集合知プログラミング
第2章　推薦を行う(2)

06T4073R 三上健太

似ている製品

- 製品同士似ているものを知る
  - Amazonでは類似製品が紹介される

↓

特定のアイテムを好きな人を探し、その人たちが好きな他のアイテムを探す

- マイナスの相関スコアもあり得る
  (Aが好きな人はBは好まない傾向など)

del.icio.usのリンクを推薦するシステム

- オンラインブックマークサイト
  - アカウントを作ればリンクを投稿し、参照が可能
  - 他のユーザが投稿したリンクの閲覧も可能
  - 多くの人々に投稿された"popular"なリンクを知れる

↓ しかし・・・

似たような人を探す、ユーザにリンクを推薦する機能はない → これを実装する

del.icio.usのAPI

- XML形式で返される

- 人々が投稿したリンクを取得するための関数をもつ
  - あるキーワードに関して最新の"popular"な投稿リストを取得する関数
  - 与えられたURLの全ての投稿を返す関数
  - 与えられたユーザの全ての投稿を返す関数 など

データセットを作る

del.icio.usから全ユーザの投稿を取得→無理がある

↓

一部を選出

選び方

- 投稿頻度が高く似たような内容の投稿を繰り返している人

↓

特定のタグでの"popular"なリンクを最近投稿した人々のリストを得る

ご近所さんとリンクの推薦

- データセットを用いればユーザに似た嗜好をもつ別のユーザを検索可能

- ユーザの観点から・・・
  - 指定したユーザが好みそうなリンクを推薦できる

- リンクの観点から・・・
  - ユーザが特に気に入った特定のリンクに似ているリンクを探し出せる

del.icio.usのリンクを推薦するシステム

- del.icio.usに推薦エンジンを付け加えることに成功

- 他にできることは・・・
  - お互い似ているタグを探し出す
    (del.icio.usはタグ検索も可能なので)
  - あるページを"popular"にするために同じリンクを複数アカウントで繰り返し投稿しているユーザの検索

アイテムベースのフィルタリング(1)

今まで作成した推薦エンジン

→ 全ユーザからのランキングが必要

人々やアイテムの数が数千件程度ならうまく動作する

↓しかし・・・

巨大サイトはデータ数が膨大

ユーザ、製品同士の比較に時間がかかる

また・・・

製品数が多いと人々の重なり合う部分が少なく、似ているユーザを決めるのが難しくなる

アイテムベースのフィルタリング(2)

- アイテムベースの協調フィルタリング
  - 対象が巨大なデータセットの場合
  - 計算の大部分は事前に実行可能
    (アイテム同士の関係はあまり頻繁に変わらない)

- あるユーザに推薦する場合
  - そのユーザが高く評価したアイテムを参照．そのアイテムに似ている順に重み付けされたリストを作成

アイテムベースのフィルタリング(3)

- アイテム間の類似度データセットを作成
  - 一度作ったら再利用できる
  - アイテムの類似度が最新に保たれる程度の頻度でOK

- 推薦を行う
  - 特定のユーザが評価した全てのアイテムを取得

↓

アイテム毎に似ているアイテムを探す

↓

類似度、評価を利用して重み付けする

MovieLensのデータセットを使う

- MovieLens
  - GroupLens Project(ミネソタ大学)が作成
  - 実際の映画の評価のデータセットが取得可
  - 1682本の映画に対する943名のユーザの評価

- アイテムベースとユーザベースのパフォーマンスの違いを理解できる
- 映画の他にも書籍、ジョークなどのデータもある

ユーザベース VS アイテムベース

- ユーザベース
  - シンプルで余分なステップを必要としない
  - メモリに収まるサイズで変更が頻繁に行われるデータセットに有効
  - リンク共有、音楽推薦サイトに適している

- アイテムベース
  - 大きなデータセットからたくさんの推薦を得れる
  - アイテム類似度テーブルをメンテナンスする必要性 → オーバーヘッドがある