

KDnuggets

[Subscribe to KDnuggets News](#)



- [Blog/News](#)
- [Opinions](#)
- [Tutorials](#)
- [Top stories](#)
- [Companies](#)
- [Courses](#)
- [Datasets](#)
- [Education](#)
- [Events \(online\)](#)
- [Jobs](#)
- [Software](#)
- [Webinars](#)



[Future-proof your career, while you work. Global Master of Management Analytics](#)

Topics: [Coronavirus](#) | [AI](#) | [Data Science](#) | [Deep Learning](#) | [Machine Learning](#) | [Python](#) | [R](#) | [Statistics](#)

[KDnuggets Home](#) » [News](#) » [2019](#) » [Sep](#) » [Tutorials, Overviews](#) » An Easy Introduction to Machine Learning Recommender Systems ([19:n34](#))

An Easy Introduction to Machine Learning Recommender Systems

[<= Previous post](#)

[Next post =>](#)

Like 120

Share 120

Tweet

Share

Share

72

Tags: [Beginners](#), [Machine Learning](#), [Python](#), [Recommendation Engine](#), [Recommender Systems](#)

Recommender systems are an important class of machine learning algorithms that offer "relevant" suggestions to users. Categorized as either collaborative filtering or a content-based system, check out how these approaches work along with implementations to follow from example code.



[The future of enterprise data science is here.](#)

[Discover what's new in Domino 4.2](#)



How does YouTube know what videos you'll watch? How does Google always seem to know what news you'll read? They use a [Machine Learning](#) technique called [Recommender Systems](#).

Practically, recommender systems encompass a class of techniques and algorithms which are able to suggest “relevant” items to users. Ideally, the suggested items are as relevant to the user as possible, so that the user can engage with those items: YouTube videos, news articles, online products, and so on.

Items are ranked according to their relevancy, and the most relevant ones are shown to the user. The relevancy is something that the recommender system must determine and is mainly based on historical data. If you've recently watched YouTube videos about elephants, then YouTube is going to start showing you a lot of elephant videos with similar titles and themes!

Recommender systems are generally divided into two main categories: collaborative filtering and content-based systems.

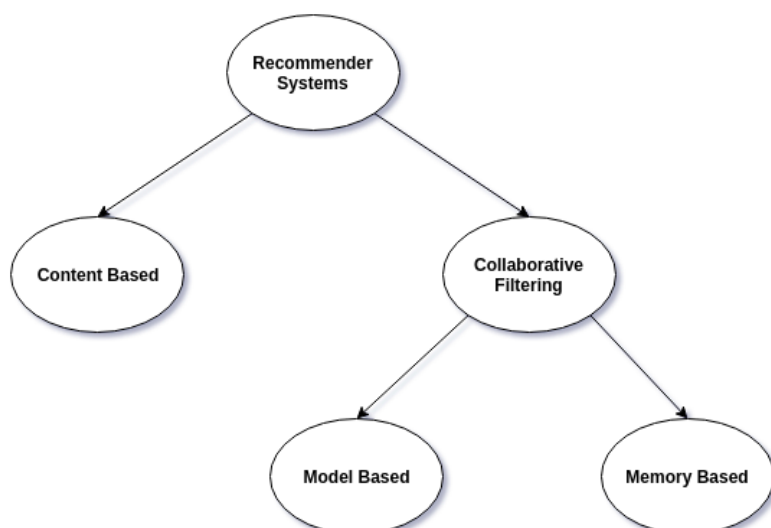


Figure 1: A tree of the different types of Recommender Systems.

Collaborative Filtering Systems

[Collaborative filtering](#) methods for recommender systems are methods that are solely based on the past interactions between users and the target items. Thus, the input to a collaborative filtering system will be all historical data of user interactions with target items. This data is typically stored in a matrix where the rows are the users, and the columns are the items.

The core idea behind such systems is that the historical data of the users should be enough to make a prediction. I.e we don't need anything more than that historical data, no extra push from the user, no presently trending information, etc.



Figure 2: Illustration of how collaborative filtering works for predicting a user's rating of 4 things: an image, a book, a video, and a video game. Based on the users' historical data, the likes and dislikes of each item, the system tries to predict how the user would rate a new item which they haven't rated yet. The predictions themselves are based the past ratings of other users, whose ratings and therefore supposed preferences, are similar to the active user.

In this case, the system made the prediction/recommendation that the active user won't like the video. [Source](#) by [Moshanin](#)

Beyond this, collaborative filtering methods are further divided into two sub-groups: memory-based and model-based methods.

Memory-based methods are the most simplistic as they use no model whatsoever. They assume that predictions can be made on pure "memory" of past data and usually just employ a simple distance-measurement approach, like nearest neighbour.

Model-based approaches, on the other hand, always assume some kind of underlying model and basically try to make sure that whatever predictions come out will fit the model well.

As an example, let's say we have a matrix of users-to-preferred lunch item where all of the users are Americans who love cheeseburgers (they are phenomenal). A memory-based method will only look at what the user has eaten over the past month, without considering that mini-fact of them being cheeseburger loving Americans. A model-based method, on the other hand, will ensure that the predictions always lean a bit more towards being a cheeseburger, since the underlying model assumption is that most people in the dataset should love cheeseburgers!

The Code

We can easily create a collaborative filtering recommender system using [Graph Lab](#)! We'll take the following steps:

1. Load up the data with pandas
2. Convert the pandas dataframes to graph lab SFrames
3. Train the model
4. Make recommendations

```
import graphlab
import pandas as pd

# Load up the data with pandas
r_cols = ['user_id', 'food_item', 'rating']
train_data_df = pd.read_csv('train_data.csv', sep='\t', names=r_cols)
test_data_df = pd.read_csv('test_data.csv', sep='\t', names=r_cols)

# Convert the pandas dataframes to graph lab SFrames
train_data = graphlab.SFrame(train_data_df)
test_data = graphlab.SFrame(test_data_df)



# Train the model
collab_filter_model = graphlab.item_similarity_recommender.create(train_data,
                                                                user_id='user_id',
                                                                item_id='food_item',
                                                                target='rating',
                                                                similarity_type='cosine')

# Make recommendations
which_user_ids = [1, 2, 3, 4]
how_many_recommendations = 5
item_recommendation = collab_filter_model.recommend(users=which_user_ids,
                                                    k=how_many_recommendations)
```

Content-based Systems

In contrast to collaborative filtering, content-based approaches will use additional information about the user and / or items to make predictions.

For example, in the gif we saw above, a content-based system might consider the age, sex, occupation, and other personal user factors when making the predictions. It's much easier to predict that the person wouldn't like the video if we knew it was about skateboarding, but the user's age is 87!

		▶ Fast and Furious	▶ Avatar
Features Male Age 26 Preferences: action, crime	 Mike	✓	✗
Features Female Age 22 Preferences: adventure, fantasy	 Kate	✗	✓

That's why when you sign up for many online websites and services, they ask you to (optionally) give your date of birth, gender, and ethnicity! It's just more data for their system to make better predictions.

Thus, content-based methods are more similar to classical machine learning, in the sense that we will build features based on user and item data and use that to help us make predictions. Our system input is then the **features** of the user and the **features** of the item. Our system output is the prediction of whether or not the user would like or dislike the item.

The Code

We can easily create a collaborative filtering recommender system using [Graph Lab](#)! We'll take the following steps:

1. Load up the data with pandas
2. Convert the pandas dataframes to graph lab SFrames
3. Train the model
4. Make recommendations

```
import graphlab
import pandas as pd

# Load up the data with pandas
r_cols = ['user_id', 'food_item', 'rating']
train_data_df = pd.read_csv('train_data.csv', sep='\t', names=r_cols)
test_data_df = pd.read_csv('test_data.csv', sep='\t', names=r_cols)

# Convert the pandas dataframes to graph lab SFrames
train_data = graphlab.SFrame(train_data_df)
test_data = graphlab.SFrame(test_data_df)

# Train the model
cotent_filter_model = graphlab.item_content_recommender.create(train_data,
                                                                user_id='user_id',
                                                                item_id='food_item',
                                                                target='rating')

# Make recommendations
which_user_ids = [1, 2, 3, 4]
how_many_recommendations = 5
item_recommendation = cotent_filter_model.recommend(users=which_user_ids,
                                                    k=how_many_recommendations)
```

Like to learn?

Follow me on [twitter](#) where I post all about the latest and greatest AI, Technology, and Science! Connect with me on [LinkedIn](#) too!

Related:

- [Building a Recommender System](#)
- [Recommender Engine- Under The Hood](#)
- [Recommendation System Algorithms: An Overview](#)



2. [Must-read NLP and Deep Learning articles for Data Scientists](#)
3. [How to Optimize Your CV for a Data Scientist Career](#)
4. [These Data Science Skills will be your Superpower](#)
5. [Know What Employers are Expecting for a Data Scientist Role in 2020](#)
6. [Working with Spark, Python or SQL on Azure Databricks](#)
7. [4 ways to improve your TensorFlow model – key regularization techniques you need to know](#)

Most Shared

1. [4 ways to improve your TensorFlow model – key regularization techniques you need to know](#)
2. [The NLP Model Forge: Generate Model Code On Demand](#)
3. [DeepMind's Three Pillars for Building Robust Machine Learning Systems](#)
4. [Beyond the Turing Test](#)
5. [Working with Spark, Python or SQL on Azure Databricks](#)
6. [Data Science Tools Illustrated Study Guides](#)
7. [How to Optimize Your CV for a Data Scientist Career](#)

More Recent Stories

- [How to Evaluate the Performance of Your Machine Learning Model](#)
- [10 Things You Didn't Know About Scikit-Learn](#)
- [Top tweets, Aug 26 – Sep 01: A realistic look at the ...](#)
- [What Is Data Enrichment And How It Works](#)
- [Computer Vision Recipes: Best Practices and Examples](#)
- [Which methods should be used for solving linear regression?](#)
- [Here is What I've Learned in 2 Years as a Data Scientist](#)
- [PyCaret 2.1 is here: What's new?](#)
- [Top Online Masters in Analytics, Business Analytics, Data Scie...](#)
- [Showcasing the Benefits of Software Optimizations for AI Workl...](#)
- [eBook: Vocabularies, Text Mining and FAIR Data: The Strategic ...](#)
- [A Curious Theory About the Consciousness Debate in AI](#)
- [Top Stories, Aug 24-30: If I had to start learning Data Scienc...](#)
- [Linguistic Fundamentals for Natural Language Processing: 100 E...](#)
- [Accelerated Computer Vision: A Free Course From Amazon](#)
- [Data is everywhere and it powers everything we do!](#)
- [Beyond the Turing Test](#)
- [Microsoft's DoWhy is a Cool Framework for Causal Inference](#)
- [Explainable and Reproducible Machine Learning Model Developmen...](#)
- [4 ways to improve your TensorFlow model – key regularization techniques you need to know](#)

[KDnuggets Home](#) » [News](#) » [2019](#) » [Sep](#) » [Tutorials, Overviews](#) » An Easy Introduction to Machine Learning Recommender Systems (19:n34)



© 2020 KDnuggets. | [About KDnuggets](#) | [Contact](#) | [Privacy policy](#) | [Terms of Service](#)

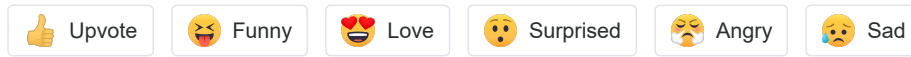
[Subscribe to KDnuggets News](#)



X

What do you think?

46 Responses



2 Comments KDnuggets Disqus' Privacy Policy

Login ▾

Recommend 2
 Tweet
 Share

Sort by Best ▾



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

**Youssef Aidani** • 9 months ago

You could elaborate more. The content is poor and lacking a lot of details.

1 ^ | ▾ • Reply • Share ›

**Raavi Kumar BN** • a year ago

In content based code you still have the same code as in collaborative filtering.

" content-based system might consider the age, sex, occupation, and other personal user

[<= Previous post](#)
[Next post =>](#)

Top Stories Past 30 Days

Most Popular

1. [Know What Employers are Expecting for a Data Scientist Role in 2020](#)
2. [If I had to start learning Data Science again, how would I do it?](#)
3. [Netflix's Polynote is a New Open Source Framework to Build Better Data Science Notebooks](#)
4. [Must-read NLP and Deep Learning articles for Data Scientists](#)
5. [These Data Science Skills will be your Superpower](#)
6. [The List of Top 10 Lists in Data Science](#)
7. [I have a joke about ...](#)

Most Shared

1. [If I had to start learning Data Science again, how would I do it?](#)
2. [Know What Employers are Expecting for a Data Scientist Role in 2020](#)
3. [Must-read NLP and Deep Learning articles for Data Scientists](#)
4. [The List of Top 10 Lists in Data Science](#)
5. [Netflix's Polynote is a New Open Source Framework to Build Better Data Science Notebooks](#)
6. [Top Google AI, Machine Learning Tools for Everyone](#)
7. [Metrics to Use to Evaluate Deep Learning Object Detectors](#)

Latest News

- [How To Decide What Data Skills To Learn](#)
- [Data Scientists think data is their #1 problem. Here's...](#)
- [The Most Important Data Science Project](#)
- [Book Chapter: The Art of Statistics: Learning from Data](#)
- [Design of Experiments in Data Science](#)
- [How to Evaluate the Performance of Your Machine Learnin...](#)

Top Stories Last Week

Most Popular

1. [If I had to start learning Data Science again, how would I do it?](#)