

RECOMMENDED R SYSTEMS

What are recommender systems?

RECOMMENDER SYSTEMS

A recommendation system aims to match users to products/items/brand/etc that they likely haven't experienced yet.


This rating is produced by analyzing other user/item ratings (and sometimes item characteristics) to provide personalized recommendations to users.

EXAMPLES OF RECOMMENDER SYSTEMS

Customers who bought this item also bought

Pa






Python for Data Analysis:
Data Wrangling with Pandas, NumPy, and...
› Wes McKinney
★★★★☆ 144
Paperback
\$26.90 ✓prime



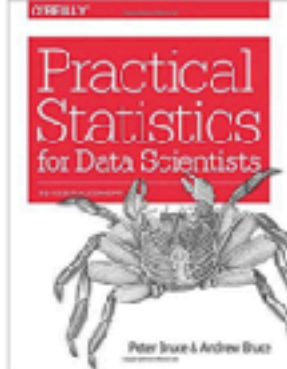
Python Machine Learning
› Sebastian Raschka
★★★★☆ 107
Paperback
\$40.49 ✓prime



Python Data Science Handbook: Essential Tools for Working with Data
› Jake VanderPlas
★★★★☆ 23
Paperback
\$45.47 ✓prime



Hands-On Machine Learning with Scikit-Learn and TensorFlow...
› Aurélien Géron
★★★★☆ 49
#1 Best Seller in Computer Neural Networks
Paperback
\$31.57 ✓prime

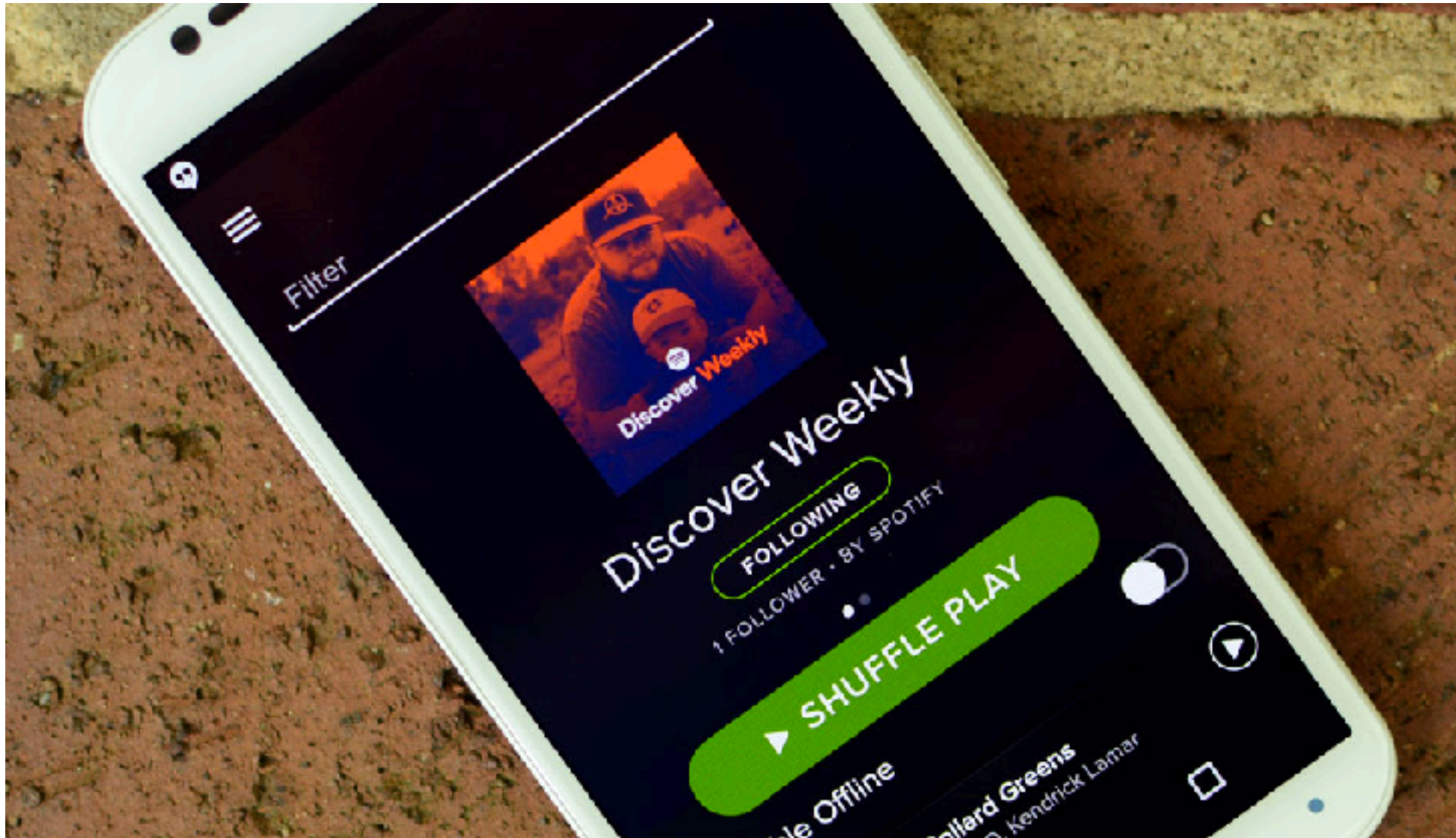


Practical Statistics for Data Scientists: 50 Essential Concepts
› Peter Bruce
★★★★☆ 10
Paperback
\$32.72 ✓prime



Introduction to Machine Learning with Python: A Guide for Data Scientists
Andreas C. Müller
★★★★☆ 19
Paperback
\$40.75 ✓prime

EXAMPLES OF RECOMMENDER SYSTEMS



EXAMPLES OF RECOMMENDER SYSTEMS



EXAMPLES OF RECOMMENDER SYSTEMS

People You May Know

Add people you know as friends and connect with public profiles you like.



Lala Lalabs ×
Add as friend



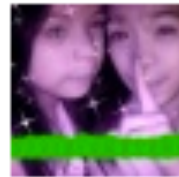
Brian Crecente ×
Add as friend



Brian Ashcraft ×
Add as friend



justin bieber ×
Add as friend



Camile Gozon ×
Add as friend



Karla Danielle Beger ×
Add as friend



Taylor-Alison Swifty ×
Add as friend

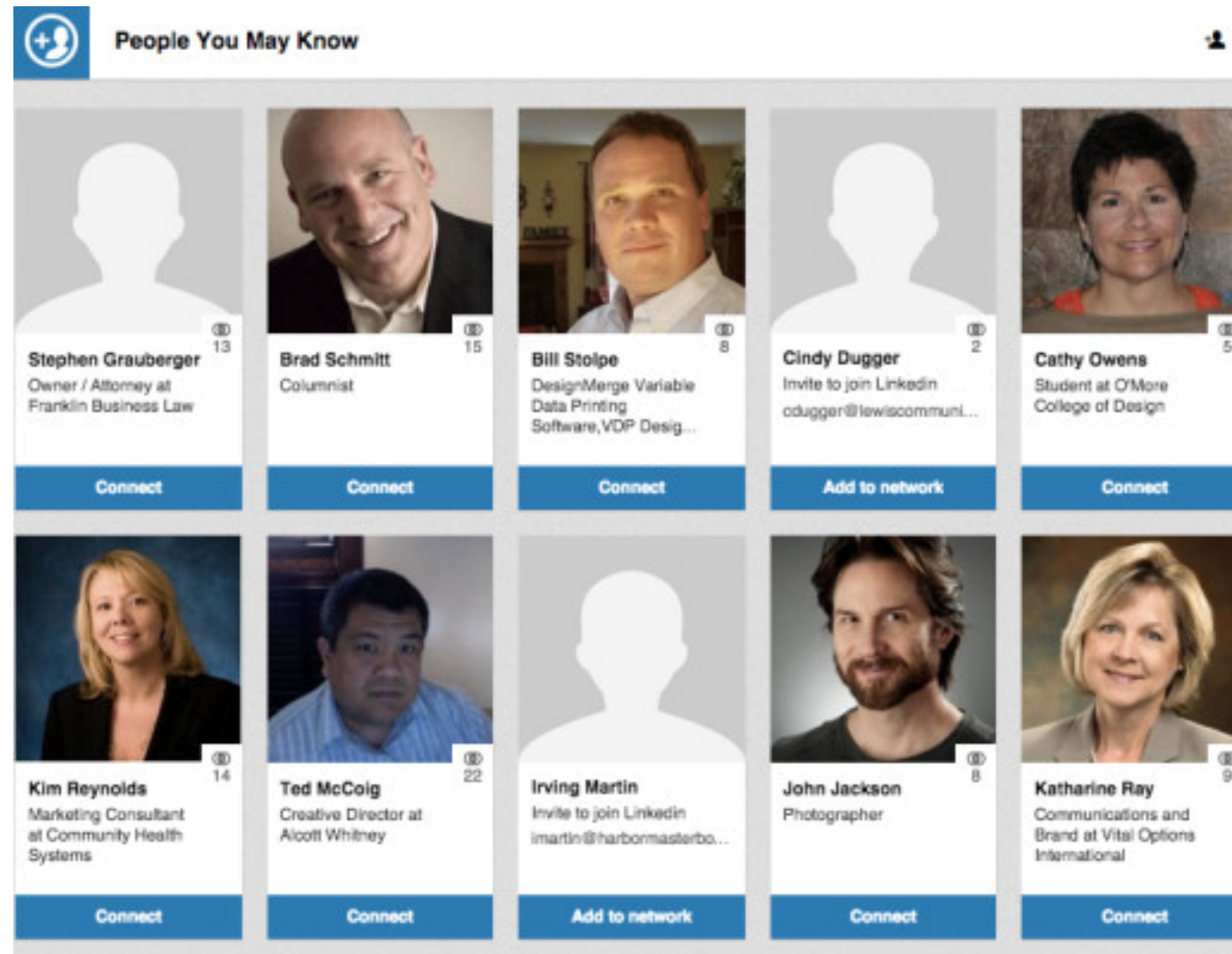


Adam Rifkin ×
Add as friend



Luke Plunkett ×
Add as friend

EXAMPLES OF RECOMMENDER SYSTEMS



EXAMPLES OF RECOMMENDER SYSTEMS - ACTIVITY

Spend 15 minutes finding as many examples of recommender systems as you can, then share your two favorites with the class while the instructor writes them on the board.

Different types of recommender systems

NON-PERSONALIZED

Non-personalized recommender systems recommend items without using any user information. Therefore, every user will receive the same recommendations. One example would include recommending the most popular products.

CONTENT

In **content-based filtering**, items are mapped into a feature space, and recommendations depend on *item characteristics*.

CONTENT

Content-based filtering begins by mapping each item into a feature space. Both users and items are represented by vectors in this space.

Item vectors measure the degree to which the item is described by each feature, and ***user vectors*** measure a user's preferences for each feature.

CONTENT

Ratings are generated by taking **dot products** of user & item vectors.

CONTENT - EXAMPLE

features = (big box office, kid friendly, famous actors)

Items (movies)

Finding Nemo = (5, 5, 2)

Mission Impossible = (3, -5, 5)

Jiro Dreams of Sushi (-4, -5, -5)

Prediction

$$5 \cdot -3 + 5 \cdot 2 + 2 \cdot -2 = -9$$

$$3 \cdot -3 + -5 \cdot 2 + 5 \cdot -2 = -29$$

$$\mathbf{-4 \cdot -3 + -5 \cdot 2 + -5 \cdot -2 = +12}$$

User

(-3, 2, -2)



COLLABORATIVE

Collaborative filtering refers to a family of methods for predicting ratings where instead of thinking about users and items in terms of a feature space, we are *only* interested in the existing user-item ratings themselves.

In this case, our dataset is a *ratings matrix* whose columns correspond to items, and whose rows correspond to users.

COLLABORATIVE

A diagram illustrating a user-item matrix. On the left, a vertical double-headed arrow is labeled "480,000 users". Above the matrix, a horizontal double-headed arrow is labeled "18,000 movies". The matrix itself is a grid of 8 rows and 6 columns, with light green cells. The first column contains 'x' in every row. The second column contains the values 1, x, x, 4, x, 5, x, 1. The third column contains the values 1, x, 3, 3, x, x, 3, x. The fourth column contains 'x' in every row. The fifth column contains '...' in every row. The sixth column contains the values x, x, x, 2, x, x, x, 2.

| | | | | | |
|-----|---|---|---|-----|---|
| x | 1 | 1 | x | ... | x |
| x | x | x | 5 | ... | x |
| x | x | 3 | x | ... | x |
| x | 4 | 3 | x | ... | 2 |
| ... | x | x | x | ... | x |
| x | 5 | x | 1 | ... | x |
| x | x | 3 | 3 | ... | x |
| x | 1 | x | x | ... | 2 |

The user-item matrix will almost always be very sparse

COLLABORATIVE

There are two types of collaborative filtering:

- 1) **User-to-user**, where we find similar users based on ratings and recommend items from those users
- 2) **Item-to-item**, where we find similar items based on ratings and recommend the most similar items

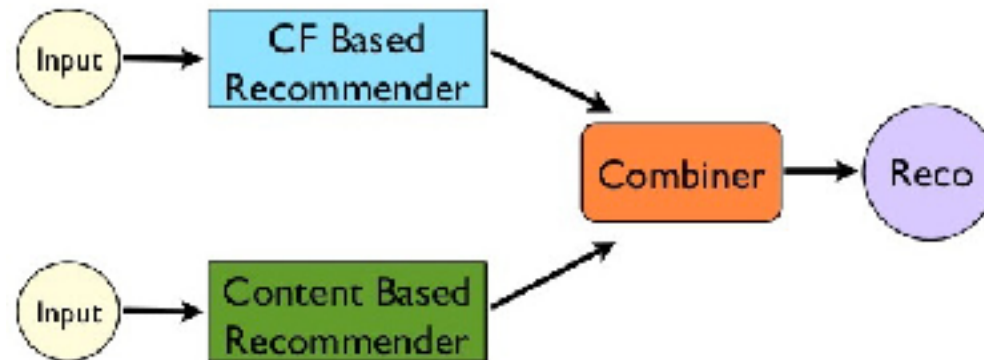
PROS AND CONS

| | Pros | Cons |
|-------------------------|--|--|
| Non-personalized | <ul style="list-style-type: none">• Doesn't require any user-data• No cold-start issues• Easy to build and maintain | <ul style="list-style-type: none">• No personalization• Recommendations are obvious |
| Content | <ul style="list-style-type: none">• No cold-start issues• Recommendations are transparent | <ul style="list-style-type: none">• Requires feature selection and/or engineering• Tend to recommend non-surprising items |
| Collaborative | <ul style="list-style-type: none">• Doesn't require feature selection and/or engineering• Can capture non-quantifiable qualities• Provides serendipitous recommendations | <ul style="list-style-type: none">• Suffers from a ramp-up problem• Can be computationally intensive |

HYBRID

Different recommender systems have different pros and cons. Combining multiple types of recommender systems can create better recommendations.

Hybrid Recommendations



Other considerations

THE COLD-START PROBLEM

The cold start problem arises because we've been relying only on ratings data, or on explicit feedback from users.

Until users rate several items, we don't know anything about their preferences!

We can get around this by enhancing our recommendations using implicit feedback, which may include things like item browsing behavior, search patterns, purchase history, etc.

THE COLD-START PROBLEM

While explicit feedback (ratings, likes, purchases) leads to high quality ratings, the data is sparse and cold starts are problematic.

Meanwhile implicit feedback (browsing behavior, etc.) leads to less accurate ratings, but the data is much more dense (and less invasive to collect).

EVALUATING RECOMMENDER SYSTEMS

There are two main ways to evaluate recommender systems:

- 1) **Offline**, where the recommender is evaluated using cross-validation. A training dataset is used to build the system, and then the system is used to predict item ratings in the test set. An error metric is then calculated.
- 2) **Online**, where systems are compared using A/B testing and metrics such as click through rate (CTR) or conversion rate (CR) are measured.

Jupyter Notebook Examples