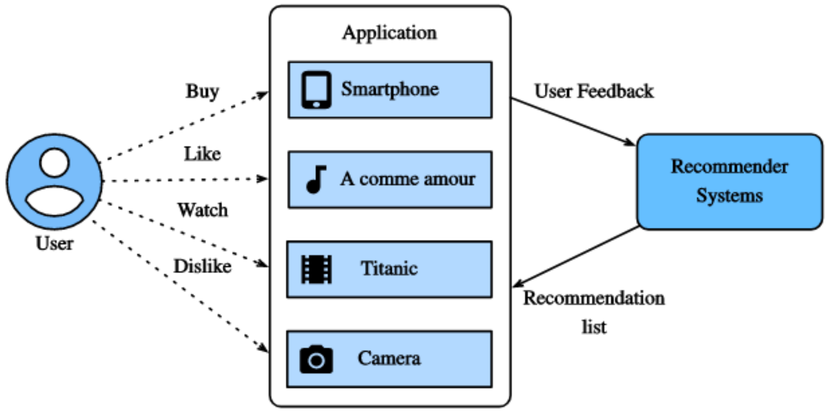
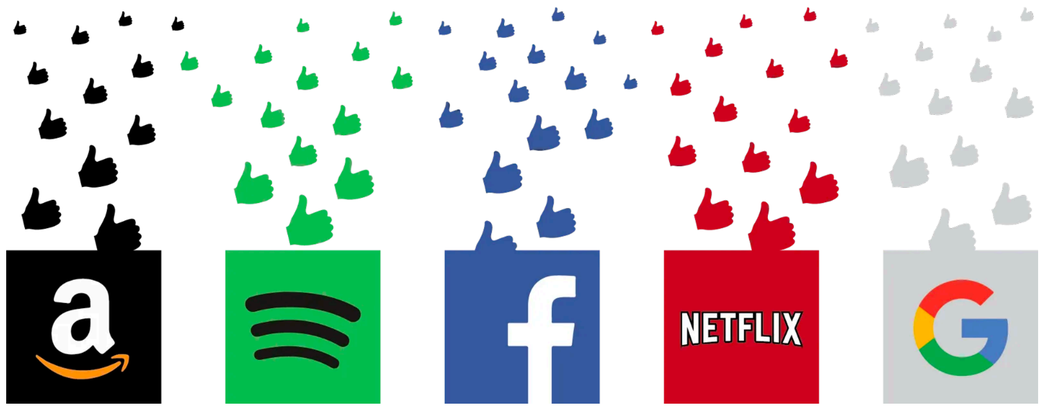
Recommender Systems

Wayne L, Feb, 28

16.1 Overview of Recommender Systems

- Large-scale online services, which profoundly changed the way we communicate, read news, buy products, and watch movies.
- Unprecedented number of items (we use the term item to refer to movies, news, books, and products.) offered online requires a system that can help us discover items that we preferred.
- Recommender systems are therefore powerful information filtering tools that can facilitate personalized services and provide tailored experience to individual users





16.1 Overview of Recommender Systems

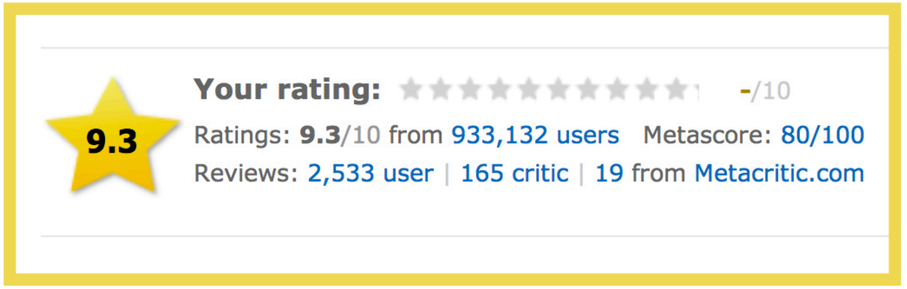
Collaborative Filtering

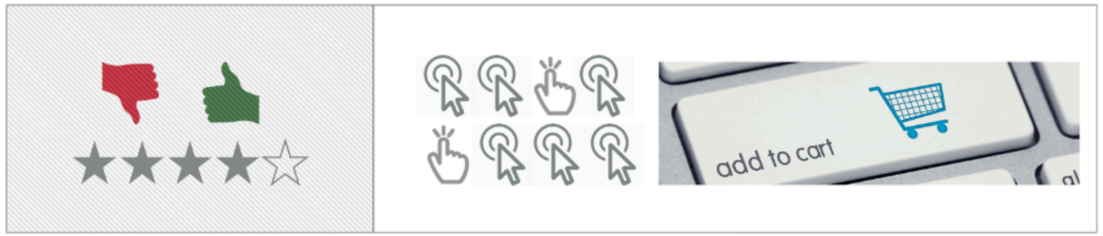
- It is the process of filtering for information or patterns using techniques involving collaboration among multiple users, agents, and data sources.
- Memory-based CF
- Model-based CF —> neural networks
- Content-based (item/users)
- Context-based (timestamp, locations)
- We may need to adjust the model types/structures when different input data is available

16.1 Overview of Recommender Systems

Explicit Feedback and Implicit Feedback

- IMDB collects star rating for movies.YouTube provides the thumb-up and thumbs-down for users to show their preferences. It is apparent that gathering explicit feedback requires users to indicate their interests proactively.
- Nonetheless, explicit feedback is not always readily available as many users my be reluctant to rate products. Relatively speaking, implicit feedback is often readily available since it is mainly concerned with modeling implicit behavior such as user clicks.
- Many systems are centered on implicit feedback which indirectly reflects user's opinion through observing user behavior.
- There are diverse forms of implicit feedback including purchase history, browsing history, watches and even mouse movements.





Explicit Feedback

Implicit Feedback: **Hits**, Time on Site, Page Views, Transaction

16.2 The MovieLens Dataset

Recommendation Tasks

- It is created in 1997 and run by GroupLens, a research lab at the University of Minnesota, in order to gather movie rating data for research purpose.
- It is widely used for recommendation research.
- https://paperswithcode.com/task/recommendation-systems

Results from the Paper

Edit

Ranked #1 on Recommendation Systems on MovieLens 100K (using extra training data)

Get a GitHub badge

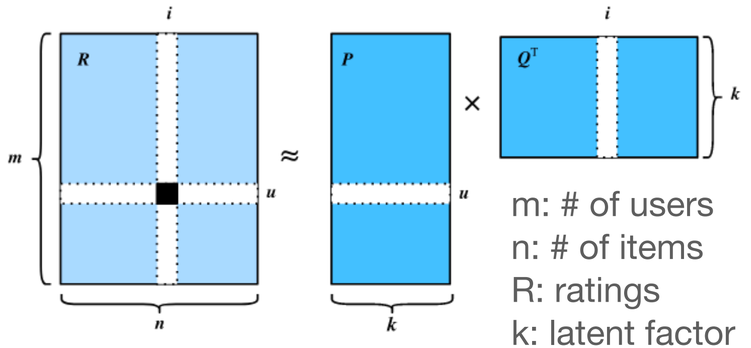
TASK	DATASET	MODEL	METRIC NAME	METRIC VALUE	GLOBAL RANK	USES EXTRA TRAINING DATA	RESULT	BENCHMARK
Recommendation Systems	MovieLens 100K	Bayesian timeSVD++ flipped + Feat	RMSE (u1 Splits)	0.886	# 2	✓		Compare
Recommendation Systems	MovieLens 100K	Bayesian timeSVD++ flipped	RMSE (u1 Splits)	0.886	# 2	✓		Compare
Recommendation Systems	MovieLens 100K	Bayesian timeSVD++ flipped + Feat w/ Ordered Probit Regression and Bayesian timeSVD++ flipped + Feat (ensemble)	RMSE (u1 Splits)	0.883	# 1	✓		Compare
Recommendation Systems	MovieLens 10M	Bayesian timeSVD	RMSE	0.7587	# 4	×		Compare
Recommendation Systems	MovieLens 10M	SGD MF	RMSE	0.772	# 9	×		Compare
Recommendation Systems	MovieLens 10M	Bayesian MF	RMSE	0.7633	# 5	×		Compare

16.3 Matrix Factorization

- It won a one million USD prize at Netfilx contest
- MF is a class of CF models
- Specifically, the model factorizes the user-item interaction matrix into the product of two lower-rank matrices, capturing the low-rank structure of the user-item interactions.
- Predicted ratings can be estimated by $\hat{\mathbf{R}} = \mathbf{P}\mathbf{Q}^\top$
- Objective function

$$\underset{\mathbf{P},\mathbf{Q},b}{\operatorname{argmin}} \sum_{(u,i)\in\mathcal{K}} \|\mathbf{R}_{ui} - \hat{\mathbf{R}}_{ui}\|^2 + \lambda(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2 + b_u^2 + b_i^2)$$

	i_1	i_2	...	i_k	...	i_n
U_1	5	?	...	3	...	4
U_2	?	?	...	4	...	5
...	...	...	...	...	...	...
U_k	2	5	...	?	...	3
...	...	...	...	...	...	...
U_m	5	4	...	2	...	?



m: # of users
n: # of items
R: ratings
k: latent factor

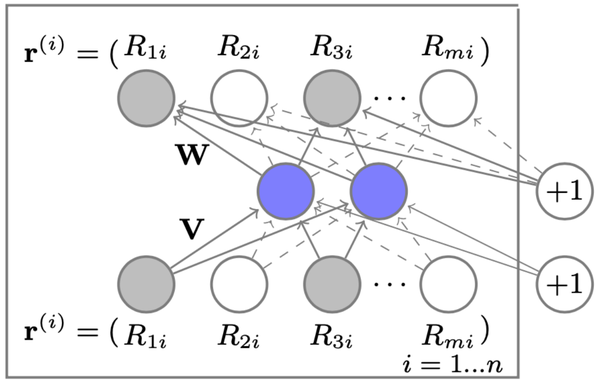
```
class MF(nn.Block):
    def __init__(self, num_factors, num_users, num_items, **kwargs):
        super(MF, self).__init__(**kwargs)
        self.P = nn.Embedding(input_dim=num_users, output_dim=num_factors)
        self.Q = nn.Embedding(input_dim=num_items, output_dim=num_factors)
        self.user_bias = nn.Embedding(num_users, 1)
        self.item_bias = nn.Embedding(num_items, 1)

    def forward(self, user_id, item_id):
        P_u = self.P(user_id)
        Q_i = self.Q(item_id)
        b_u = self.user_bias(user_id)
        b_i = self.item_bias(item_id)
        outputs = (P_u * Q_i).sum(axis=1) + np.squeeze(b_u) + np.squeeze(b_i)
        return outputs.flatten()
```

16.4 AutoRec: Rating Prediction with Autoencoders

WWW 2015

- Although the MF model achieves decent performance on the rating prediction task, it is essentially a linear model.
- Thus, such models are not capable of capturing complex nonlinear and intricate relationships that may be predictive of user preferences.
- $h(\mathbf{R}_{*i}) = f(\mathbf{W} \cdot g(\mathbf{V}\mathbf{R}_{*i} + \mu) + b)$
- activation function: $f(\cdot), g(\cdot)$
  weight matrices: $\mathbf{W}, \mathbf{V}$
  whole network: $h(\cdot)$
- Objective function

$$\underset{\mathbf{W},\mathbf{V},\mu,b}{\operatorname{argmin}} \sum_{i=1}^{M} \| \mathbf{R}_{*i} - h(\mathbf{R}_{*i}) \|_{\mathcal{O}}^2 + \lambda(\|\mathbf{W}\|_F^2 + \|\mathbf{V}\|_F^2)$$




```python
class AutoRec(nn.Block):
    def __init__(self, num_hidden, num_users, dropout=0.05):
        super(AutoRec, self).__init__()
        self.encoder = nn.Dense(num_hidden, activation='sigmoid',
                                use_bias=True)
        self.decoder = nn.Dense(num_users, use_bias=True)
        self.dropout = nn.Dropout(dropout)

    def forward(self, input):
        hidden = self.dropout(self.encoder(input))
        pred = self.decoder(hidden)
        if autograd.is_training():  # Mask the gradient during training
            return pred * np.sign(input)
        else:
            return pred
```