

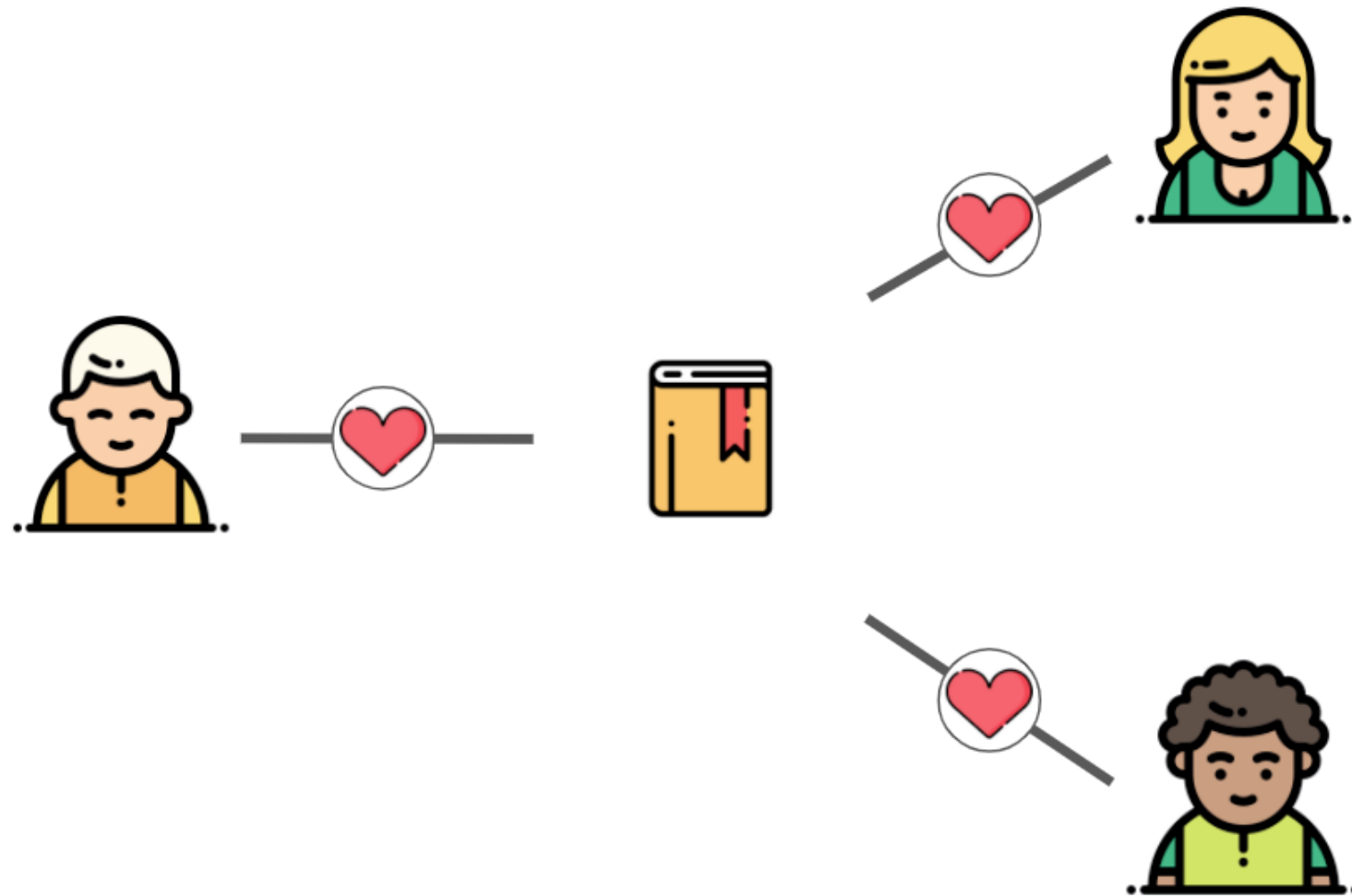
Collaborative filtering

BUILDING RECOMMENDATION ENGINES IN PYTHON

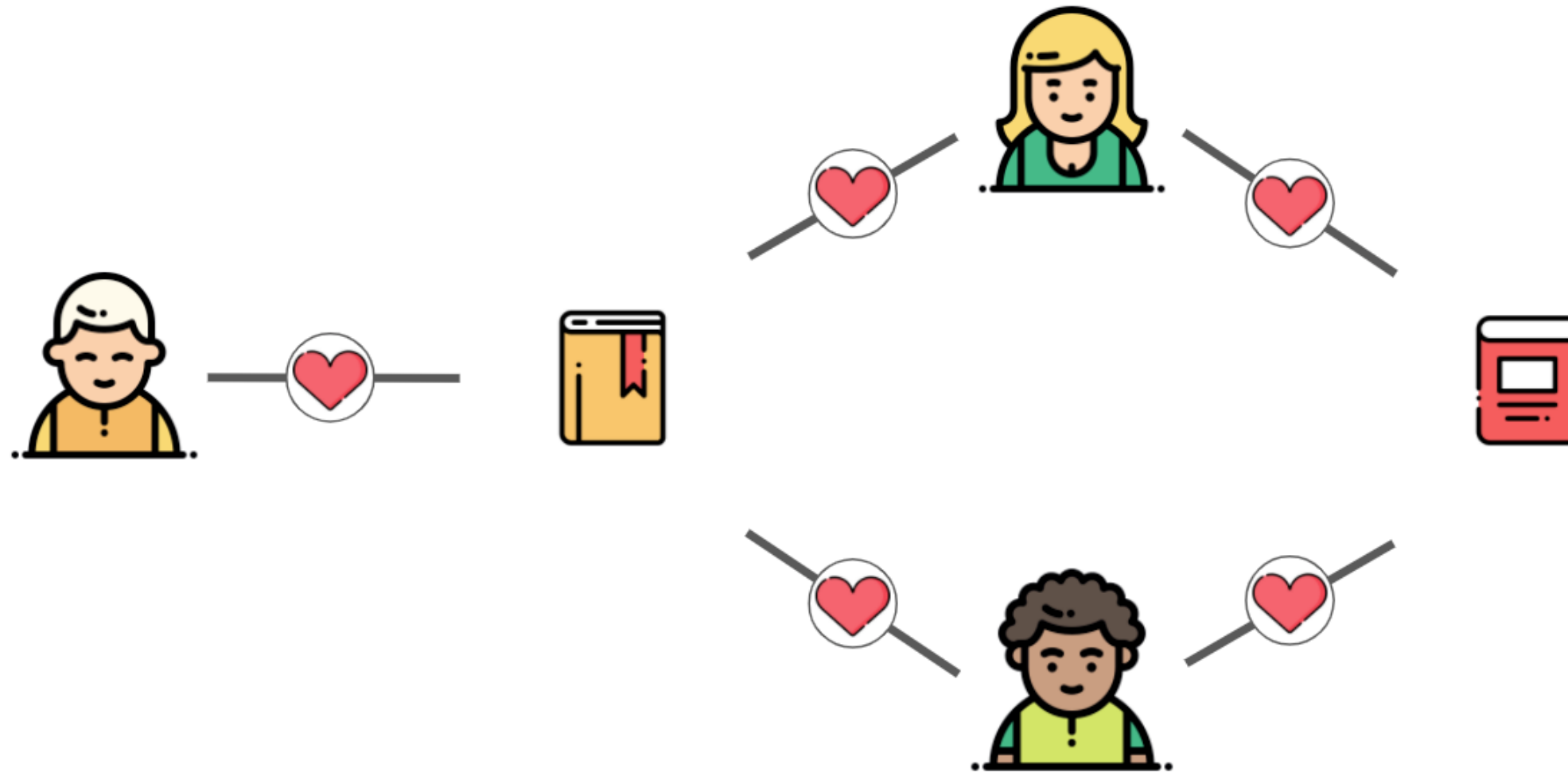


Rob O'Callaghan
Director of Data

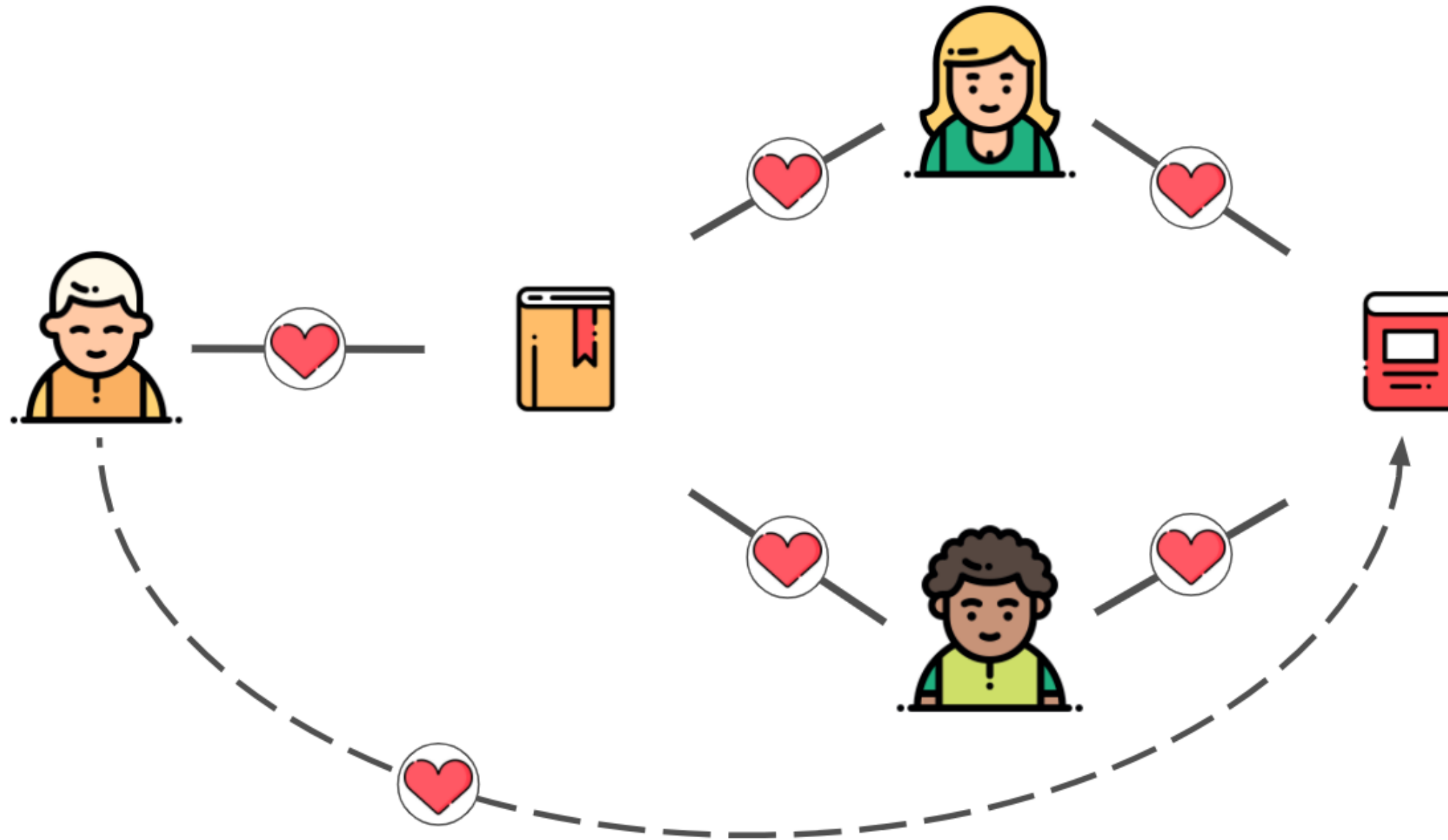
Collaborative filtering









Collaborative filtering









Collaborative filtering









Finding similar users

User ID	Item ID	Review
User 1	Item 1	
User 1	Item 4	
User 2	Item 2	
User 2	Item 4	
User 3	Item 1	
User 3	Item 4	

Finding similar users

User ID	Item ID	Review
User 1	Item 1	
User 1	Item 4	
User 2	Item 2	
User 2	Item 4	
User 3	Item 1	
User 3	Item 4	



	Item 1	Item 2	Item 3	Item 4
User 1				
User 2				
User 3				

Working with real data

`user_ratings` DataFrame:

User	Book	Rating
User_233	The Great Gatsby	3.0
User_651	The Catcher in the Rye	5.0
User_131	The Lord of the Rings	3.0
User_965	The Great Gatsby	4.0
User_651	Fifty Shades of Grey	4.0
...

Pivoting our data

```
user_ratings_pivot = user_ratings.pivot(index='User',  
                                         columns='Book',  
                                         values='Rating')  
  
print(user_ratings_pivot)
```

title	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User			
User_233	3.0	NaN	NaN
User_651	NaN	5.0	4.0
User_965	4.0	3.0	NaN
...

Data sparsity

title	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User			
User_233	3.0	NaN	NaN
User_651	NaN	5.0	4.0
User_965	4.0	3.0	NaN
...

```
print(user_ratings_pivot.dropna())
```

```
Empty DataFrame
Columns: ["The Great Gatsby", "The Catcher in the Rye", "Fifty Shades of Grey"]
Index: []
```

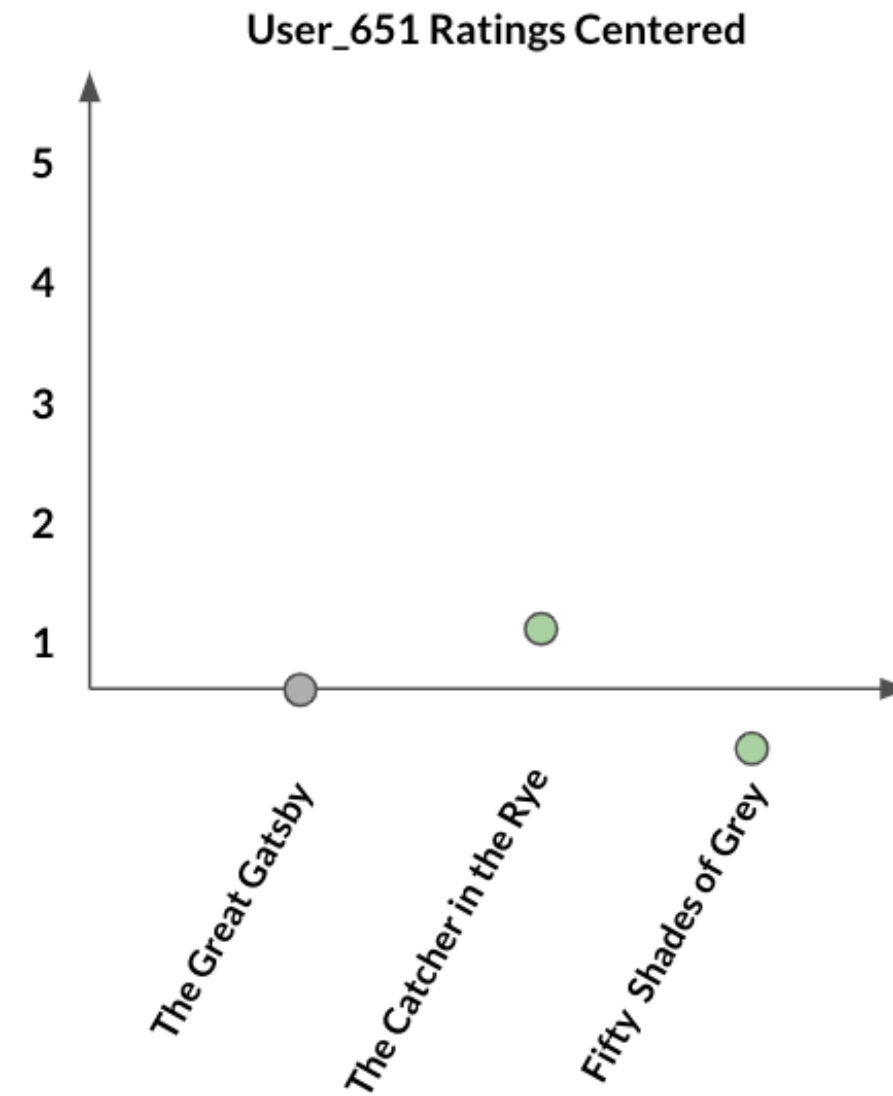
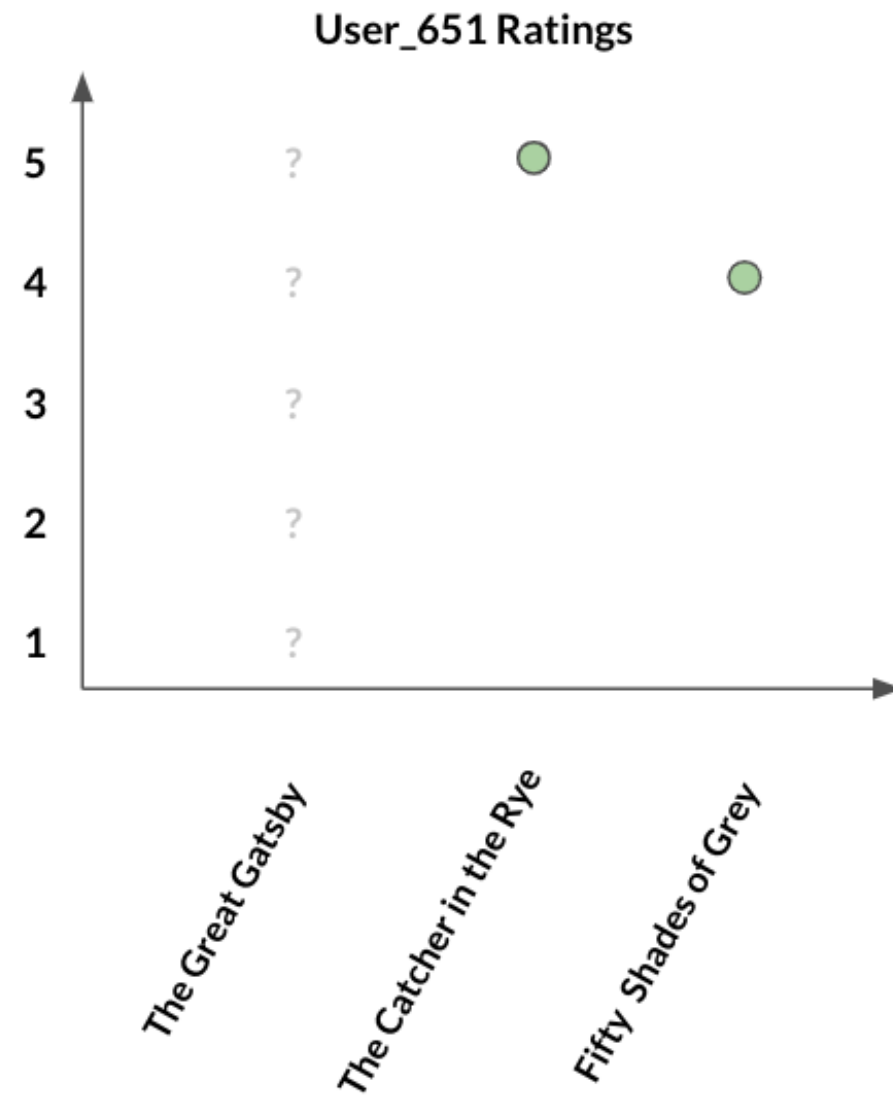
Filling the missing values

title	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User			
User_233	3.0	NaN	NaN
User_651	NaN	5.0	4.0
User_965	4.0	3.0	NaN
...

```
print(user_ratings_pivot["User_651"].fillna(0))
```

User_651	0.0	5.0	4.0
----------	-----	-----	-----

Filling the missing values



Filling the missing values

```
avg_ratings = user_ratings_pivot.mean(axis=1)
user_ratings_pivot = user_ratings_pivot.sub(avg_ratings, axis=0)
print(user_ratings_pivot)
```

title	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User			
User_233	0.0	NaN	NaN
User_651	NaN	0.5	-0.5
User_965	0.5	-0.5	NaN
...

Filling the missing values

```
user_ratings_pivot.fillna(0)
```

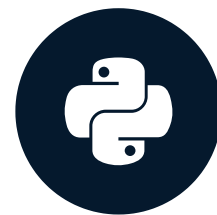
title	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User			
User_233	0.0	0.0	0.0
User_651	0.0	0.5	-0.5
User_965	0.5	-0.5	0.0
...

Let's practice!

BUILDING RECOMMENDATION ENGINES IN PYTHON

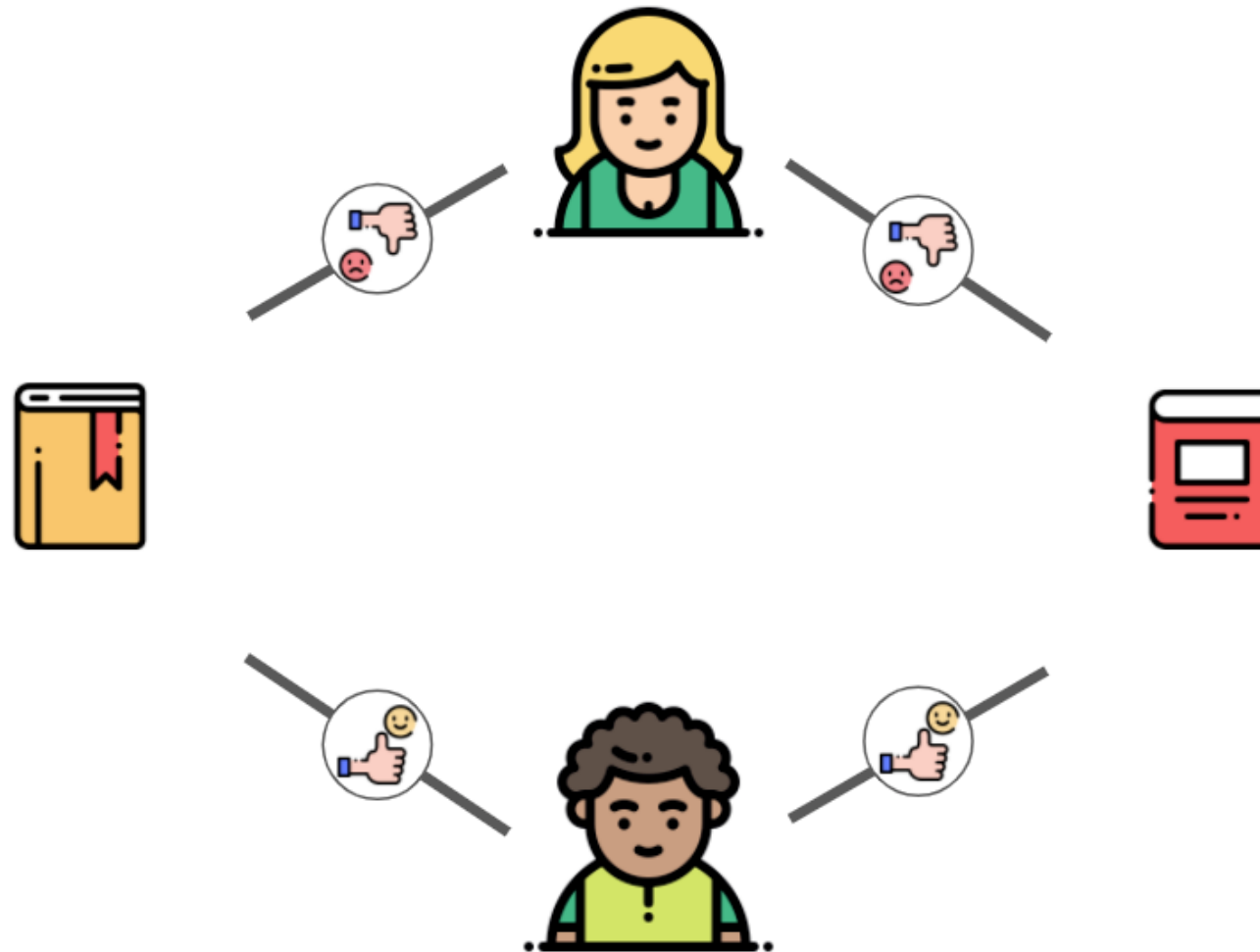
Finding similarities

BUILDING RECOMMENDATION ENGINES IN PYTHON

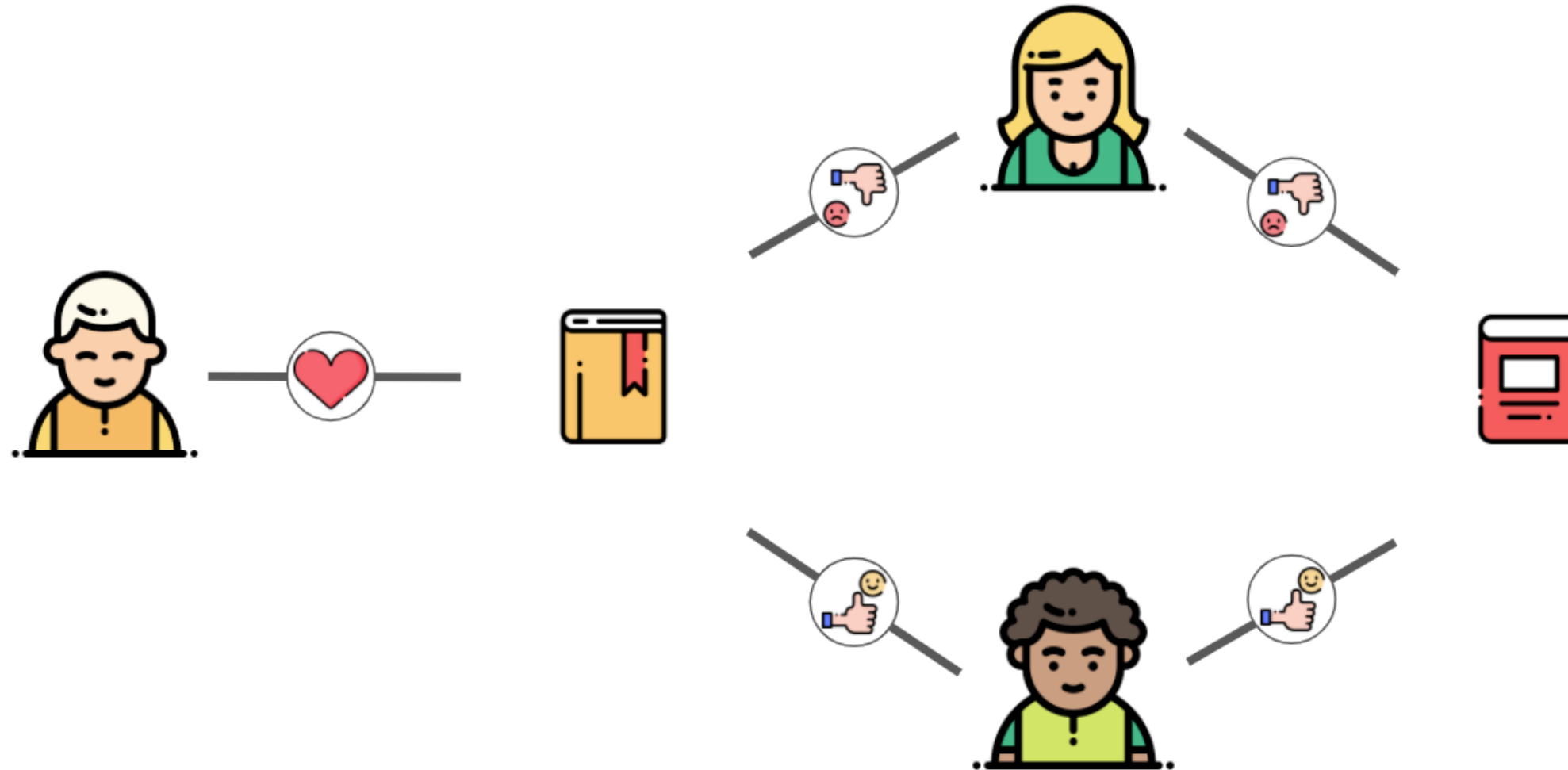


Rob O'Callaghan
Director of Data

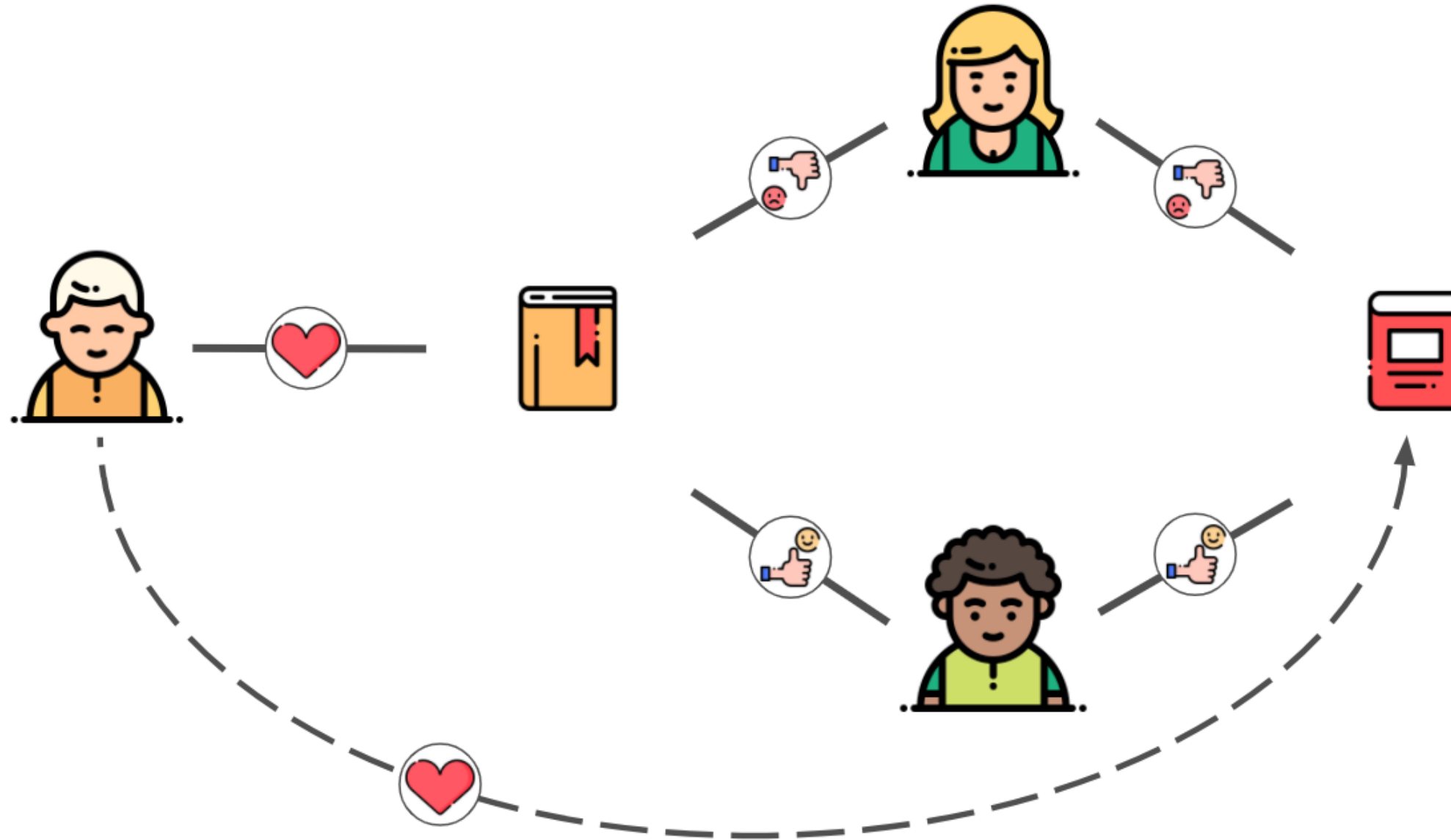
Item-based recommendations









Item-based recommendations



Item-based recommendations



User-based to item-based

	Item 1	Item 2	Item 3	Item 4
User 1				
User 2				
User 3				

User-based to item-based

User-based

	Item 1	Item 2	Item 3	Item 4
User 1				
User 2				
User 3				

Transpose



Item-based

	User 1	User 2	User 3
Item 1			
Item 2			
Item 3			
Item 4			

User-based to item-based

```
print(user_ratings_pivot):
```

	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
User_233	0.0	0.0	0.0
User_651	0.0	0.5	-0.5
User_965	0.5	-0.5	0.0
...

```
book_ratings_pivot = user_ratings_pivot.T  
print(book_ratings_pivot)
```

	User_233	User_651	User_965
The Great Gatsby	0.0	0.0	0.5
The Catcher in the Rye	0.0	0.5	-0.5
Fifty Shades of Grey	0.0	-0.5	0.0
...

Cosine similarities

book_ratings_pivot :

```
          User_233          User_651          User_965
The Great Gatsby      0.0            0.0            0.5
The Catcher in the Rye 0.0            0.5           -0.5
Fifty Shades of Grey  0.0           -0.5            0.0
...                  ...            ...            ...
```

Cosine similarities

```
cosine_similarity(  
    ,  
)
```

Cosine similarities

```
cosine_similarity(book_ratings_pivot.loc['Lord of the Rings', :],  
                  book_ratings_pivot.loc['The Hobbit', :])
```


Cosine similarities

```
cosine_similarity(book_ratings_pivot.loc['Lord of the Rings', :].values  
                  book_ratings_pivot.loc['The Hobbit', :].values)
```

Cosine similarities

```
cosine_similarity(book_ratings_pivot.loc['Lord of the Rings', :].values.reshape(1, -1),  
                  book_ratings_pivot.loc['The Hobbit', :].values.reshape(1, -1))
```

0.43

```
cosine_similarity(book_ratnngs.loc['Lord of the Rings', :].values.reshape(1, -1),  
                  book_ratnngs.loc['Twilight', :].values.reshape(1, -1))
```

-0.64

Cosine similarities

```
similarities = cosine_similarity(book_ratings_pivot)
cosine_similarity_df = pd.DataFrame(book_ratings_pivot,
                                   index=book_ratings_pivot.index,
                                   columns=book_ratings_pivot.index)

cosine_similarity_df.head()
```

	The Great Gatsby	The Catcher in the Rye	Fifty Shades of Grey
The Great Gatsby	1.0	0.0	-0.3
The Catcher in the Rye	0.0	1.0	-0.5
Fifty Shades of Grey	-0.3	-0.5	1.0
...

Cosine similarities

```
cosine_similarity_series = cosine_similarity_df.loc['The Hobbit']  
ordered_similarities = cosine_similarity_series.sort_values(ascending=False)  
print(ordered_similarities)
```

```
The Hobbit          1.00  
Lord of the Rings   0.43  
The Silmarillion    0.37  
...
```

Let's practice!

BUILDING RECOMMENDATION ENGINES IN PYTHON

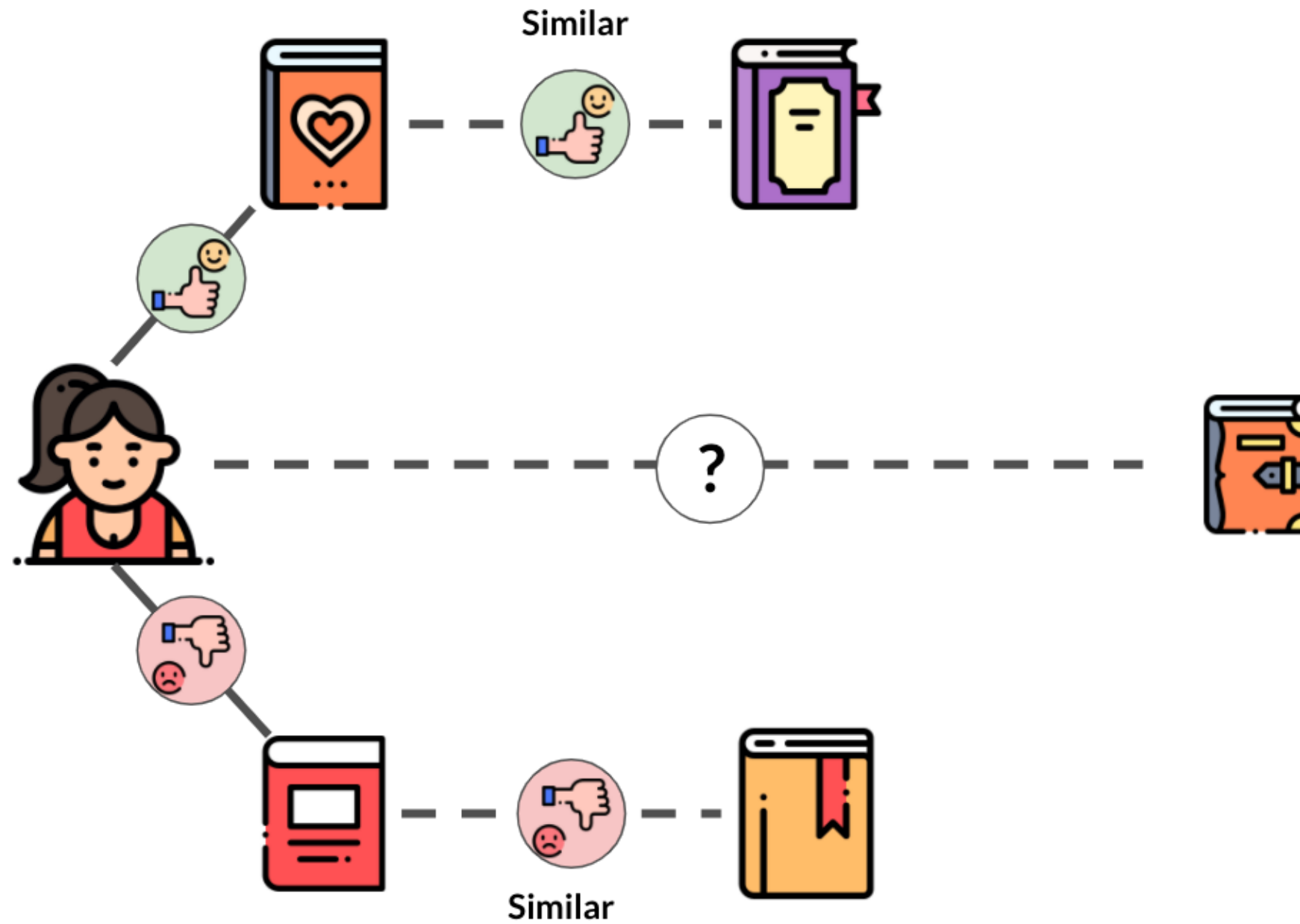
Using K-nearest neighbors

BUILDING RECOMMENDATION ENGINES IN PYTHON

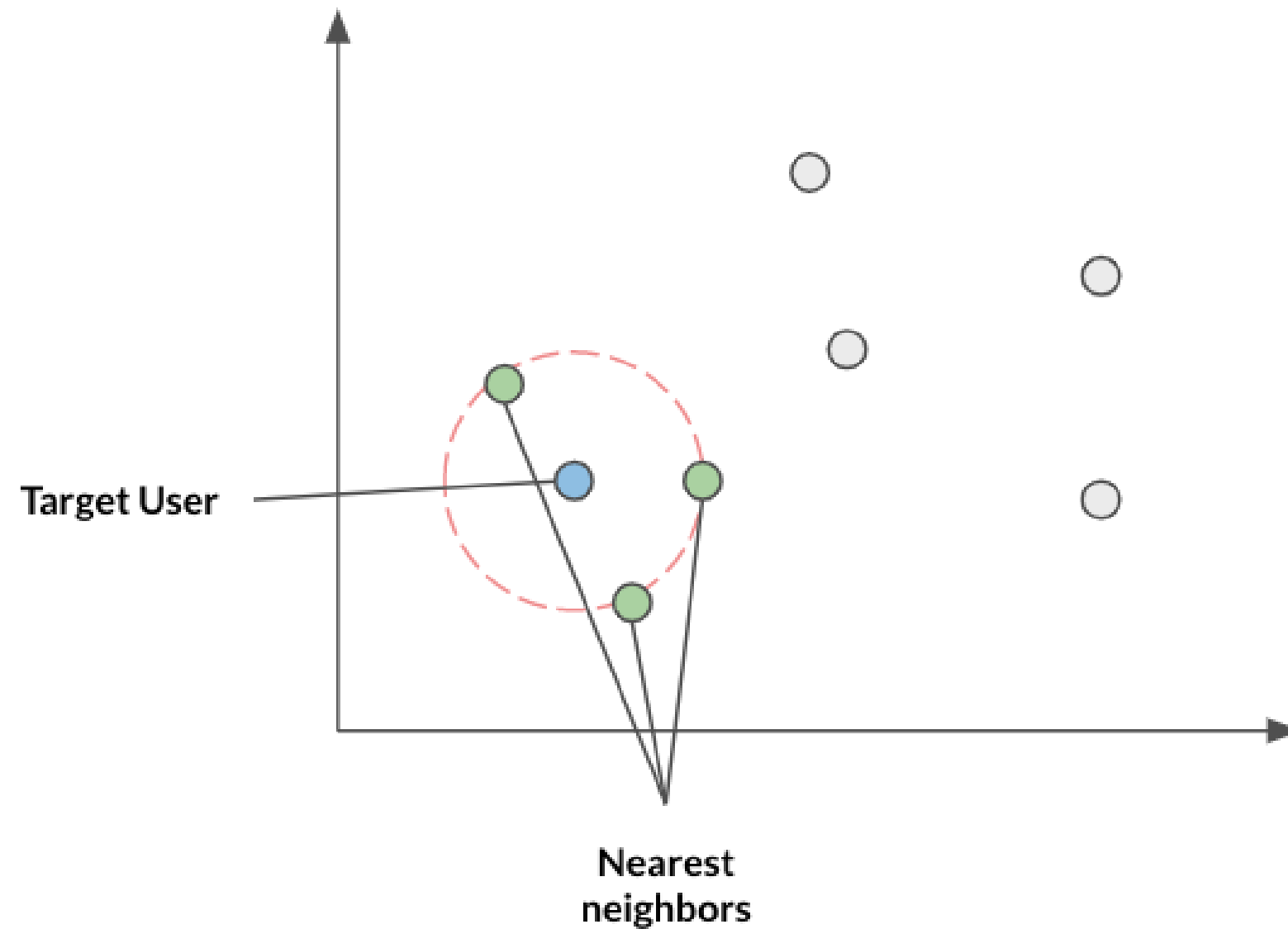


Rob O'Callaghan
Director of Data

Beyond similar items



K-nearest neighbors



User-user similarity

```
similarities = cosine_similarity(user_ratings_pivot)
cosine_similarity_df = pd.DataFrame(user_ratings_pivot,
                                    index=user_ratings_pivot.index,
                                    columns=user_ratings_pivot.index)

cosine_similarity_df.head()
```

	User 001	User 002	User 003
User 001	1.0	-0.4	0.3
User 002	-0.4	1.0	-0.5
User 003	0.3	-0.5	1.0
...

Understanding the similarity matrix

	User 001	User 002	User 003
User 001	1.0	-0.4	0.3
User 002	-0.4	1.0	-0.5
User 003	0.3	-0.5	1.0
...

Understanding the similarity matrix

	v		
	User 001	User 002	User 003
User 001	1.0	-0.4	0.3 <-
User 002	-0.4	1.0	-0.5
User 003	0.3	-0.5	1.0
...

Understanding the similarity matrix

	v		
	User 001	User 002	User 003
User 001	1.0	-0.4	<- 0.3
User 002	-0.4	1.0	-0.5
User 003	0.3	-0.5	1.0
...

Step by step KNN

```
user_similarity_series = user_similarities.loc['user_001']  
ordered_similarities = user_similarity_series.sort_values(ascending=False)  
nearest_neighbors = ordered_similarities[1:4].index  
print(nearest_neighbors)
```

```
user_007  
user_042  
user_003
```

Step by step KNN

```
neighbor_ratings = user_ratings_table.reindex(nearest_neighbors)
neighbor_ratings['Catch-22'].mean()
```

3.2

Using scikit-learn's KNN

```
print(user_ratings_pivot)
```

	The Great Gatsby	Catch-22	Fifty Shades of Grey
User_233	0.0	0.0	0.0
User_651	0.0	0.5	-0.5
...

```
print(user_ratings_table)
```

	The Great Gatsby	Catch-22	Fifty Shades of Grey
User_233	NaN	NaN	NaN
User_651	NaN	5.0	4.0
...

Using scikit-learn's KNN

```
user_ratings_pivot.drop("Catch-22", axis=1, inplace=True)
target_user_x = user_ratings_pivot.loc[["user_001"]]
print(target_user_x)
```

	The Great Gatsby	Fifty Shades of Grey	Iliad
User_001	4.0	2.0	3.0

```
other_users_y = user_ratings_table["Catch-22"]
print(other_users_y)
```

```
[NaN, '5.0', '3.0', '4.0', '5.0' ...]
```


Using scikit-learn's KNN

```
other_users_x = user_ratings_pivot[other_users_y.notnull()]  
print(other_users_x)
```

	The Great Gatsby	Fifty Shades of Grey	Iliad
User_651	0.0	-0.5	-0.5
User_442	1.0	0.0	1.0
...

```
other_users_y.dropna(inplace=True)  
print(other_users_y)
```

```
['5.0', '3.0', '4.0', '5.0' ...]
```

Using scikit-learn's KNN

```
from sklearn.neighbors import KNeighborsRegressor
user_knn = KNeighborsRegressor(metric='cosine', n_neighbors=3)
user_knn.fit(other_users_x, other_users_y)
user_user_pred = user_knn.predict(target_user_x)
print(user_user_pred)
```

3.3

Using scikit-learn's KNN

```
from sklearn.neighbors import KNeighborsClassifier
user_knn = KNeighborsClassifier(metric='cosine', n_neighbors=3)
user_knn.fit(other_users_x, other_users_y)
user_user_pred = user_knn.predict(target_user_x)
print(user_user_pred)
```

3

Let's practice!

BUILDING RECOMMENDATION ENGINES IN PYTHON

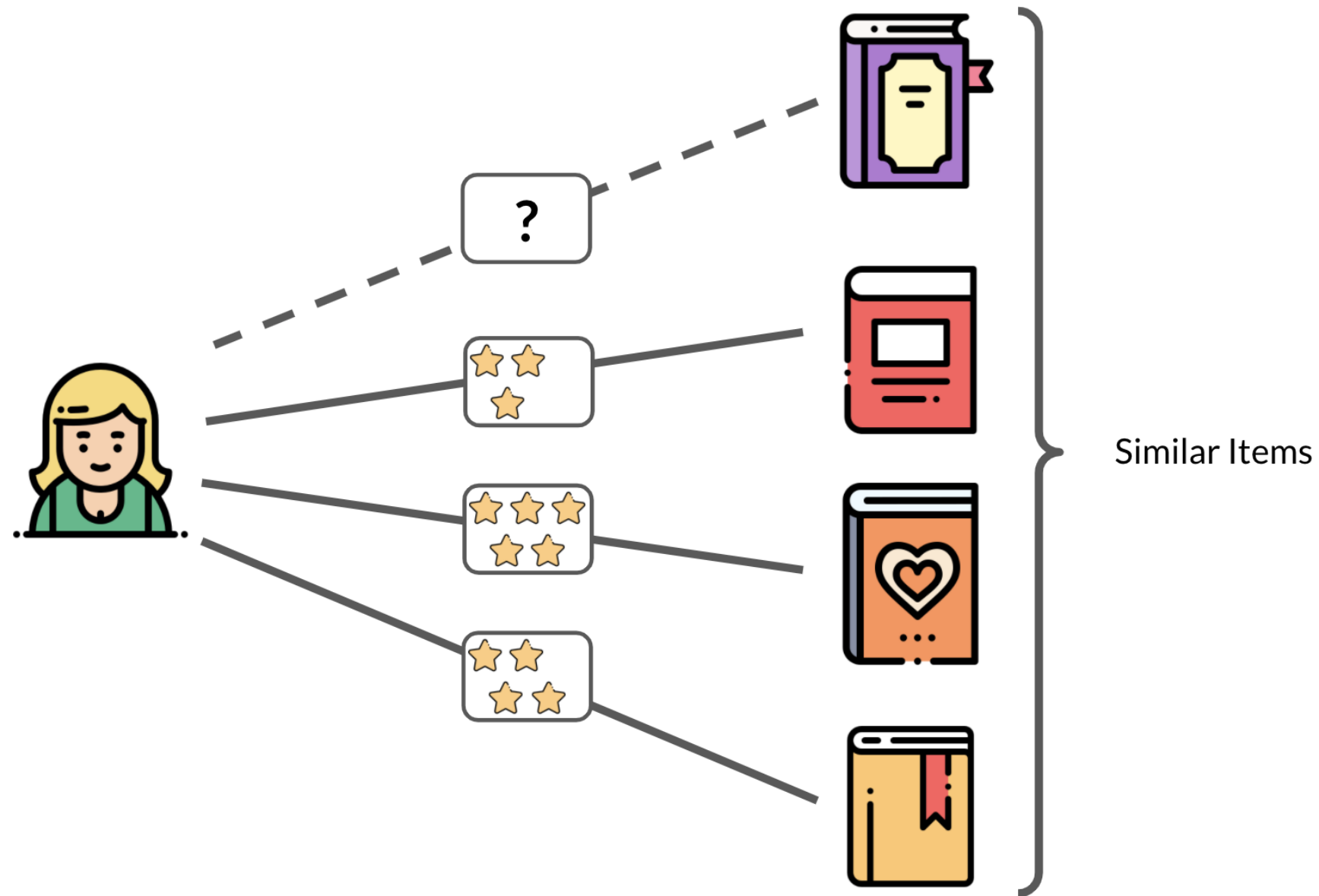
Item-based or user-based

BUILDING RECOMMENDATION ENGINES IN PYTHON

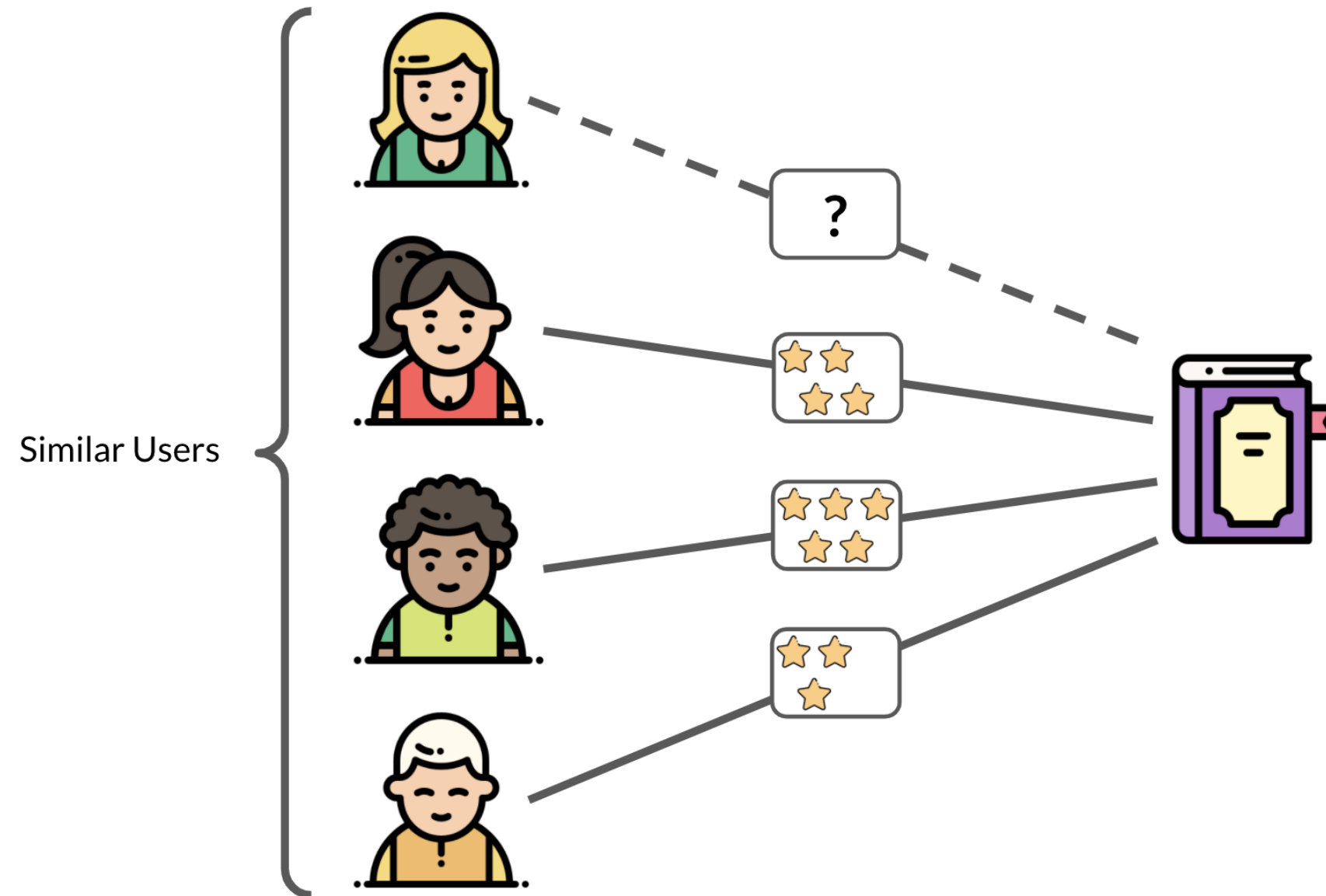


Rob O'Callaghan
Director of Data

Item-based filtering



User-based filtering



Why use item-based filtering?

Pros:

- Item-based recommendations are more consistent over time
- Item-based recommendations can be easier to explain
- Item-based recommendations can be pre-calculated

Cons:

- Item-based recommendations result in very obvious suggestions

Why use user-based filtering?

Pros:

- User-based recommendations can create a lot more interesting suggestions

Cons:

- Generally beaten by item-based recommendations using standard metrics

Let's practice!

BUILDING RECOMMENDATION ENGINES IN PYTHON