Recommendation Algorithms
(Amazon Recommendations With Item-to-Item Collaborative Filtering)

By: Daniel Izadnegahdar
Date: 09/18/17
What is a recommendation algorithm?

A way of predicting a user’s “preference”.
Estimating scores (TF-IDF)

- The idea is to fill in the white boxes, even though ratings are not available.
- The goal is to interpolate, or guess these ratings.
- These guesses can be interpolated based on item attributes or other customers.
- Weight are also added, and can be dynamic based on changing trends.
- Tables like these are often used in other business sectors, for making difficult decisions given a number of items.

<table>
<thead>
<tr>
<th></th>
<th>Avatar</th>
<th>Star Wars</th>
<th>Pirates</th>
<th>Rango</th>
</tr>
</thead>
<tbody>
<tr>
<td>David</td>
<td>?</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kim</td>
<td></td>
<td>3</td>
<td>5</td>
<td>?</td>
</tr>
<tr>
<td>Charlie</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
## Estimating scores (TF-IDF)

<table>
<thead>
<tr>
<th>company</th>
<th>interest</th>
<th>glassdoor rating</th>
<th>pay</th>
<th>work-life</th>
<th>stability</th>
<th>location</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>weights</td>
<td>18%</td>
<td>20%</td>
<td>25%</td>
<td>18%</td>
<td>10%</td>
<td>10%</td>
<td>N/A</td>
</tr>
<tr>
<td>company01</td>
<td>4</td>
<td>3.2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5.00</td>
</tr>
<tr>
<td>company02</td>
<td>5</td>
<td>3.3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3.33</td>
</tr>
<tr>
<td>company05</td>
<td>4</td>
<td>4.1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>2.50</td>
</tr>
<tr>
<td>company08</td>
<td>5</td>
<td>3.4</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3.33</td>
<td>3.98</td>
</tr>
<tr>
<td>company03</td>
<td>3</td>
<td>3.6</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3.96</td>
</tr>
<tr>
<td>company09</td>
<td>2</td>
<td>4.5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>3.93</td>
</tr>
<tr>
<td>company07</td>
<td>4</td>
<td>3.9</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3.91</td>
</tr>
<tr>
<td>company04</td>
<td>4</td>
<td>3.6</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3.86</td>
</tr>
<tr>
<td>company10</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3.85</td>
</tr>
<tr>
<td>company12</td>
<td>2</td>
<td>4.3</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3.81</td>
</tr>
<tr>
<td>company11</td>
<td>3</td>
<td>4.1</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>2.50</td>
</tr>
<tr>
<td>company06</td>
<td>5</td>
<td>2.9</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1.25</td>
</tr>
<tr>
<td>company16</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1.25</td>
</tr>
<tr>
<td>company13</td>
<td>2</td>
<td>3.7</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3.33</td>
</tr>
<tr>
<td>company15</td>
<td>2</td>
<td>3.7</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1.52</td>
</tr>
<tr>
<td>company17</td>
<td>4</td>
<td>3.4</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>1.25</td>
</tr>
<tr>
<td>company14</td>
<td>0</td>
<td>3.1</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5.00</td>
</tr>
</tbody>
</table>

**Sort**
Example Recommendation Applications

- **Shopping:**
  - Amazon.com

- **Movies:**
  - Netflix.com

- **Music:**
  - Pandora.com

- **Online Dating:**
  - Match.com

- **Food:**
  - Yelp.com
Recommendations need data

• **Ways to receive data:**
  • **Explicit:**
    • Ask people to enter data (problem is that not everyone rates)
  • **Implicit:**
    • Estimate rating from other people (problem is that not everyone rates the same)

• **Solution:**
  • Combine both implicit and explicit data when determining a recommendation.
Types of recommendations

- **Editorial / Hand-curated Recommendation:**
  - Recommendations based on experts instead of customers (not customer input).
  - **Food industry example:** Michelin 3-star reviews.

- **Simple Aggregate Recommendation:**
  - Recommendations based on customers instead of experts (up-votes).
  - **Food industry example:** Yelp.com.

- **Algorithm Recommendation:** *(focus of this lecture)*
  - Using recommendation algorithms to provide recommendations.
  - Tailored to individual users.
Advantages of a recommendation algorithm

- **Saves company and customer time:**
  - Recommended listings have higher click-ratios than randomized listings.
  - Customers save time by searching relevant listings.

- **Customers can get a match from an unknown archive:**
  - A user only searches what he/she knows, but a recommendation can assess a user’s preference, and match an item that he/she wouldn’t know about.
The long tail

- Items that are not popular, are not stocked at retail stores, and therefore unknown.

- But a recommendation algorithm can identify and recommend these unknown items.
Case Study Example of long tail algorithm advantage

**Study:**
- Let A and B be different books.
- Book A is popular, but B is not.
- Some who bought A also bought B.
- This led to a recommendation, where people who bought A may also like B.
- Overt time, B sold more books than A.

**Conclusion:**
- Without recommendation algorithms, book B may have never become popular.
Recommendation challenges

- **Big Data:**
  - Large amount of product and customer data.

- **Speed:**
  - Producing quality recommendations in less than ½ a second.

- **Quality:**
  - Produce recommendations that truly match the user’s preference.

- **Insufficient Information:**
  - New customers provide limited information to determine a quality recommendation.

- **Fluctuation:**
  - Customer data is volatile.
Existing solutions

- **Recommendation methods based on customers:**
  - Traditional collaborative filtering
  - Cluster models

- **Recommendation methods based on products:**
  - Search-based models
  - Item-to-item collaborative filtering *(preferred method by Amazon)*
Traditional Collaborative Filtering

**Background:**
- One of the most common models.
- **Supervised** machine learning (i.e. pattern is pre-defined).

**How it works:**
- Assesses user’s past purchases and their ratings.
- These numbers are interpolated to predict recommendations.
- Uses vectors to assess the favorability of an item.
- Vector components are positive for good ratings, and negative for poor ratings.
- Also uses vectors to match other people's ratings, and base a recommendation accordingly.

**Disadvantages:**
- Takes a long time to compute.

<table>
<thead>
<tr>
<th></th>
<th>Avatar</th>
<th>Star Wars</th>
<th>Pirates</th>
<th>Rango</th>
</tr>
</thead>
<tbody>
<tr>
<td>David</td>
<td>?</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kim</td>
<td></td>
<td></td>
<td>5</td>
<td>?</td>
</tr>
<tr>
<td>Charlie</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Vectors

- Vectors are a common way to assess similarity in recommendation algorithms.
- Vectors have magnitude and direction. 2 vectors are generated for each item to compare (can be products, customers,...etc).
- The angle between the vectors is calculated with the equation below. The smaller the angle, the closer the similarity.
- Equation:

\[ \text{similarity}(\vec{A}, \vec{B}) = \cos(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \cdot \|\vec{B}\|} \]
Cluster models

- **Background:**
  - To improve the speed of traditional collaborative filters, cluster models were developed.
  - *Un-supervised* machine learning (i.e. pattern is **NOT** pre-defined).

- **How it works:**
  - Assigns groups of customers that are similar to each other.
  - To determine similarity, it uses vectors as well, for finding the strength of relationships.
  - It then takes the ratings of that group holistically to generate recommendations.

- **Issues:**
  - Recommendation quality is poor.
Search-based model

- **Background:**
  - Product-based model that matches item attributes such as author, director, artist,…etc.
  - Example, if a user buys the Godfather DVD, the system will recommend other crime titles.

- **How it works:**
  - Performs offline to determine attribute matches.

- **Issues:**
  - Doesn’t work well for users with lots of purchases and ratings.
  - The model has problems finding a summary recommendation from lots of data.
  - Therefore, the quality of the recommendation goes down for large data.
  - The recommendations are often too general or too narrow.
Search-based model

**Precision vs Accuracy**

- ![Precision](image1)
- ![Precision](image2)
- ![Precision](image3)
- ![Precision](image4)
Summary of problems with the existing models

- **Traditional collaborative filtering:**
  - Doesn’t scale well with large data sets because there is no sampling/partitioning.
  - Very slow, takes a lot of computing power:
  - **Big O-Analysis:** $O(MN)$, where $M$ is # of customers and $N$ is number of items

- **Cluster models:**
  - Are faster, but recommendation quality is poor.

- **Search-based models:**
  - Fail to provide quality recommendations because only using basic keyword, categories, or author indexes.
Item-to-item collaborative filtering

- **Background:**
  - Product-based model, assesses product similarities instead of customer similarities.
  - Was developed by Amazon, because existing recommendation methods couldn’t effectively scale to the data sizes that Amazon has to deal with.
  - Amazon has 29 million customers and 480 million catalog items.

- **How it works:**
  - Assesses the items that customers bought together.
  - Specifically looks at products that were bought together from the same customer. This was found to be more effective than looking at products with similar attributes i.e. manufacturer, size, performance…etc.
Item-to-item collaborative filtering: Pseudo

```python
for each item in product catalog I1
    for each customer C who purchased I1
        for each item I2 purchased by customer C
            record that a customer purchased I1 and I2
    for each item I2
        compute the similarity between I1 and I2
```

- Essentially, the algorithm assesses the products that were bought together by a unique customer.
- It’s important to note that the items must be bought on the same order.
Item-to-item collaborative filtering: **Benefits**

- Recommendations are very fast, almost real-time.
- Produces high-quality recommendations.
- Scales to massive data sets.
Conclusion

- Recommendation algorithms effectively improve the customer experience.

- There are customer-based recommendations:
  - Traditional collaborative Filtering
  - Cluster models

- And, product-based recommendations:
  - Search-based models
  - Item-to-item collaborative filtering

- Of these, Amazon recommends item-to-item collaborative filtering, because it scales to large data sizes, its fast, and adapts quickly to changes.
Looking ahead...

• Expand recommendation algorithms to other business sectors such as:
  • Postal service
  • Coupons
  • Teaching
  • Manufacturing
  • Smarter search engines
  • Assistants (Microsoft Clippit)

• Improve the quality of recommendations.
References

- https://www.coursera.org/specializations/recommender-systems
End