddings (Embedding) multiple                  1937406
_________________________________________________________________
track_bias (Embedding)       multiple                  645802
=================================================================
Total params: 2,603,208
Trainable params: 2,603,208
Non-trainable params: 0
_________________________________________________________________
```

Matrix of latent factors for playlists and tracks

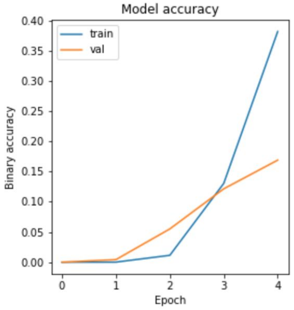
Adding bias layers to capture effects that happen independently of playlist-track interactions.

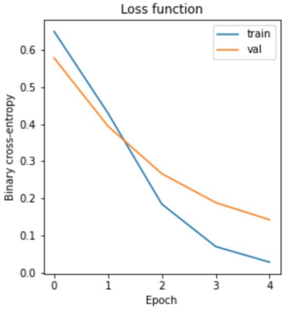
Final model specification:

sigmoid(dot_product(playlist_factors, track_factors) + playlist_bias + track_bias)

Comparison to Baseline Models

Models evaluated:	Binary Accuracy Scores		
	Train	**Validation**	**Test**
Our model: *Matrix factorization-based CF model*	35%	18%	15%
Naive baseline 1: *Recommend tracks by same artists*	—	8%	
Naive baseline 2: *Recommend tracks from same albums*	—	7%	



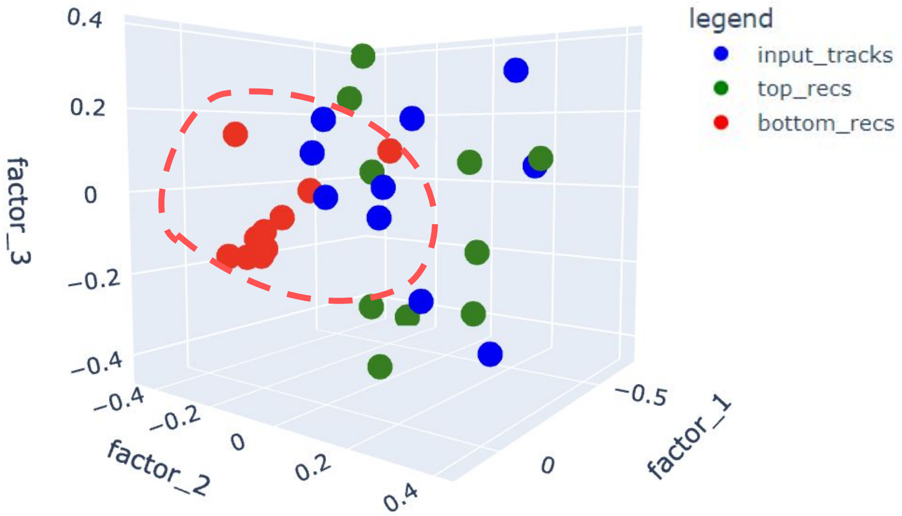


We gain further confidence in our results by investigating the embedding weights and model outputs for a workout playlist (1/2)

Input tracks...

1. X Gon' Give It To Ya (DMX)
2. Bad and Boujee (Migos)
3. Look At Me Now (Chris Brown)
4. Pump Up The Jam (Technotronic)
5. 'Till I Collapse (Eminem)

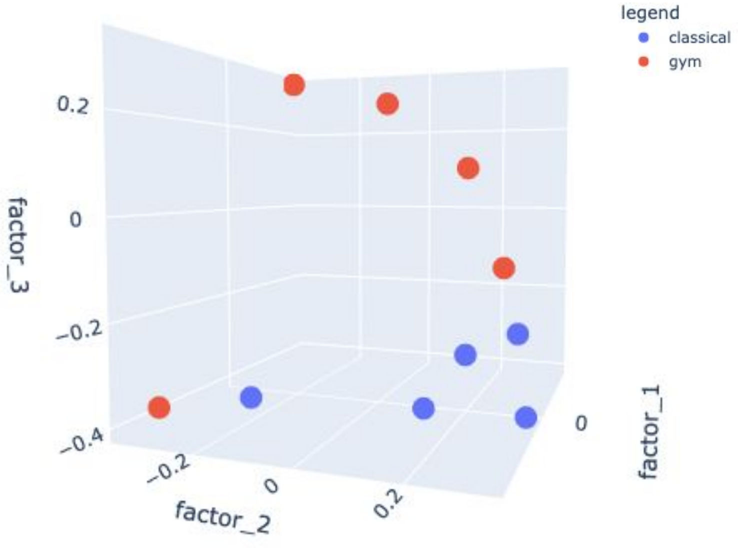
...decomposed into 3d-space



Model-assigned weights cause least appropriate songs to lie within a distinct area in 3d-space. Examples of inappropriate tracks include: Te Extrano (by Xtreme) – a bachata song – which will get you sweaty in different ways

We gain further confidence in our results by investigating the embedding weights and model outputs for a workout playlist (2/2)

Distinctly different playlists also located within distinct areas (kinda)...



...giving us confidence that our recommendations are personalized

```
Top 5 most relevant tracks:
---------------------------------------------------
1. Despacito - Remix (Luis Fonsi from "Despacito Feat. Justin B
ieber")
2. Wanted (Hunter Hayes from "Hunter Hayes (Encore)")
3. Issues (Julia Michaels from "Nervous System")
4. Redbone (Childish Gambino from ""Awaken, My Love!"")
5. Perfect (Ed Sheeran from "÷")
```

```
Top 5 least relevant tracks:
---------------------------------------------------
1. Estamos Quites - Ao Vivo (Zé Neto & Cristiano from "Ao Vivo
Em São José do Rio Preto")
2. Paredes - Ao Vivo (Jorge & Mateus from "Como. Sempre Feito.
Nunca (Ao Vivo)")
3. Mi Gente (A.B. Quintanilla III from "4")
4. Sky and Sand (Paul Kalkbrenner from "Berlin Calling (The Sou
ndtrack by Paul Kalkbrenner)")
5. Los Infieles (Aventura from "K.O.B. Live")
```

We see three broad areas for improvement

Model specification

- Address overfitting by including dropout layers (or just train the model with more data)
- Exploring non-binary definitions of utility by weighting binary labels based on
    - Number of followers
    - Sequence in playlist

Hyperparameter tuning

Some hyperparameters to consider:

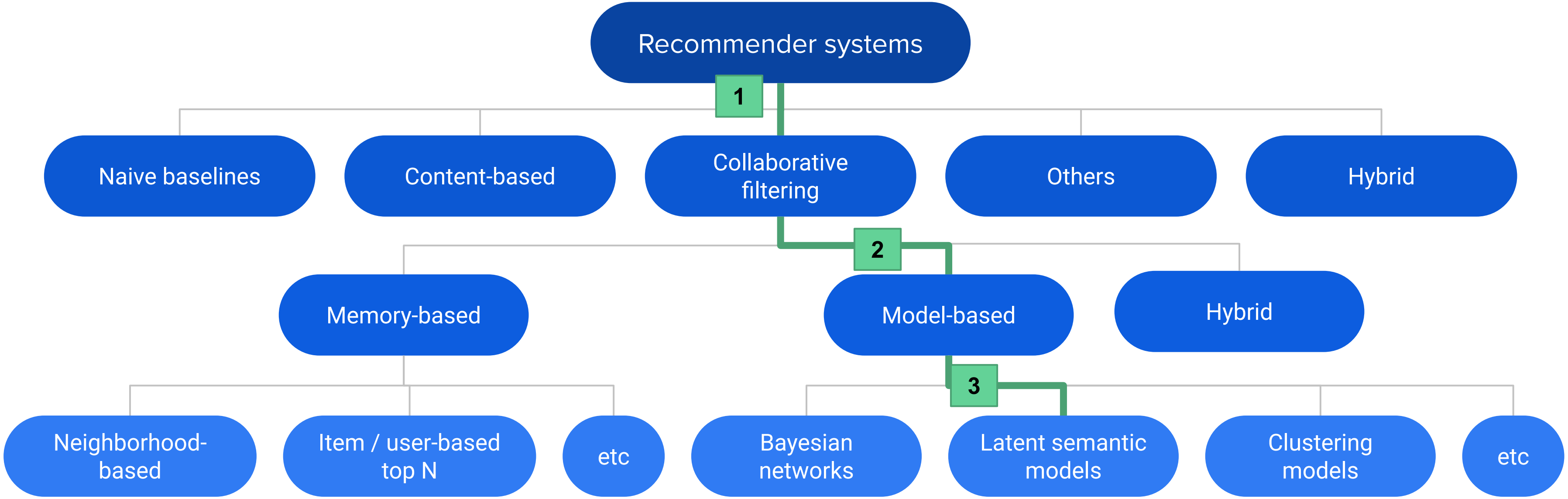
1. Embedding dimensions for playlists
2. Embedding dimensions for songs
3. Methods for reducing overfitting (e.g. pooling, regularizers)

Ensemble methods

- Ensemble models tend to be more robust because different approaches correct for each other's weaknesses
- Can explore combination of content-based and collaborative filtering to counter cold-start problem
- Can subsequently adjust voting weights for different models

Thank you

Appendix: Incomplete overview of recommender systems



1 Naive baselines cannot account for user preferences, problem has strong social element, which supports collaborative filtering

2 Memory-based suffers from problem of sparsity.

3 Data has very high dimensionality so dimensionality reduction techniques are suitable and should help robustness of model