Problem Statement

Select a multi-set (set with number) of products subject to certain constraints, that maximize profit.

Essence of Selling

- What products do I stock in my stores?
  - Constraint: capital tied up in keeping products in stores (inventory)
- What products do I keep in my end-caps (checkout counters)?
  - Constraint: shelf-space
- What paid-listings do I show first in a search?
  - Constraint: online real-estate
- For a given customer, what's the best product to advertise?
  - Constraint: online real-estate

Two Scenarios

- Focus on aggregate customer behavior
  - Problem definition
    - E.g. what products do I stock in my stores?
    - No information available about individual customers
- Focus on individual customer
  - Personalization

General Framework

$X$: Person; $P_j$: Product; $E[X, P_j]$: Expected number of $P_j$ that $X$ buys (clicks through, etc...); $M_j$: Profit-Margin on $P_j$.

Aggregate User Case

Collapse all the $X$'s to one node

$\Sigma D_j = \Sigma E[X, P_j]$
Problem Statement

Profit: \( s_j = k_j M_j \)

Maximize: \( \sum_j k_j M_j \)

Turns: \( k_j = 0, 1, 2, \ldots \) (number of \( P_j \) selected)

Subject to: \( \sum_j k_j c_j \leq C \)

\( c_j \) = cost associated with \( P_j \)

\( k_j \leq D_j \) not to exceed demand

Example

<table>
<thead>
<tr>
<th></th>
<th>Margin</th>
<th>Demand</th>
<th>Cost</th>
<th>Margin/Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>3</td>
<td>12</td>
<td>25</td>
<td>12%</td>
</tr>
<tr>
<td>P2</td>
<td>9</td>
<td>3</td>
<td>40</td>
<td>22.5%</td>
</tr>
<tr>
<td>P3</td>
<td>10</td>
<td>1</td>
<td>55</td>
<td>18.2%</td>
</tr>
</tbody>
</table>

Constraint: total cost \( \leq 100 \) \( (C) \)

Greedy (pick maximal margin/cost at each step): \( /P_j^*\)

LP: \( /P_2, P_3/ \)

Retailers and LP

◆ In general product selection can be set up as a linear/integer program (LP)

◆ Retailers are giant multi-stage LP execution engines!

In real life...

◆ Space of products may be too large
  - Eg. Wal-mart has millions of products to consider

◆ All information may not be available

◆ Implementation complexity and Performance impact
  - Problems too large to run in real-time

◆ Intractability

◆ Buyers do the job of product selection
  - More in line with greedy algorithm

Product Selection in Retailers

◆ If all retailers solve the same equations, why don’t they all have the same products?

◆ Product Selection defines Retailer (brand)
  - Brand constraint: maximize profits in the future
    - Eg. Wal-mart brand constraint: select only products that will be bought by 80% of population
  - Eg. Gucci brand constraint: select only high-value (margin) products

Example

<table>
<thead>
<tr>
<th></th>
<th>Margin</th>
<th>Demand</th>
<th>Cost</th>
<th>Margin/Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>3</td>
<td>12</td>
<td>25</td>
<td>12%</td>
</tr>
<tr>
<td>P2</td>
<td>9</td>
<td>3</td>
<td>40</td>
<td>22.5%</td>
</tr>
<tr>
<td>P3</td>
<td>10</td>
<td>1</td>
<td>55</td>
<td>18.2%</td>
</tr>
</tbody>
</table>

Constraint: total cost \( \leq 100 \) \( (C) \)

Wal-mart brand constraint: maximize turns \( /P_j^*\)

Gucci brand constraint: no low-margin products \( /P_3, P_2/ \)
Classifying Retailers

Turns

Wal-mart

Costco

Newco

JCPenney’s

Efficient frontier

Margin

Online Search

- Overture
- Amazon
- Google
Personalization

◆ Given customer □ what products do I recommend to her?
  • □ is a loyal customer □ purchase history available
  • Collaborative-Filtering based Recommender Systems
  • □ is a new customer □ has done certain operations on the sites like search, view products, etc...
  • Assortment of techniques
  • □ is a new customer □ know nothing about her
  • Mass merchandising as in offline retailers, bestsellers,...
◆ In practice, combination of all of the above

Personalization

◆ Offline retail: merchandisers (analog of buyers) pick products to advertise
  • One size fits all □ no personalization
◆ Millions of customers, cannot have human merchandizing to each customer
◆ Algorithms that look at only customer’s data do not work well
◆ **Heuristic: customers help each other**
  • Algorithms enable this to happen!

Recommender Systems

Purchase History of X, available
What new products to advertise to X?

Given set of products that X has bought B = { P1, P2, ..., Pn }

Find P for such that E(Xi|P) is maximum

Collaborative Filtering

◆ Representation of Customer and Product data
◆ Neighborhood formation (find my friends)
◆ Recommendation Generation from neighborhood
Representation

◆ M*N customer product matrix, R
  • r_{ij} = 1 if X_i has bought P_j, 0 otherwise
◆ Issues:
  • Sparsity
    • Mostly 0’s. E.g. Amazon.com 2 million books, less than 0.1% is 1
  • Scalability
  • Very large data sets
  • Authority
    • Take into account similarity between products
      - E.g. paperback “Cold Mountain” is same as hard cover “Cold Mountain”

Finding Neighbors

◆ Similar to clustering
  • cluster around a given customer
◆ First compute similarity between customers: X_a \cdot X_b
  • X_a\sim -- corresponding product vector
  • Cosine measure
    • Cosine of angle between vectors gives similarity
      \text{Sim}(X_a, X_b) = (X_a\sim \cdot X_b\sim) / (|X_a\sim| \cdot |X_b\sim|)
    • See class on Clustering for examples, more info

Neighbors

◆ Pearson Correlation
  • How “proportional” are the vectors
    • Is there a linear relationship between them?
◆ Good indicator of both strength and direction of similarity (correlation)
  • +1: strongly, positively correlated
  • 0: no correlation
  • -1: strongly, negatively correlated

Example

X_a\sim = (1 2 3)
X_b\sim = (2 5 6)

Pearson correlation measures how close to a line 
(1,2) (2,5) (3,6) are
\text{Sim}(X_a, X_b) = \frac{(X_a\sim \cdot X_b\sim - (\sum X_a\sim \cdot \sum X_b\sim) / N)}{\sqrt{(\sum X_a\sim^2 - (\sum X_a\sim^2 / N) \cdot (\sum X_b\sim^2 - (\sum X_b\sim^2 / N)))}}

= 0.9608 (strongly positively correlated)

Neighborhood

◆ Now compute neighborhood of X_a
  • Center-based
    • Select k closest neighbors to X_a
  • Centroid-based
    • Assume j closest neighbors selected
    • Select j+1^\text{st} neighbor by picking customer closest to centroid of first j neighbors
    • Repeat 1..k

Generating Recommendations

◆ From the neighborhood among products X_a has not bought yet, pick:
  • most frequently occurring
  • Weighted Average based on similarity
  • Based on Association Rules

◆ See Sarwar et al (sections 1-3)
  (http://www-users.cs.umn.edu/~karypis/publications/Papers/PDF/ec00.pdf)
Example

<table>
<thead>
<tr>
<th></th>
<th>Shrek</th>
<th>Star Wars</th>
<th>MB</th>
<th>Harry Potter</th>
<th>X-Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Pete</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jeff</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ellen</td>
<td>?</td>
<td>1</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

What new movie should we recommend to Ellen?

Similarity Function

<table>
<thead>
<tr>
<th></th>
<th>Shrek</th>
<th>Star Wars</th>
<th>MB</th>
<th>Harry Potter</th>
<th>X-Files</th>
<th>Similarity to Ellen</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1/\sqrt{6} = 0.41</td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1/\sqrt{2} = 0.71</td>
</tr>
<tr>
<td>Pete</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>1/2</td>
</tr>
<tr>
<td>Jeff</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1/2</td>
</tr>
<tr>
<td>Ellen</td>
<td>?</td>
<td>1</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td></td>
</tr>
</tbody>
</table>

Use Cosine measure for similarity

Neighbors

<table>
<thead>
<tr>
<th></th>
<th>Shrek</th>
<th>Star Wars</th>
<th>MB</th>
<th>Harry Potter</th>
<th>X-Files</th>
<th>Similarity to Ellen</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0.41</td>
</tr>
<tr>
<td>Jane</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0.71</td>
</tr>
<tr>
<td>Pete</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>Jeff</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>Ellen</td>
<td>?</td>
<td>1</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td></td>
</tr>
</tbody>
</table>

Use Center-based approach and pick 3 closest neighbors

Recommendation

<table>
<thead>
<tr>
<th></th>
<th>Shrek</th>
<th>Star Wars</th>
<th>MB</th>
<th>Harry Potter</th>
<th>X-Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jane</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Pete</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jeff</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ellen</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Recommend Star Wars

Implementation Issues

◆ Serious application
  - Large data sizes: millions of users * millions of products
  - CPU cycles
◆ Scalability key
  - Partition the data set and the processing
◆ Real-time vs Batch
  - Real-time can lead to poor response times
  - Real-time preferable – recommend immediately after a customer purchase!
  - Incremental solution key for real time

◆ Sparsity
  - Use navigation history along with purchase history
  - Poorer data quality but reduces sparsity somewhat
Personalization with Limited Information

- Based only on navigation history and current location of customer
- Crucial to relate products to one another
  - Richer user experience
  - Each link drives potential revenue
  - Links built by human labor, explicit customer information, derived customer information, manufacturer info, etc...
  - Much effort in online retailers spent here

Relating Products

- Product Authority
  - Same as one another. E.g. paperback/h.c.
- By Attributes
  - Same author, star, band, manufacturer,...
- By Usage
  - Accessories
- By Explicit User Grouping
  - Lists on Amazon.com
- By Similar Customers Purchasing
  - Customers who bought A also bought B

Quality

- Quality between products and customers
  - Can use interchangeably in problem formulation
  - Real-life feasibility/value
- E.g. Recommender Systems
  - Use purchase history of customers to recommend new product most similar to other products bought by active customer
  - If you’re Tiny, check out this new Star Wars
  - Use buying history of products to recommend new customer, most similar to other customers that have purchased the active product
  - If you’re on the Star Wars page, check out the home page of this customer from Seattle, WA

Summary

- Product Selection is the essence of retailing
- Personalization is unique to online retailing
  - Every customer can have their own store
- Most successful personalization techniques, get customers to help one another
  - Algorithms, like C2, enable this interaction
- In real life, algorithms are complex monsters due to scaling issues, repeated tweaking, etc...