原文发表在：

Greg Linden, Brent Smith, Jeremy York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76-80, Jan./Feb. 2003, doi:10.1109/MIC.2003.1167344

Amazon.com 的推荐

从商品到商品的协同过滤

Greg Linden, Brent Smith, and Jeremy York •
Amazon.com

推荐算法以其在电子商务网站的用途而著称[1]，它们利用有关一个顾客的兴趣作为输入，来产生一个推荐商品的列表。很多应用仅仅使用顾客购买并明确表示代表其兴趣的商品，但它们也可以利用其他属性，包括已浏览的商品、人口统计特征数据、主题兴趣，以及偏爱的艺术家。

在Amazon.com，我们利用推荐算法，对每位顾客提供在线商店个性化。在顾客兴趣的基础上，商店有了彻底的改观，向一个软件工程师展示编程类标题，向一位新妈妈展示婴儿玩具。点击率和转化率——基于网络和邮件广告的两个重要评估指标——极大地超越了那些未定向内容，比如banner广告和热卖列表。

电子商务推荐算法经常要运行在一个充满挑战的环境里。例如：

- 大型零售商有海量的数据，以千万计的顾客，以及数以百万计的登记在册的不同商品。
- 许多应用要求结果实时返回，在半秒之内，还

Amazon.com Recommendations

Item-to-Item Collaborative Filtering

Greg Linden, Brent Smith, and Jeremy York •
Amazon.com

Recommendation algorithms are best known for their use on e-commerce Web sites,1 where they use input about a customer's interests to generate a list of recommended items. Many applications use only the items that customers purchase and explicitly rate to represent their interests, but they can also use other attributes, including items viewed, demographic data, subject interests, and favorite artists.

At Amazon.com, we use recommendation algorithms to personalize the online store for each customer. The store radically changes based on customer interests, showing programming titles to a software engineer and baby toys to a new mother. The click-through and conversion rates — two important measures of Web-based and email advertising effectiveness — vastly exceed those of untargeted content such as banner advertisements and top-seller lists.

E-commerce recommendation algorithms often operate in a challenging environment. For example:

- A large retailer might have huge amounts of data, tens of millions of customers and millions of distinct catalog items.
- Many applications require the results set to be

要产生高质量的推荐。

- 新顾客很典型，他们的信息很有限，只能以少量购买或产品评级为基础。
- 较老的顾客信息丰沛，以大量的购买和评级为基础。
- 顾客数据不稳定：每一次交互都可提供有价值的顾客数据，算法必须立即对新的信息作出响应。

解决推荐问题有三个通常的途径：传统的协同过滤，聚类模型，以及基于搜索的方法。在此，我们就这些方法与我们的算法——我们称之为商品到商品的协同过滤——进行对比。与传统协同过滤不同，我们算法的在线计算规模，与顾客数量和产品目录中的商品数量无关。我们的算法实时产生推荐，计算适应海量数据集，并生成高质量的推荐。

推荐算法

大多数推荐算法，都始于先找出一个顾客集合，他们买过和评级过的商品，与当前用户买过和评级过的商品有重叠[2]。算法把来自这些相似顾客的商品聚集起来，排除该用户已经购买过或评级过的商品，并向该用户推荐其余的商品。这些算法有两个最常见的版本：*协同过滤* 和*聚类模型*。其他算法——包括基于搜索的方法以及我们自己的商品到商品协同过滤——都集中于寻找相似的商品，而不是相似的顾客。针对用户所购买和评级的每一件商品，算法试图找到相似的产品，然后聚集这些相似的商品，并给予推荐。

传统的协同过滤

传统的协同过滤算法把顾客描绘成商品的N维向量，其中N是登记在册的不同商品的数量。对于买过或正面评级的商品，向量分量为正，对于负面评级的商品，向量分量为负。为弥补最热卖的商品，算法典型地会把向量分量乘以反向频率（已

returned in realtime, in no more than half a second, while still producing high-quality recommendations.

- New customers typically have extremely limited information, based on only a few purchases or product ratings.
- Older customers can have a glut of information, based on thousands of purchases and ratings.
- Customer data is volatile: Each interaction provides valuable customer data, and the algorithm must respond immediately to new information.

There are three common approaches to solving the recommendation problem: traditional collaborative filtering, cluster models, and search-based methods. Here, we compare these methods with our algorithm, which we call *item-to-item collaborative filtering*. Unlike traditional collaborative filtering, our algorithm's online computation scales independently of the number of customers and number of items in the product catalog. Our algorithm produces recommendations in realtime, scales to massive data sets, and generates highquality recommendations.

Recommendation Algorithms

Most recommendation algorithms start by finding a set of customers whose purchased and rated items overlap the user's purchased and rated items.[2] The algorithm aggregates items from these similar customers, eliminates items the user has already purchased or rated, and recommends the remaining items to the user. Two popular versions of these algorithms are *collaborative filtering* and *cluster models*. Other algorithms — including search-based methods and our own item-to-item collaborative filtering — focus on finding similar items, not similar customers. For each of the user's purchased and rated items, the algorithm attempts to find similar items. It then aggregates the similar items and recommends them.

Traditional Collaborative Filtering

A traditional collaborative filtering algorithm represents a customer as an *N*-dimensional vector of items, where *N* is the number of distinct catalog items. The components of the vector are positive for purchased or positively rated items and negative for negatively rated items. To compensate for best-selling items, the

购买或评级该商品的顾客数量的倒数），以使不太知名的商品更加相关[3]。对几乎所有的顾客来说，这个向量非常稀疏。

在与该用户最相似的少数顾客基础上，算法产生推荐。算法能够测量两个顾客的相似性，如A和B，有多种方式；一种常见的方法是测量这两个向量之间的夹角余弦值[4]：

$$similarity(\vec{A},\vec{B}) = \cos(\vec{A},\vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}.$$

算法也能从相似顾客的商品里选择推荐，有多种可以利用的方法，常见的一种技术是，按照购买该商品的相似顾客数量，对每件商品进行排序。

利用协同过滤来产生推荐，很耗计算。最坏的情况是O(MN)，其中M是顾客数量，N是产品目录中商品的数量，因为算法要验算M个顾客，并且对每个顾客最多要计算N种商品。但是，由于顾客向量的平均值很稀疏，算法的执行更倾向于接近O(M + N)。扫描每一个顾客大约是O(M)，而不是O(MN)，因为几乎所有顾客向量都只含有很少的商品，无需考虑产品目录的规模。但有少数顾客，他们买过或评级过的商品在产品目录中占有值得注意的百分比，需要O(N)处理时间。因此，算法最终执行的大约是O(M + N)。尽管如此，对非常大的数据集来说——比如1千万以上的顾客，以及1百万以上登记在册的商品——算法也会遭受严峻的性能和计算量问题。

通过减小数据量，可能部分缓解这些计算量的问题[4]。我们能够减小M，通过对顾客进行随机抽样，或丢弃那些购买很少的顾客；我们也能减小N，通过丢弃那些极热门和极冷门的商品。我们还可能减少所需计算的商品数量，通过一个小的常数因子，在产品类别或主题分类的基础上，对商品空间进行区隔。诸如聚类和主分量分析等维度降低

algorithm typically multiplies the vector components by the inverse frequency (the inverse of the number of customers who have purchased or rated the item), making less well-known items much more relevant.[3] For almost all customers, this vector is extremely sparse.

The algorithm generates recommendations based on a few customers who are most similar to the user. It can measure the similarity of two customers, A and B, in various ways; a common method is to measure the cosine of the angle between the two vectors: [4]

$$similarity(\vec{A},\vec{B}) = \cos(\vec{A},\vec{B}) = \frac{\vec{A} \bullet \vec{B}}{\|\vec{A}\| * \|\vec{B}\|}.$$

The algorithm can select recommendations from the similar customers’ items using various methods as well, a common technique is to rank each item according to how many similar customers purchased it.

Using collaborative filtering to generate recommendations is computationally expensive. It is O(MN) in the worst case, where M is the number of customers and N is the number of product catalog items, since it examines M customers and up to N items for each customer. However, because the average customer vector is extremely sparse, the algorithm’s performance tends to be closer to O(M + N). Scanning every customer is approximately O(M), not O(MN), because almost all customer vectors contain a small number of items, regardless of the size of the catalog. But there are a few customers who have purchased or rated a significant percentage of the catalog, requiring O(N) processing time. Thus, the final performance of the algorithm is approximately O(M + N). Even so, for very large data sets — such as 10 million or more customers and 1 million or more catalog items — the algorithm encounters severe performance and scaling issues.

It is possible to partially address these scaling issues by reducing the data size.4 We can reduce M by randomly sampling the customers or discarding customers with few purchases, and reduce N by discarding very popular or unpopular items. It is also possible to reduce the number of items examined by a small, constant factor by partitioning the item space based on product category or subject classification. Dimensionality

技术，也能很大程度减小M和N。

不幸的是，所有这些方法也会以各种形式降低推荐的品质。首先，如果算法只是验算了一小部分顾客样本，那么被选定顾客与当前用户会较少相似。其次，商品空间区隔会把推荐限制在特定产品或主题领域之内。第三，如果算法丢弃了最热门或最冷门的商品，这些商品将不会出现在推荐中，并且，只购买过这些商品的顾客，将不会得到推荐。向商品空间应用维度降低技术，会得到与排除冷门商品那样相同的效果。向顾客空间应用维度降低技术，能有效地把相似顾客组合为群组，正如我们现在所说的，这样的聚类也会降低推荐的品质。

聚类模型

为了寻找与当前用户相似的顾客，聚类模型对顾客基础进行细分，并把这个任务当做为分类问题。算法的目标是，把该用户分配到含有最相似顾客的细分人群里，然后，算法再利用该细分顾客人群的购买和评级，来生成推荐。典型地说，顾客细分的建立，会采用一种聚类或其他无人管理的学习算法，尽管某些应用也用了手工决定的人群细分。利用一种相似性度量标准，聚类算法把最相似的顾客，分组聚合起来，形成聚类或细分人群。由于对大型数据集进行最理想的聚类不切合实际，大多数应用都采用了各种形式的greedy聚类生成。典型的情况是，这些算法始于各细分人群的一个初始集，每个初始集通常包含一个随机选定的顾客，然后算法不断重复地把顾客与现有的细分人群进行匹配，一般也会某些规定，以创建新的细分人群或是合并人群[6]。对于非常大的数据集——尤其是维度很高——抽样或维度降低也是必要的。

一旦算法生成了细分人群，就计算当前用户与概要描述每一细分人群的向量的相似性，然后选择

reduction techniques such as clustering and principal component analysis can reduce M or N by a large factor.5

Unfortunately, all these methods also reduce recommendation quality in several ways. First, if the algorithm examines only a small customer sample, the selected customers will be less similar to the user. Second, item-space partitioning restricts recommendations to a specific product or subject area. Third, if the algorithm discards the most popular or unpopular items, they will never appear as recommendations, and customers who have purchased only those items will not get recommendations. Dimensionality reduction techniques applied to the item space tend to have the same effect by eliminating low-frequency items. Dimensionality reduction applied to the customer space effectively groups similar customers into clusters; as we now describe, such clustering can also degrade recommendation quality.

Cluster Models

To find customers who are similar to the user, cluster models divide the customer base into many segments and treat the task as a classification problem. The algorithm’s goal is to assign the user to the segment containing the most similar customers. It then uses the purchases and ratings of the customers in the segment to generate recommendations. The segments typically are created using a clustering or other unsupervised learning algorithm, although some applications use manually determined segments. Using a similarity metric, a clustering algorithm groups the most similar customers together to form clusters or segments. Because optimal clustering over large data sets is impractical, most applications use various forms of greedy cluster generation. These algorithms typically start with an initial set of segments, which often contain one randomly selected customer each. They then repeatedly match customers to the existing segments, usually with some provision for creating new or merging existing segments.6 For very large data sets — especially those with high dimensionality — sampling or dimensionality reduction is also necessary.

Once the algorithm generates the segments, it computes the user’s similarity to vectors that summarize each segment, then chooses the segment

相似性最大的细分人群，并因此而对该用户进行分类。某些算法把用户分类进入多个细分人群，并对每组关系的强度进行描述。

较之协同过滤，聚类模型有更好的在线可扩展性和性能[3]，因为它们把当前用户与可控数量的细分人群进行对比，而不是整个顾客基数。复杂和昂贵的聚类计算会离线运行。然而，推荐品质却是低的[1]。聚类模型把无数的顾客分组进入细分人群，匹配一个用户与一个细分人群，然后，以相似顾客细分人群里的所有顾客，来考虑产生推荐的目的。由于聚类模型发现的相似顾客并不是最相似的顾客，因而产生的推荐较少相关。通过大量精细粒度的细分人群，也可能提高推荐的品质，但那样一来，在线的用户-细分人群分类，就会变得与利用协同过滤来寻找相似顾客几乎一样昂贵。

基于搜索的方法

基于搜索或内容的方法，将推荐问题视为相关商品的搜索[8]。给定该用户已买过和评级过的商品，算法构造一个搜索查询，以寻找其他热卖的商品，通过同一作者、艺术家或导演，或利用相似的关键词或主题。例如，如果一个顾客买了Godfather（教父）的DVD系列，系统就会推荐其他的犯罪剧，Marlon Brando出演的其他剧目，或由Francis Ford Coppola导演的其他电影。

如果该用户只有少数购买或评级，基于搜索的推荐算法在计算量和性能上都不错。然而，对于有数千次购买的用户，要以针对所有商品的查询为基础也不太可行。算法必须使用一个数据的子集或概要，因此降低了推荐的品质。在所有各种情况下，推荐品质相对较差。推荐通常就是要么太宽泛（比如最热卖的剧集DVD），要么太狭窄（比如同一个作者的全部图书）。推荐应该要帮助顾客找到和发现新的、相关的、有趣的商品。同一作者或同一主题领域的热卖商品，没有满足这一目标。

with the strongest similarity and classifies the user accordingly. Some algorithms classify users into multiple segments and describe the strength of each relationship.7

Cluster models have better online scalability and performance than collaborative filtering3 because they compare the user to a controlled number of segments rather than the entire customer base. The complex and expensive clustering computation is run offline. However, recommendation quality is low.1 Cluster models group numerous customers together in a segment, match a user to a segment, and then consider all customers in the segment similar customers for the purpose of making recommendations. Because the similar customers that the cluster models find are not the most similar customers, the recommendations they produce are less relevant. It is possible to improve quality by using numerous finegrained segments, but then online user–segment classification becomes almost as expensive as finding similar customers using collaborative filtering.

Search-Based Methods

Search- or content-based methods treat the recommendations problem as a search for related items.8 Given the user’s purchased and rated items, the algorithm constructs a search query to find other popular items by the same author, artist, or director, or with similar keywords or subjects. If a customer buys the Godfather DVD Collection, for example, the system might recommend other crime drama titles, other titles starring Marlon Brando, or other movies directed by Francis Ford Coppola.

If the user has few purchases or ratings, searchbased recommendation algorithms scale and perform well. For users with thousands of purchases, however, it’s impractical to base a query on all the items. The algorithm must use a subset or summary of the data, reducing quality. In all cases, recommendation quality is relatively poor. The recommendations are often either too general (such as best-selling drama DVD titles) or too narrow (such as all books by the same author). Recommendations should help a customer find and discover new, relevant, and interesting items. Popular items by the same author or in the same subject category fail to achieve this goal.

商品到商品的协同过滤

Amazon.com在很多邮件营销活动，以及在其大多数的网页上，包括流量极大的网站首页，都把推荐作为一种定向营销工具。点击“你的推荐”链接，会把顾客引向一个区域，在那里，顾客可以通过产品线和主题领域，进行推荐的筛选，为被推荐的商品进行评级，为以前的购买进行评级，并查看为什么这些商品被推荐了（见图1，图在最后）。

如图2（图在最后）所示，即我们的购物车推荐，以其购物车中的商品为基础，向顾客给出产品建议。这一特性与超市结账台路线上的冲动购买类商品很类似，但我们的冲动购买类商品定向到每位顾客。

Amazon.com广泛地采用推荐算法，针对每个顾客的兴趣进行网站的个性化。因为现有的推荐算法，与Amazon.com千万级的用户和产品数量不能相称，我们开发了自己的算法。我们的算法，也就是商品到商品的协同过滤，符合海量的数据集和产品量，并能实时得到高品质的推荐。

它如何工作

与把当前用户匹配到相似顾客的做法不同，商品到商品的协同过滤，把该用户所购买和评级的商品，匹配到相似的商品，然后组合这些相似的商品进入推荐列表[9]。

对于给定的一件商品，为了决定最相似的匹配，算法通过发现顾客倾向于一起购买的商品，建立一个相似商品的表格。利用对所有产品配对的迭代，以及为每个产品配对计算相似性测度，我们能建立一个产品到产品的矩阵。然而，许多产品配对没有普通顾客，因此在处理时间和内存使用上，这种方法没有效率。下述迭代算法提供了一种更好的方法，通过计算一件商品与所有相关产品之间的相似性：

Item-to-Item Collaborative Filtering

Amazon.com uses recommendations as a targeted marketing tool in many email campaigns and on most of its Web sites’ pages, including the hightraffic Amazon.com homepage. Clicking on the “Your Recommendations” link leads customers to an area where they can filter their recommendations by product line and subject area, rate the recommended products, rate their previous purchases, and see why items are recommended (see Figure 1).

As Figure 2 shows, our shopping cart recommendations, which offer customers product suggestions based on the items in their shopping cart. The feature is similar to the impulse items in a supermarket checkout line, but our impulse items are targeted to each customer.

Amazon.com extensively uses recommendation algorithms to personalize its Web site to each customer’s interests. Because existing recommendation algorithms cannot scale to Amazon.com’s tens of millions of customers and products, we developed our own. Our algorithm, item-to-item collaborative filtering, scales to massive data sets and produces high-quality recommendations in real time.

How It Works

Rather than matching the user to similar customers, item-to-item collaborative filtering matches each of the user’s purchased and rated items to similar items, then combines those similar items into a recommendation list.9

To determine the most-similar match for a given item, the algorithm builds a similar-items table by finding items that customers tend to purchase together. We could build a product-to-product matrix by iterating through all item pairs and computing a similarity metric for each pair. However, many product pairs have no common customers, and thus the approach is inefficient in terms of processing time and memory usage. The following iterative algorithm provides a better approach by calculating the similarity between a single product and all related products:

```
For 每件商品 in 产品目录, $I_1$
  For 每位顾客 $C$ 购买过 $I_1$ 的
      For 每件商品 $I_2$ 由顾客 $C$ 所购买的
        记录一顾客所购买的 $I_1$ 和 $I_2$
  For 每件商品 $I_2$
      计算相似度 在 $I_1$ 与 $I_2$ 之间
```

计算两个商品之间的相似性可以有多种方法，但通常的方法是，利用我们前面描述的余弦值，其中每个向量对应于一件商品而不是一位顾客，并且向量的M维度对应于已购买过该商品的顾客。

这个相似商品表格的离线计算极费时间，最糟糕时需要O(N^{2}M)。但在实际运行中，它接近O(NM)，因为大多数顾客只有很少的购买。对购买最热门商品顾客的抽样，进一步减少了运行时间，同时对推荐的品质略有降低。

对于给定的相似商品表格，算法发现与当前用户每次购买和评级相似的商品，把这些商品聚集起来，然后推荐最畅销或关联最强的商品。这一计算很快速，仅仅取决于该用户购买或评级过商品的数量。

可扩展性：比较

Amazon.com有超过2900万顾客，以及数百万登记在册的商品。其他主要零售商也有同等大小的数据源。在所有这些数据提供机会的同时，也是一种诅咒，远远突破了那些针对小三个数量级的数据集所设计的算法的限度。几乎全部的现有算法，都是在小数据集上评估的。如，MovieLens数据集[4]包含35000名顾客和3000件商品，EachMovie数据集[3]包含4000名顾客和1600件商品。

对于非常大的数据集，一个可扩展的推荐算法必须离线运行最昂贵的计算。正如下面的简要对比所显示的，现有方法达不到这样的要求：

- 传统的协同过滤只做很少或不做离线计算，其

```
For each item in product catalog, $I_1$
  For each customer $C$ who purchased $I_1$
      For each item $I_2$ purchased by
         customer $C$
        Record that a customer purchased $I_1$
         and $I_2$
  For each item $I_2$
      Compute the similarity between $I_1$ and $I_2$
```

It’s possible to compute the similarity between two items in various ways, but a common method is to use the cosine measure we described earlier, in which each vector corresponds to an item rather than a customer, and the vector’s Mdimensions correspond to customers who have purchased that item.

This offline computation of the similar-items table is extremely time intensive, with O(N^{2}M) as worst case. In practice, however, it’s closer to O(NM), as most customers have very few purchases. Sampling customers who purchase best-selling titles reduces runtime even further, with little reduction in quality.

Given a similar-items table, the algorithm finds items similar to each of the user’s purchases and ratings, aggregates those items, and then recommends the most popular or correlated items. This computation is very quick, depending only on the number of items the user purchased or rated.

Scalability: A Comparison

Amazon.com has more than 29 million customers and several million catalog items. Other major retailers have comparably large data sources. While all this data offers opportunity, it’s also a curse, breaking the backs of algorithms designed for data sets three orders of magnitude smaller. Almost all existing algorithms were evaluated over small data sets. For example, the MovieLens data set4 contains 35,000 customers and 3,000 items, and the EachMovie data set3 contains 4,000 customers and 1,600 items.

For very large data sets, a scalable recommendation algorithm must perform the most expensive calculations offline. As a brief comparison shows, existing methods fall short:

- Traditional collaborative filtering does little or no offline computation, and its online computation scales

在线计算量取决于顾客和登记在册商品的数量。在大数据集的情况下，这样的算法不可行，除非使用维度降低、抽样或区隔——所有这些都降低了推荐的品质。

- 聚类模型能离线运行大量的计算，但推荐品质相对较差。出于改进，可以增加人群细分的数量，但这会使在线的用户-细分人群的分类变得昂贵。
- 基于搜索的模型离线建立起关键词、范畴、作者索引，但不能提供符合兴趣、定向内容的推荐。对于购买和评级很多的顾客来说，这些算法的扩展性不佳。

商品到商品协同过滤的可扩展性和性能的关键是，它离线建立耗时巨大的相似商品表格。该算法的在线部分——针对当前用户的购买和评级来寻找相似的商品——计算量独立于商品目录的规模或顾客的总数；仅仅取决于该用户买过或评级过多少个商品。因此，甚至是对于超大数据集，算法也很快速。由于该算法能推荐高度关联的相似商品，推荐的品质就很出色[10]。与传统的协同过滤不同，该算法在用户数据有限的情况下也能运行良好，在少至2到3件商品的基础上，产生高品质的推荐。

结论

通过为每位顾客建立个性化的购物体验，推荐算法提供了一种有效的定向营销形式。对于Amazon.com这样的大型零售商，良好的推荐算法可在海量顾客基数和商品目录上进行扩展，只需要子秒处理时间就能产生在线推荐，能对用户数据里的变化立即做出反应，并能为所有用户提供引人关注的推荐，而无需考虑购买和评级的数量。与其他算法不同，商品到商品的协同过滤能满足这样的挑战。

未来，我们期望零售业为定向营销更广泛地应用推荐算法，包括网上和网下。对个性化来说，电子商务拥有最方便的工具，而同时，较之传统撒

with the number of customers and catalog items. The algorithm is impractical on large data sets, unless it uses dimensionality reduction, sampling, or partitioning — all of which reduce recommendation quality.

- Cluster models can perform much of the computation offline, but recommendation quality is relatively poor. To improve it, it’s possible to increase the number of segments, but this makes the online user–segment classification expensive.
- Search-based models build keyword, category, and author indexes offline, but fail to provide recommendations with interesting, targeted titles. They also scale poorly for customers with numerous purchases and ratings.

The key to item-to-item collaborative filtering’s scalability and performance is that it creates the expensive similar-items table offline. The algorithm’s online component — looking up similar items for the user’s purchases and ratings — scales independently of the catalog size or the total number of customers; it is dependent only on how many titles the user has purchased or rated. Thus, the algorithm is fast even for extremely large data sets. Because the algorithm recommends highly correlated similar items, recommendation quality is excellent.10 Unlike traditional collaborative filtering, the algorithm also performs well with limited user data, producing high-quality recommendations based on as few as two or three items.

Conclusion

Recommendation algorithms provide an effective form of targeted marketing by creating a personalized shopping experience for each customer. For large retailers like Amazon.com, a good recommendation algorithm is scalable over very large customer bases and product catalogs, requires only subsecond processing time to generate online recommendations, is able to react immediately to changes in a user’s data, and makes compelling recommendations for all users regardless of the number of purchases and ratings. Unlike other algorithms, item-to-item collaborative filtering is able to meet this challenge.

In the future, we expect the retail industry to more broadly apply recommendation algorithms for targeted marketing, both online and offline. While e-commerce

大网的方式，该技术对转化率的提升，也会引起网下零售商的关注，可用于信件、优惠券及其他顾客通信中。

作者

Greg Linden是Amazon.com个性化部门的共同创始人、研究员及高级经理，他设计和开发了推荐算法。他目前是斯坦福大学商业研究生院Sloan Program中的管理学研究生。他的研究兴趣包括推荐系统、个性化、数据挖掘以及人工智能。Linden在华盛顿大学获得了计算机科学的理学硕士。联系方式：Linden_Greg@gsb.stanford.edu。

Brent Smith领导着Amazon.com的自动化销售团队。他的研究兴趣包括数据挖掘、机器学习以及推荐系统。他在San Diego的加州大学获得了数学的学士，在华盛顿大学获得了数学的理学硕士，并在那里做了些微分几何的研究工作。联系方式：smithbr@amazon.com。

Jeremy York领导着Amazon.com的自动化内容选择和交付团队。他的兴趣包括分类数据的统计模型、推荐系统，以及网站展示内容的最佳选择。他从华盛顿大学获得了统计学的博士学位，在那里，他的论文获得了Leonard J. Savage奖的贝叶斯应用计量经济学和统计学的最佳论文奖。联系方式：jeremy@amazon.com。

参考文献

businesses have the easiest vehicles for personalization, the technology's increased conversion rates as compared with traditional broad-scale approaches will also make it compelling to offline retailers for use in postal mailings, coupons, and other forms of customer communication.

Greg Linden was cofounder, researcher, and senior manager in the Amazon.com Personalization Group, where he designed and developed the recommendation algorithm. He is currently a graduate student in management in the Sloan Program at Stanford University's Graduate School of Business. His research interests include recommendation systems, personalization, data mining, and artificial intelligence. Linden received an MS in computer science from the University of Washington. Contact him at Linden_Greg@gsb.stanford.edu.

Brent Smith leads the Automated Merchandising team at Amazon. com. His research interests include data mining, machine learning, and recommendation systems. He received a BS in mathematics from the University of California, San Diego, and an MS in mathematics from the University of Washington, where he did graduate work in differential geometry. Contact him at smithbr@amazon.com.

Jeremy York leads the Automated Content Selection and Delivery team at Amazon.com. His interests include statistical models for categorical data, recommendation systems, and optimal choice of Web site display components. He received a PhD in statistics from the University of Washington, where his thesis won the Leonard J. Savage award for best thesis in applied Bayesian econometrics and statistics. Contact him at jeremy@amazon.com.

References

1. J.B. Schafer, J.A. Konstan, and J. Reidl, "E-Commerce Recommendation Applications," Data Mining and Knowledge Discovery, Kluwer Academic, 2001, pp. 115-153.

2. P. Resnick et al., “GroupLens: An Open Architecture for Collaborative Filtering of Netnews,” Proc. ACM 1994 Conf. Computer Supported Cooperative Work, ACM Press, 1994, pp. 175-186.

3. J. Breese, D. Heckerman, and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” Proc. 14th Conf. Uncertainty in Artificial Intelligence, Morgan Kaufmann, 1998, pp. 43-52.

4. B.M. Sarwarm et al., “Analysis of Recommendation Algorithms for E-Commerce,” ACM Conf. Electronic Commerce, ACM Press, 2000, pp.158-167.

5. K. Goldberg et al., “Eigentaste: A Constant Time Collaborative Filtering Algorithm,” Information Retrieval J., vol. 4, no. 2, July 2001, pp. 133-151.

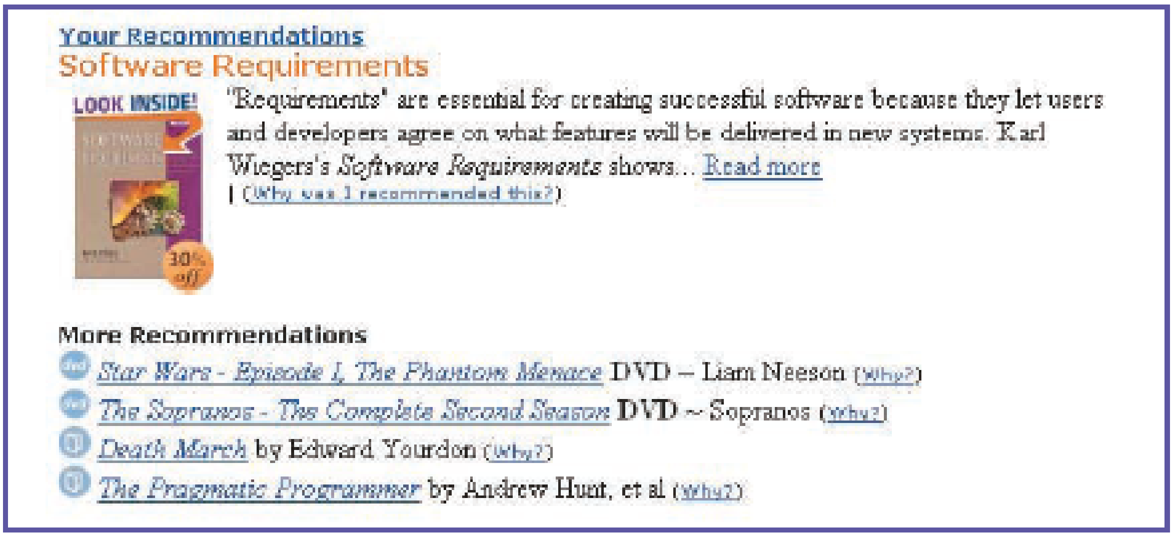
6. P.S. Bradley, U.M. Fayyad, and C. Reina, “Scaling Clustering Algorithms to Large Databases,” Knowledge Discovery and Data Mining, Kluwer Academic, 1998, pp. 9-15.

7. L. Ungar and D. Foster, “Clustering Methods for Collaborative Filtering,” Proc. Workshop on Recommendation Systems, AAAI Press, 1998.

8. M. Balabanovic and Y. Shoham, “Content-Based Collaborative Recommendation,” Comm. ACM, Mar. 1997, pp. 66-72.

9. G.D. Linden, J.A. Jacobi, and E.A. Benson, Collaborative Recommendations Using Item-to-Item Similarity Mappings, US Patent 6,266,649 (to Amazon.com), Patent and Trademark Office, Washington, D.C., 2001.

10. B.M. Sarwar et al., “Item-Based Collaborative Filtering Recommendation Algorithms,” 10th Int’l World Wide Web Conference, ACM Press, 2001, pp. 285-295.



Your Recommendations

Software Requirements

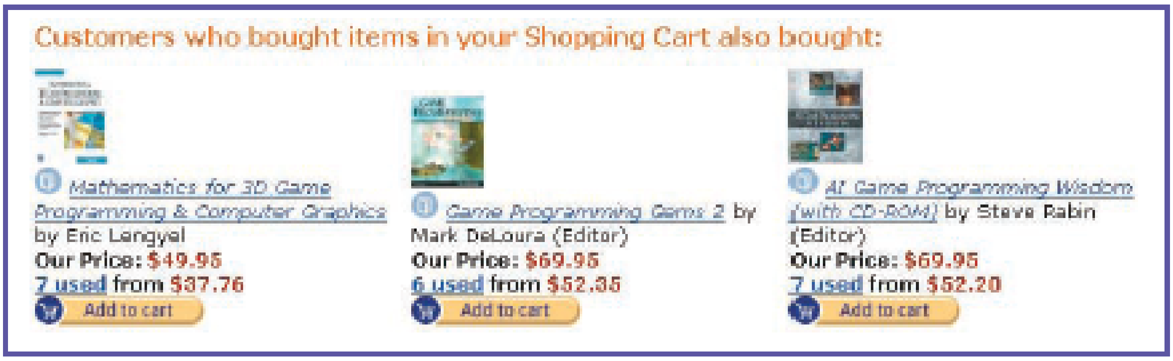
"Requirements" are essential for creating successful software because they let users and developers agree on what features will be delivered in new systems. Karl Wiegers's *Software Requirements* shows... Read more

| (Why was I recommended this?)

More Recommendations

- *Star Wars - Episode I, The Phantom Menace* DVD ~ Liam Neeson (Why?)
- *The Sopranos - The Complete Second Season* DVD ~ Sopranos (Why?)
- *Death March* by Edward Yourdon (Why?)
- *The Pragmatic Programmer* by Andrew Hunt, et al (Why?)

Figure 1. The "Your Recommendations" feature on the Amazon.com homepage. Using this feature, customers can sort recommendations and add their own product ratings.



Customers who bought items in your Shopping Cart also bought:

Mathematics for 3D Game Programming & Computer Graphics by Eric Lengyel
Our Price: $49.95
7 used from $37.76
Add to cart

Game Programming Gems 2 by Mark DeLoura (Editor)
Our Price: $69.95
6 used from $52.35
Add to cart

AI Game Programming Wisdom (with CD-ROM) by Steve Rabin (Editor)
Our Price: $69.95
7 used from $52.20
Add to cart

Figure 2. Amazon.com shopping cart recommendations. The recommendations are based on the items in the customer's cart: The Pragmatic Programmer *and* Physics for Game Developers.

图 1 Amazon.com 主页上“你的推荐”特性。
利用这一特性，顾客能够对推荐进行排序，并加入他们自己的产品评级。

图 2 Amazon.com 的购物车推荐。
该推荐的基础是顾客购物车中的商品：Pragmatic Programmer 和 Physics for Game Developers。