# What is a Recommendation Engine?

## Who Cares? Okay, fine… then how does it work?

**Zach Miller - 2/15/18**

# Recommendation Engines are everywhere.

# What is a recommendation engine really?

- The goal of an Recommendation Engine is to play a game of “these are the most similar in this group.”

# What is a recommendation engine really?

- The goal of an Recommendation Engine is to play a game of “these are the most similar in this group.”
- You can modify that to do neat things though:
    - Find similar songs
    - Choose a book genre a user might like
    - Find movies a user might like
    - Measure dating compatibility

# What is a recommendation engine really?

- The goal of an Recommendation Engine is to play a game of “these are the most similar in this group.”
- You can modify that to do neat things though:
  - Find similar songs
  - Choose a book genre a user might like
  - Find movies a user might like
  - Measure dating compatibility
- These all do the same thing: try to measure some sort of “affinity”

# Let’s take a look at an example…

- I go to Amazon and browse books about puppies
- Amazon notices - “hey, that dude likes puppies.”
- Amazon then adds a book called, “Puppies Wearing Hats” to the *You Might Also Like…* section for me
- I buy a book called, “Puppies Wearing Hats”

# Let’s take a look at an example…

- I go to Amazon and browse books about puppies
- Amazon notices - “hey, that dude likes puppies.”
- **Amazon then adds a book called, “Puppies Wearing Hats” to the *You Might Also Like…* section for me**
- I buy a book called, “Puppies Wearing Hats”

**We’re going to focus on this section today. First, how does Amazon know which books are about puppies?**

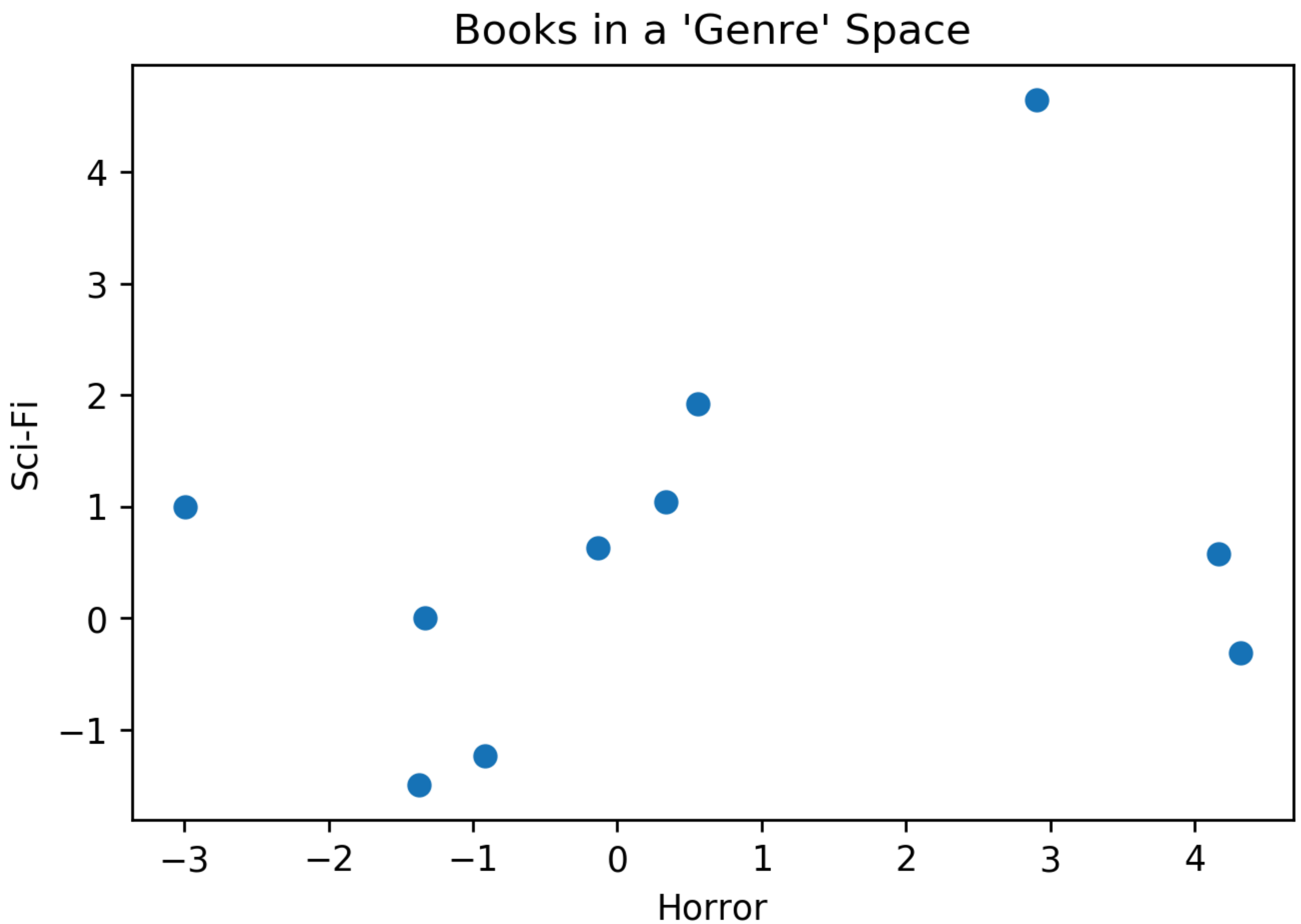
# Types of Recommendation Engines

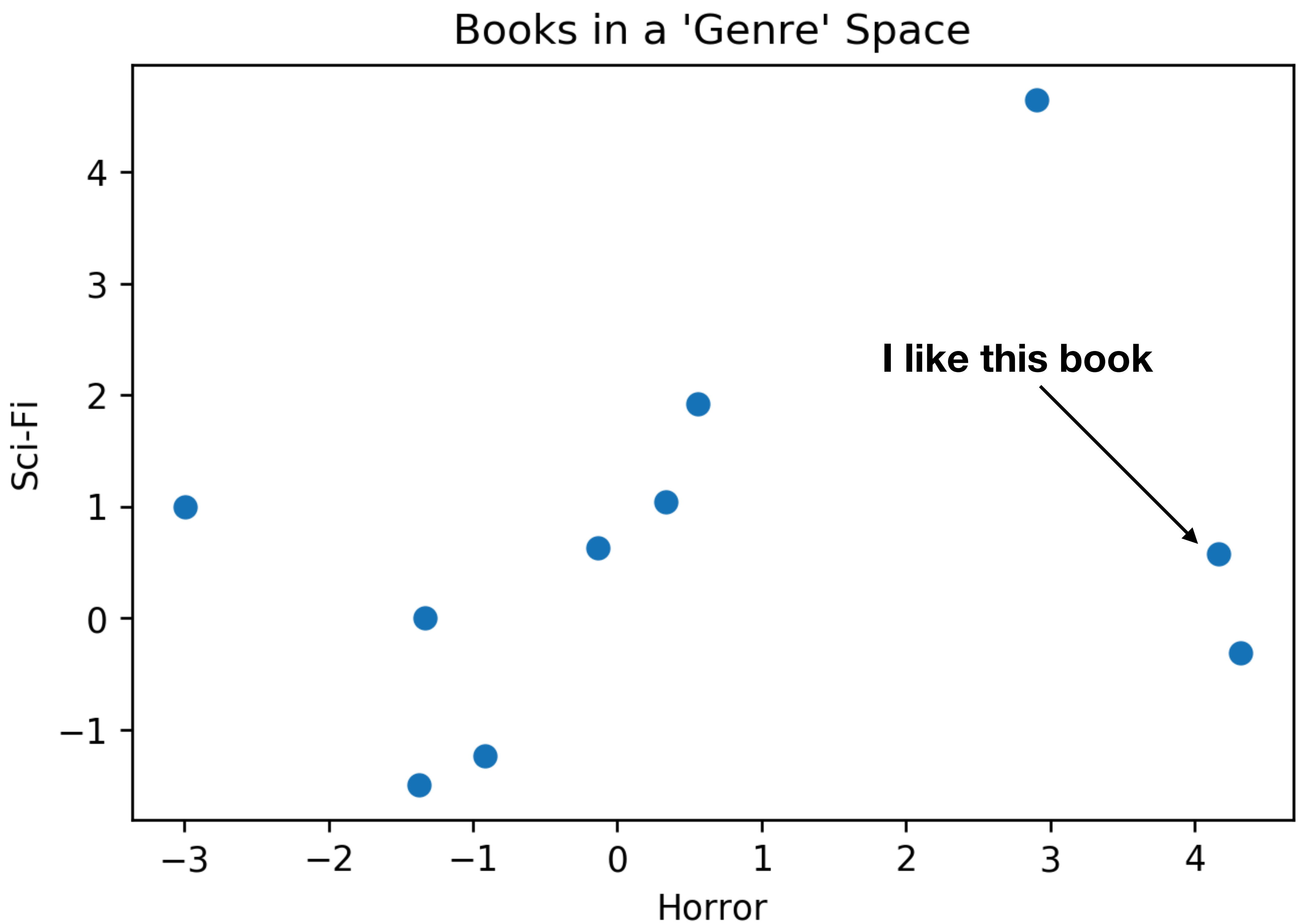
1. Content Based Filtering
2. Collaborative Filtering

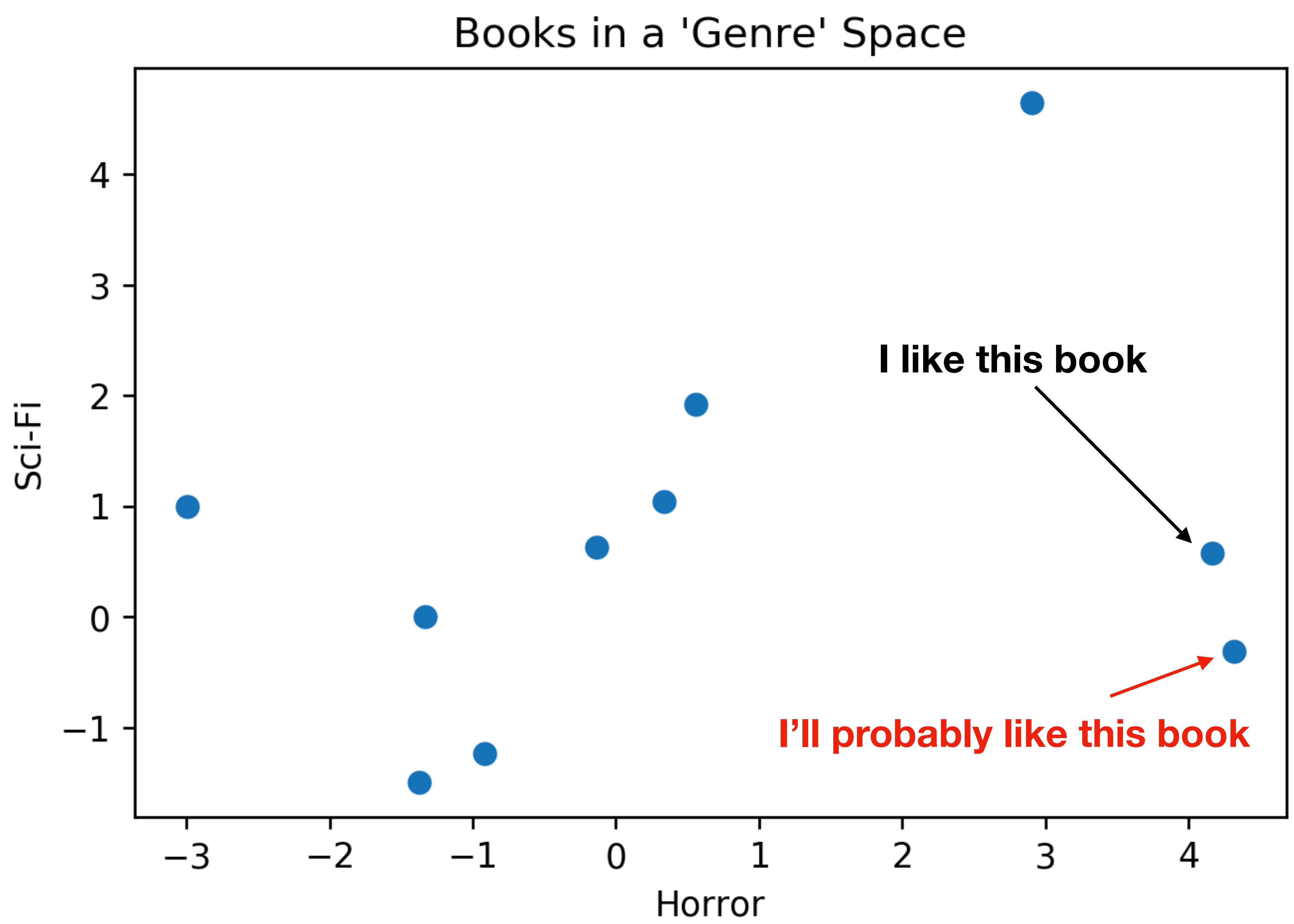
# Content Based Filtering

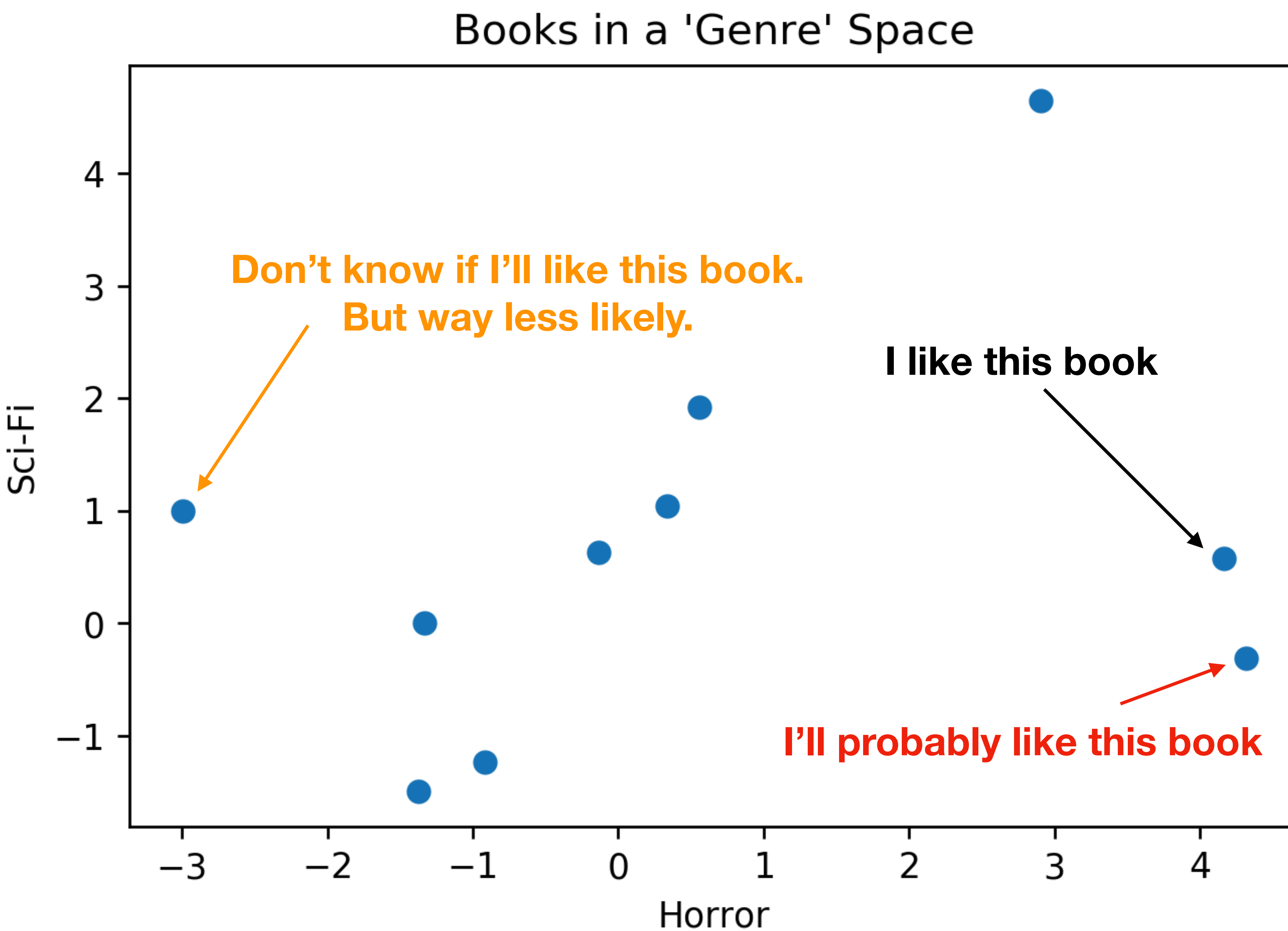
- Every item you want to recommend must have an explicit breakdown of what “makes it unique.” All of these breakdowns must use the same columns.

- Example: Pandora
  - Every song gets labeled by genre, beats per minute, lyric type, musicality, key, major vs minor, does have electric guitar, etc.
  - If someone likes a song that is lyric heavy, very fast, has lots of electric guitars, and is in a minor key… we just find other songs like that and recommend them!


Books in a 'Genre' Space
Sci-Fi
Horror
4
3
2
1
0
−1
−3
−2
−1
0
1
2
3
4

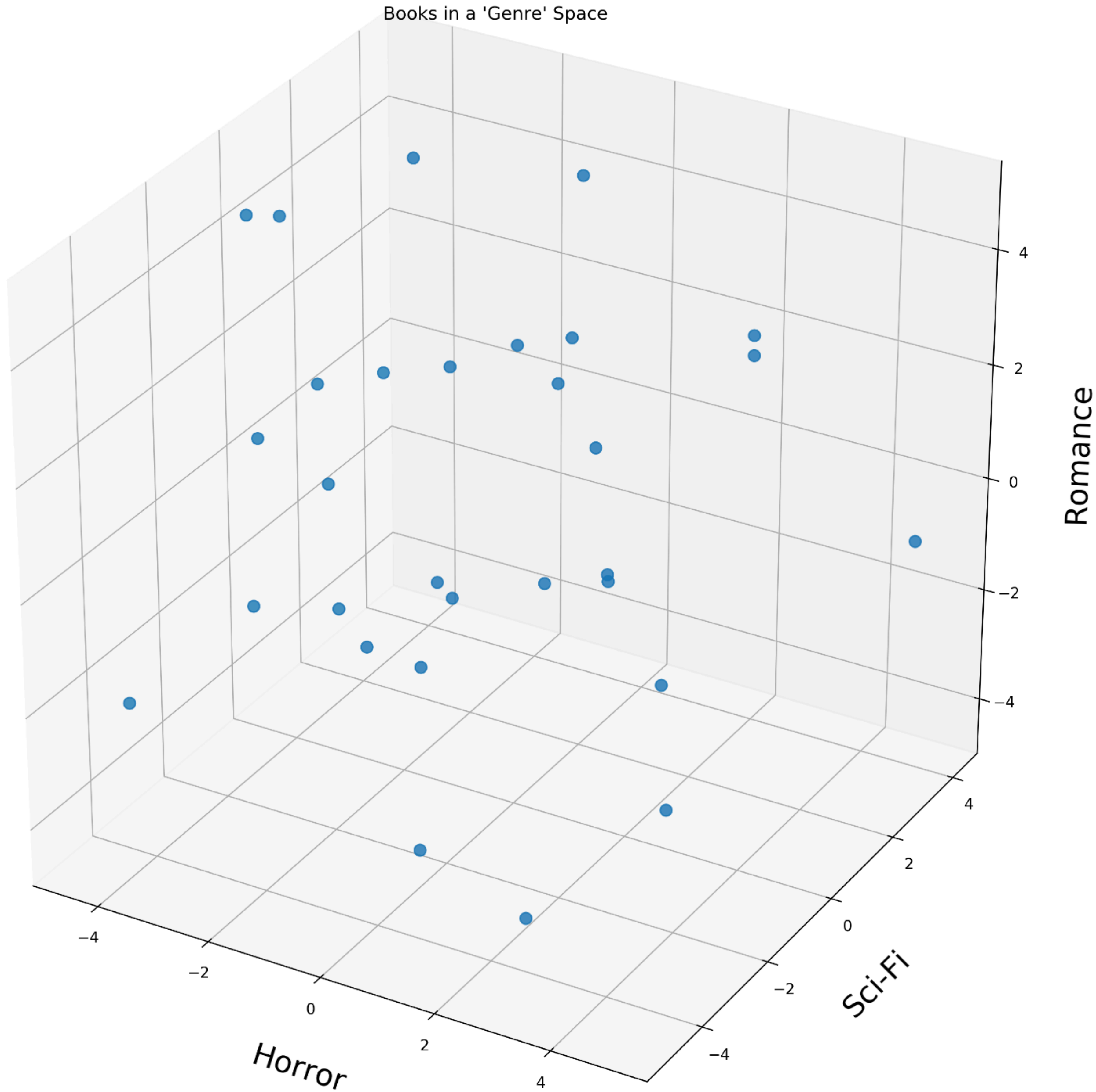

Books in a 'Genre' Space
I like this book
Sci-Fi
Horror
4
3
2
1
0
−1
−3
−2
−1
0
1
2
3
4


Books in a 'Genre' Space
Sci-Fi
Horror
4
3
2
1
0
−1
−3
−2
−1
0
1
2
3
4
I like this book
I'll probably like this book


Books in a 'Genre' Space
Don’t know if I’ll like this book.
But way less likely.
I like this book
I’ll probably like this book
Sci-Fi
Horror
4
3
2
1
0
−1
−3
−2
−1
0
1
2
3
4

# Content Based Filtering

- Can we get all of the items into the same “understanding” space?
- How far apart are they in that understanding space? If they’re close, then they are similar.
- Are we stuck in just 2D?


Books in a 'Genre' Space
Horror
Sci-Fi
Romance
−4
−2
0
2
4

# Content Based Filtering

- We can have as many dimensions as we need to correctly label things. Pandora has literally hundreds of dimensions for every song.
- How does it work in our Amazon example?
  - Amazon sees that I’m searching for a puppy book. They check out all their books and find the one with the most similar level of puppies and hats to the book I just looked at. They recommend the book with the same amount of puppies and hats.
- GREAT. So pros/cons?

# Content Based Filtering

Pros

- Allows us to recommend more of what a user likes
- Simple to understand - just recommend the most similar items
- Doesn't have to just be items - can map users and items to the same space and then recommend items closest to a user!

Cons

- Always recommend more of the same
- Have to map the items into the space - and that's usually done by hand!
- Hard to recommend across content type. You don't categorize books the same way you do songs - so we can't recommend across domains

If we always end up recommending more of the same, we’ll never find correlations like: “You like Star Wars, and most Star Wars fans also like Lord of the Rings - so we should recommend Lord of the Rings.”

# Collaborative Filtering

- We still need to create a “understanding space,” but now we do so by finding correlated “likes.”
- If we do this cleverly, we don’t even need to know anything about the specifics of the data. We don’t have to hand label the genre, the beats per minute, etc.
- We want to exploit what our users have already told us by rating our products - to figure out what other users will think of that product.

| | Star Wars | Lord of the Rings | Star Trek | The Notebook |
|---|---|---|---|---|
| Steve | 5 | 5 | 5 | 2 |
| Monroe | 1 | 1 | 1 | 5 |
| Graham | 3 | 3 | 3 | 4 |
| Lisa | 5 | 5 | ? | 1 |

| | Star Wars | Lord of the Rings | Star Trek | The Notebook |
|---|---|---|---|---|
| Steve | 5 | 5 | 5 | 2 |
| Monroe | 1 | 1 | 1 | 5 |
| Graham | 3 | 3 | 3 | 4 |
| Lisa | 5 | 5 | ? | 1 |

| | Star Wars | Lord of the Rings | Star Trek | The Notebook |
|---|---|---|---|---|
| Steve | 5 | 5 | 5 | 2 |
| Monroe | 1 | 1 | 1 | 5 |
| Graham | 3 | 3 | 3 | 4 |
| Lisa | 5 | 5 | ? | 1 |

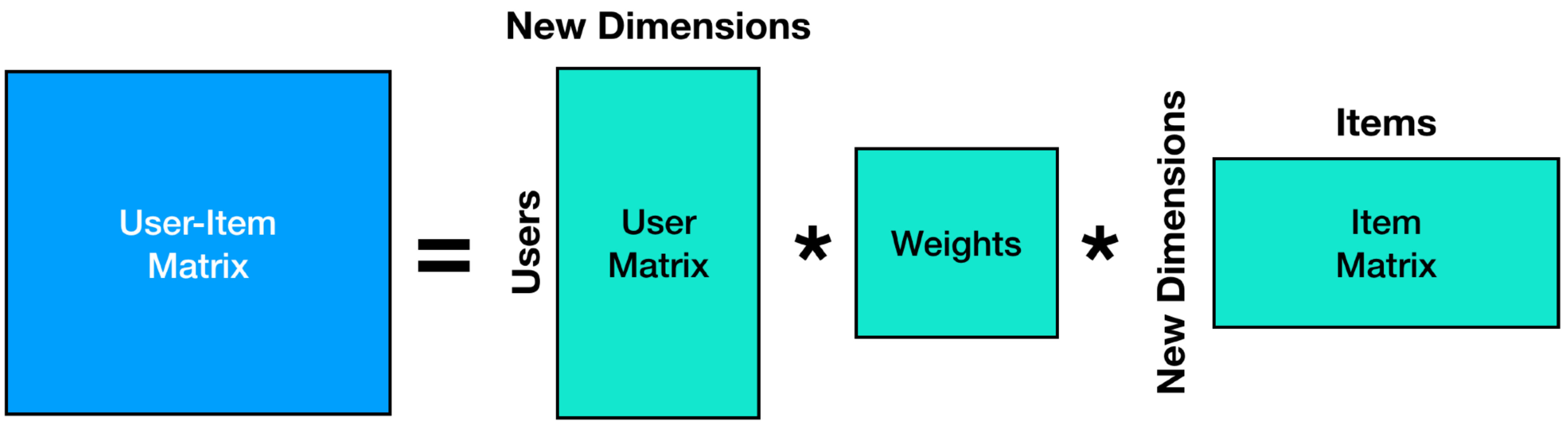
| | Star Wars | Lord of the Rings | Star Trek | The Notebook |
|---|---|---|---|---|
| Steve | 5 | 5 | 5 | 2 |
| Monroe | 1 | 1 | 1 | 5 |
| Graham | 3 | 3 | 3 | 4 |
| Lisa | 5 | 5 | **5** | 1 |

# Creating the latent space…

- We don’t want to always compare every movie that every person has seen to get an understanding of what movies they’ll like. It’s WAY too much data (as we’ll see later).

- Instead, we can use a method called Matrix Decomposition.

# Matrix Decomposition




**This specific form of Matrix Decomp. is called “Singular Value Decomposition” or SVD**

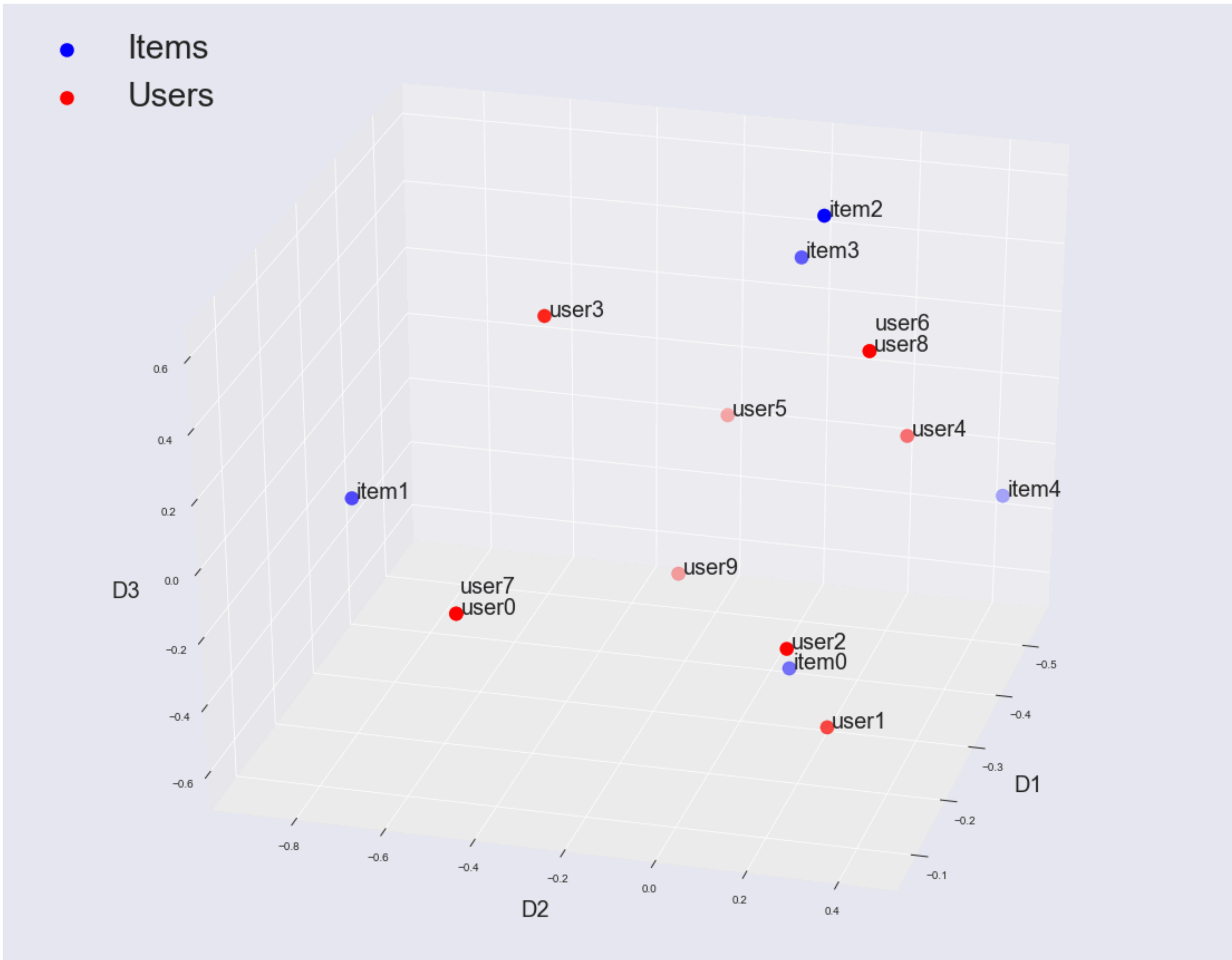
# What are these “New Dimensions?”

- They are “concepts” in the data. Example: if we were looking at movie data, these might be different genres that show up. However, we won’t really know what those genres are.

- We’ll see this in action in a bit. For now, think about it as a “concept space.”

- Once we have our concept space, we can still use the “who is closest to me” approach.

# What are these "New Dimensions?"



# Collaborative Filtering

- How does this work for our Amazon example?
  - Amazon looks at my profile and says, “Hey, that guy has liked a book about puppies. Of the 900 other users that liked that book, 650 also liked this book about puppies in hats. Let’s recommend that book to him.”
  - More specifically, they would find the “hidden concept space” based on what the users have liked and say, “in this hidden space, that guy is really close to the puppies in hats book… let’s recommend that.”

# Collaborative Filtering

Pros

- Exploits hidden correlations in our data
- Doesn’t require expensive hand mapping
- Can be applied across domains if we have user ratings/likes

Cons

- Need LOTS of data to start getting useful results
- Data tends to be REALLY sparse, so we have to handle that
- Every new user needs to give you lots of data before we can do anything.

# Hybrid Methods

- You can merge Content-based and Collaborative filtering methods. Most modern recommendation engines do this.

- Brings together the best of both worlds. Allowing for content to help guide the recommendations you see from users. Even better when you have TONS of items.

- Helps offset how much data you need from a new user.

So let's get started on some collaborative filtering with Python.