機械学習やその関連技術の大規模ウェブサービスにおけるレコメンデーションへの応用

ヱヂリウム株式会社
渡邉卓也
中里裕司　室井浩明

日本行動計量学会　グローバルビジネス学会
第一回ビッグデータセミナー
2016年5月23日

発表の枠組み

- 対象者
  - 事前知識のない方から専門家まで
- レコメンデーション一般について
  - 網羅的な知識を提供することは目的としない
  - 必要に応じて要点を解説
- 実際の事例に基づいて
  - どのような課題に対して何を考え、どう対処したかを説明
  - 専門的表記はあまり使用しないで表現
  - 具体的な数字についてはお伝えできないことが多いです
- 質問や指摘
  - 随時お願いします
  - 発表後にも質疑応答時間を設けます
- 本発表資料は弊社ウェブページに掲載予定
  - http://www.edirium.co.jp/

発表の流れ

- 前置き
  - 「ビッグデータ」について
  - 大規模レコメンデーションに関する知識
- レコメンデーションの簡単な導入
- 事例紹介
  - 住宅情報サービス
    - 簡潔なユーザインタフェースによる効率的な推薦対象の決定
  - 動画配信サービス
    - リアルタイムな推薦の最適化や推薦の多様化
  - ショッピングモール
    - 力押し大規模並列計算とリアルタイムな類似商品推薦
- おわりに

「ビッグデータ」について

- どこからが「ビッグ」か
  - 従来のRDBMSでは扱いにくい大きさから？
  - Excelでは扱いにくい大きさから「ビッグ」扱いも
  - 本発表で紹介する事例は商品数やユーザ数が1000万のオーダ
- なぜ「ビッグ」なのか
  - ウェブベースのサービスの発展
    - 通信販売
    - 記事や動画の配信
    - SNS
  - 行動履歴を活用しようとするとデータ点数が大きく増える
    - 一人のユーザが数百のデータ点を産み出す
    - 従来はあまり活用されていなかった
  - データを産み出す機器の増大
    - 伝統的業務のデジタル化: POS機器、自動改札機
    - 新しい領域への進出: IoT機器

大規模レコメンデーションに関する知識1

- 計算機科学の基礎
  - 情報量
  - アルゴリズムと計算量
- 統計
  - 各種確率分布
  - 統計分析手法
- 計算機科学諸分野
  - 情報検索
  - データマイニング
  - 機械学習
  - 自然言語処理
- プログラミング言語
  - 各言語の特性
  - 言語処理系やライブラリ

大規模レコメンデーションに関する知識2

- 計算機システム
  - ソフトウェアプロダクトやサービスの使い方、運用方法
  - 計算機ネットワークの運用方法
- ユーザインタフェース
- ドメイン知識
  - 商品特性
  - 業務フロー
  - 意思決定の構造

レコメンデーションとは

- 商品やユーザの特徴あるいはユーザの行動等の情報に基づいて、ユーザに好まれるであろう商品を推薦すること
  - 個々のユーザに最適化した推薦が望ましいとされる
  - 売り手の売りたい商品ではなく、個々のユーザに売れそうな商品は何かをまず考える点で単純な広告とは異なる
- 活用される場面
  - 本発表では主にウェブサービスにおける商品の推薦を対象とする
  - 最近は店頭接客や電話応対との統合が注目されている
- 頻出キーワード
  - 近傍探索
  - 次元圧縮
  - 協調フィルタリング
  - コールドスタート問題

レコメンデーションの分類

- 商品やユーザの特徴に基づく
  - 商品同士の類似度やユーザの特徴との親和性を利用
  - 手法
    - 類似商品の推薦
    - ユーザの登録した興味のある分野の商品を推薦
- ユーザの行動履歴に基づく
  - 閲覧や購買といった行動に関する情報から好みの商品を推定
  - 手法
    - 協調フィルタリング
    - 売り上げランキング
      - これをレコメンデーションとは呼ばない人も
- それらの複合
  - 商品やユーザの特徴とユーザの行動とを結びつけてユーザの好みを推定することで、より精度を上げることを狙う

商品やユーザの特徴に基づく推薦

- 特性
  - 商品やユーザの特徴を直接的に利用する
  - 特徴間の類似度や近縁性に基づいて推薦する
  - 商品説明文の解析といった前処理を伴うことが多い
  - 安定した結果が得られることが多い反面、つまらない推薦となる可能性も
- 大量の同等な商品がある場合への対処が必要
  - 推薦結果の多様化が必要
  - 色違い等であれば、代表する商品のみ表示するといったビジネスロジックでの対処も有効

ユーザの行動履歴に基づく推薦

- 特性
  - 商品やユーザの特徴は直接的には利用しない
  - 商品の特徴をユーザの目で判断した結果としての、行動履歴を利用する
    - 商品とユーザの特徴との関係は、行動履歴に間接的に現れている
  - ユーザの集合的な行動パターンを利用することもある
  - 意外な商品間の関係性が見つかることがある
    - ビールとおむつ
- コールドスタート問題
  - 行動履歴が集まるまでは有効な推薦ができない
  - 多くの商品やユーザについて望ましい数の履歴が集まらないことがある（ロングテール）

事例1: 住宅情報サービス

- 機械学習による賃貸物件の推薦
  - ユーザが二つの選択肢から好みの方を選ぶことで、どんどん好みの物件に収束していく
- 論文
  - 中野猛, 下垣徹, 橋本拓也, 渡邉卓也, "Jubatusの機能を利用した二者択一型不動産賃貸物件推薦サービスの開発", デジタルプラクティス, vol. 5, no. 2, pp. 130-138, 情報処理学会, 2014.

機械学習

- 機械学習とは
  - 明示的にプログラムせずに、人間が学習するように機械に学習させること
  - 有名な定義（簡略化してあります）
    - 経験によって課題に対する成績を向上させること
  - 応用例
    - 郵便物の宛先自動読み取り
    - 迷惑メール判定
- データマイニングとの違い
  - 手法の面では相当重なる部分が多い
  - 目的が異なる
    - データマイニングは未知の知識を発見することに重点
    - 機械学習は既知の知識を習得し、そこから予測することに重点
  - 機械学習であるかのおおよその判定基準（個人的見解）
    - 人間のやっていることを学習を通じて代替しているかどうか
      - データマイニングは生身の人間には見つけられないものを見つける

オンライン機械学習フレームワークJubatus

- http://jubat.us/
- NTTとPFI（今はPFN名義）が開発
- 提供する機能
  - 分類
  - 回帰
  - 推薦（近傍探索）
  - クラスタリング
  - 外れ値検知
  - バースト検知
  - 多腕バンディット
    - 本発表で紹介する事例で開発したものが元になっている
- 最新版: 0.9.0
  - Pythonからより簡単に利用可能に

分類器

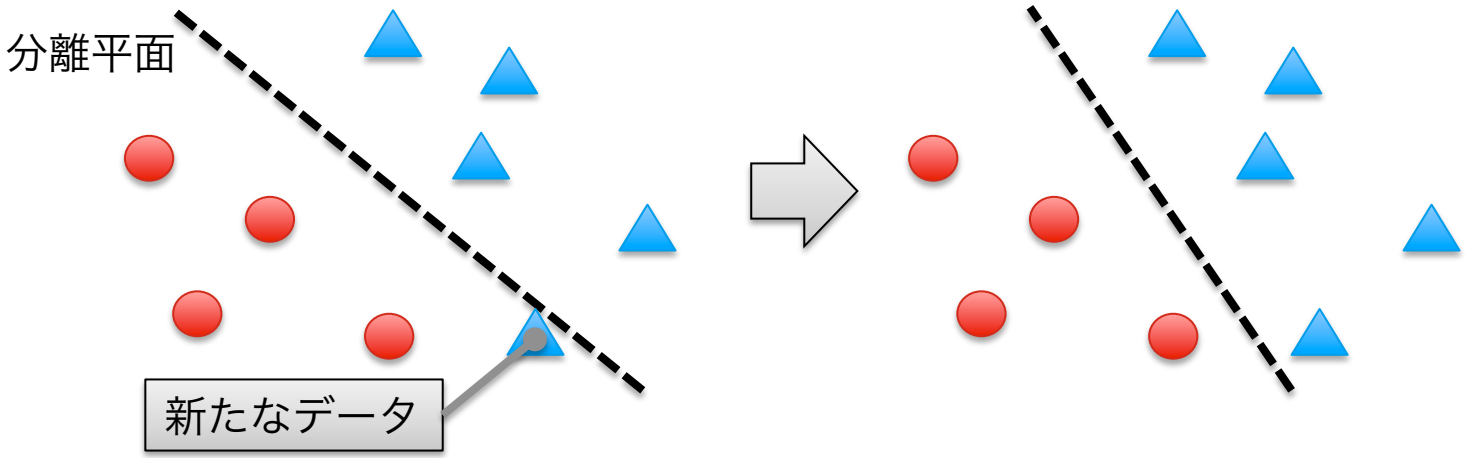
- 分類器(classifier)とは
  - 入力とその所属クラスから何らかの手段で学習し、
  - 未知の入力に対してその所属クラスを答えるもの
- 入力
  - 特徴ベクトル
    - 各特徴の値を格納したベクトル
  - 学習時には所属クラスも
    - つまり教師あり学習
- 出力
  - 所属クラス
- 様々なアルゴリズム
  - パーセプトロン
  - サポートベクトル機械
  - 決定木
  - 単純ベイズ

Jubatusで利用可能な分類器

- Jubatusではパーセプトロンとその亜種を利用可能
  - パーセプトロンは
    - オンラインアルゴリズム
      - データを順次受け取り、その都度内部状態を更新する
    - 線形分類器
  - Koby Crammerと弟子による一連の改良版も実装されている
    - Passive-Aggressive algorithm (PA)
    - Confidence Weighted learning (CW)
    - Adaptive Regularization Of Weight vectors (AROW)、など

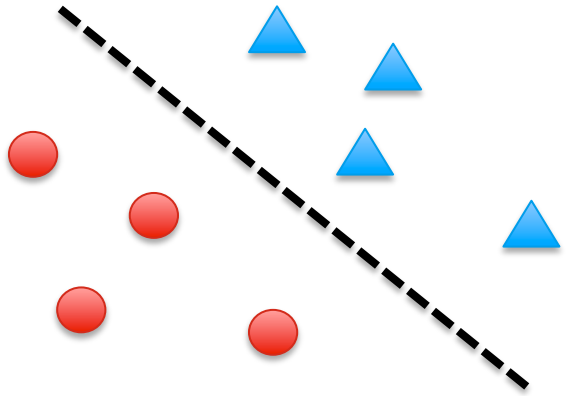
パーセプトロン

- パーセプトロン(perceptron)とは
  - 神経回路における学習過程をモデル化するのが元々の目的
    - なのでオンラインアルゴリズムとなっている
  - 線形分類器
    - 線形分離可能なデータ集合のみ扱える
  - 入力データを各クラスに識別する超平面を求める
- 動き
  - 不正解だった場合に分離平面を回転あるいは平行移動させる
    - 内部的には重みベクトルの更新が行われる

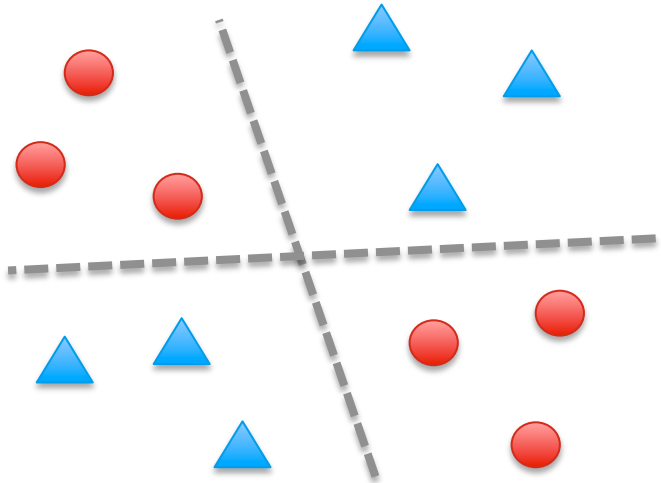


線形分離可能とは

- 定義
  - 二次元平面上の二つの点集合が線形分離可能であるとは、それらが一本の直線で分離できることをいう
  - 一般化して、n次元空間の点集合がn-1次元の超平面で分離できることも線形分離可能という
- 例



線形分離可能

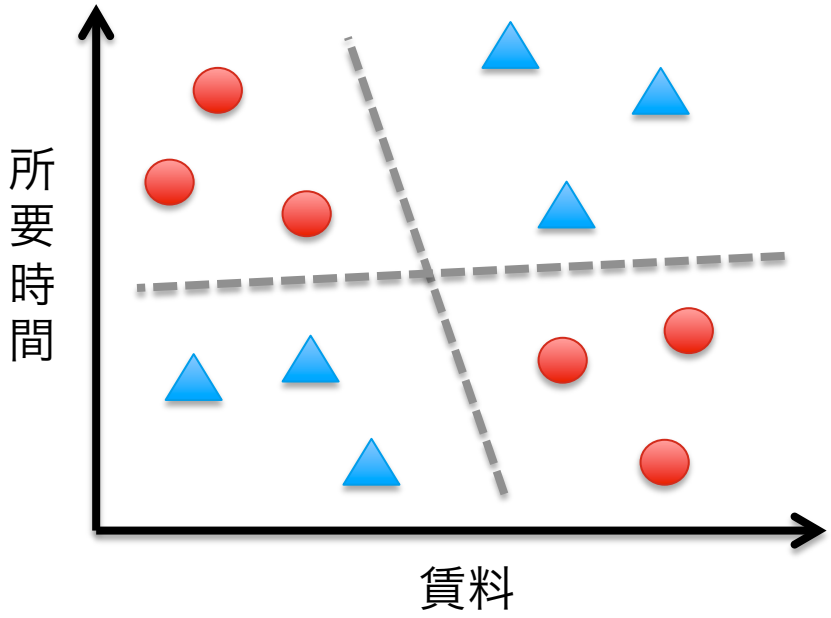


線形分離不可能

- パーセプトロンで分類する際には、対象が線形分離可能でないと収束しない

住宅データは線形分離可能か

- 線形分離不可能！
- 例
  - 特徴として以下を考える
    - 賃料（万円）
    - 駅からの所要時間（分）
  - 利用者が「家賃が高いが駅から近い物件」と「家賃は安いが駅から遠い物件」の両方を好むことはありうる



ダミー変数の導入

- ダミー変数とは
  - 性別のように数量的に表現できない定性的なものを回帰モデルに取り入れるための特別な変数
- 例：就業年数と年収の関係
  - 性別は数値で表されるものではないので、男性のとき1、女性のとき0となるダミー変数を導入する

	就業年数	性別	年収
A	2	1	200
B	10	0	500
C	12	1	700

  - 回帰式
    - 年収 = 50 × 就業年数 + 100 × 性別
- 物件データのダミー変数化
  - 物件の特徴データを一定の幅の区間で区切る
  - 特徴の値に対応する区間のみ1にする
    - 例：賃料12.5万円であれば、12万円から14万円の区間を1にする

重みの伝播

- ダミー変数化の問題点
  - 隣接区間に属する物件であることを情報として保持できない
  - 「賃料11.5万円」の物件と「賃料12.5万円」の物件が無関係になってしまう

	6万	8万	10万	12万	14万	16万	18万
物件A	0	0	1	0	0	0	0
物件B	0	0	0	1	0	0	0

- 対処法：ビットを隣接区間に伝播させる

	6万	8万	10万	12万	14万	16万	18万
物件A	0.33	0.67	1	0.67	0.33	0	0
物件B	0	0.33	0.67	1	0.67	0.33	0

  - 実際には、区間の何%の位置にいるかで伝播度を変化させている
- 注意：多重共線性の問題が生じる
  - 学習が不安定になる可能性がある
  - 割り切りが必要

多重共線性にまつわる問題

- 複数の独立変数同士は無相関という仮定が成り立たないことにより起こる
  - 回帰係数が直感に反する値になることがある
  - 回帰係数が不安定になりやすい
- 例：定期テスト
  - 理科が従属変数、数学・国語が独立変数
  - 数学と国語に高い相関（なんと1.0）があるとする

	理科	数学	国語
生徒A	81	90	90
生徒B	54	60	60

  - 理科 = 1.2 × 数学 – 0.3 × 国語
    - 国語ができるほど理科の得点が下がる？
  - 理科 = 0.8 × 数学 + 0.1 × 国語
  - 理科 = 0.45 × 数学 + 0.45 × 国語
    - どちらも正しいが、係数は大きく異なる

二択型物件推薦システム

- 二物件提示し、利用者にどちらが好みか選んでもらう
- 一定回数選んだら検索条件を生成し、一覧画面へ

分類器による推薦

- 学習
  - 学習は利用者にとって関心がある・ないの二クラスで行う
  - その際に必要な負例（関心がない）を明示的に選択させることはしない
    - その代わりに選択されなかった物件を負例とする
  - これによって操作の簡潔さを保ちつつ選択毎に学習を進められる
- 推薦
  - 関心があると分類された物件を推薦する？
  - しかし学習が十分進むまでに相当回の学習を行う必要がある
    - より効率的に推薦対象を絞り込む必要がある

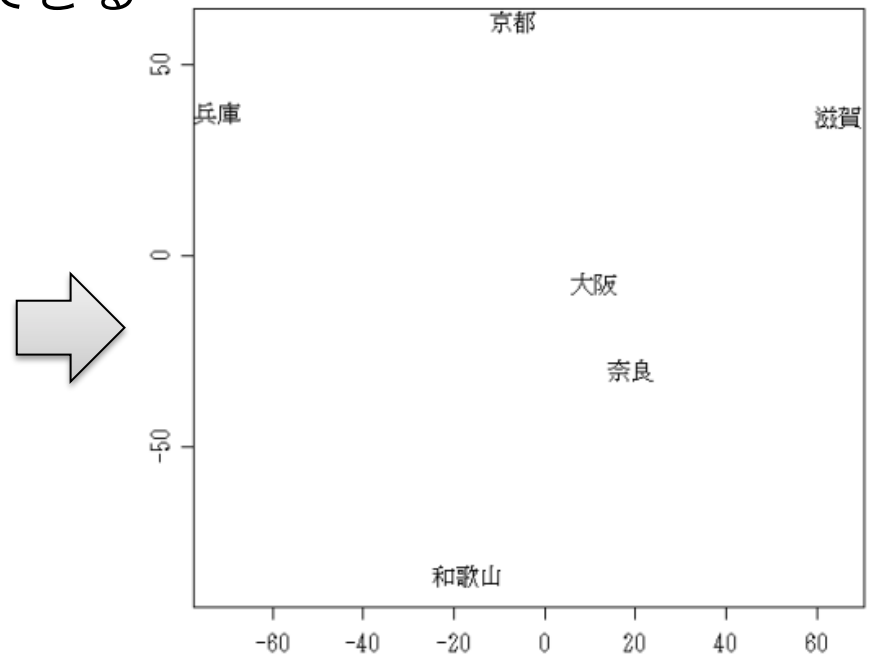
二分探索による探索空間の絞り込み

- 基本的な着想
  - 二分探索によって物件の特徴空間を絞り込んでいけばいいのでは
- 一特徴の場合は単純
  - 0から100の区間に分布しているとき、20と80の物件を提示する
  - 20が選ばれたら、次は10と40の物件を提示する
- 二特徴以上の場合
  - 特徴毎に絞り込んだら手間がかかりすぎる
    - その上既存の検索インタフェースを面倒にしただけの何かである
  - 物件自体を提示（全特徴まとめて提示）したらどうか
    - 賃料はこっちがいいんだけど面積はあっちが、ということがあり得る
    - 全ての特徴の勾配の組み合わせについて提示すると、組み合わせ爆発
  - MDSによって二次元空間に落とし込んでしまえばいいのでは
    - 二次元なら二組四物件提示すれば場合を尽くせる
    - ただしあくまで提示するのは物件（の特徴情報）
    - 提示する物件を、二次元空間中の位置により選定する

多次元尺度構成法(MDS)

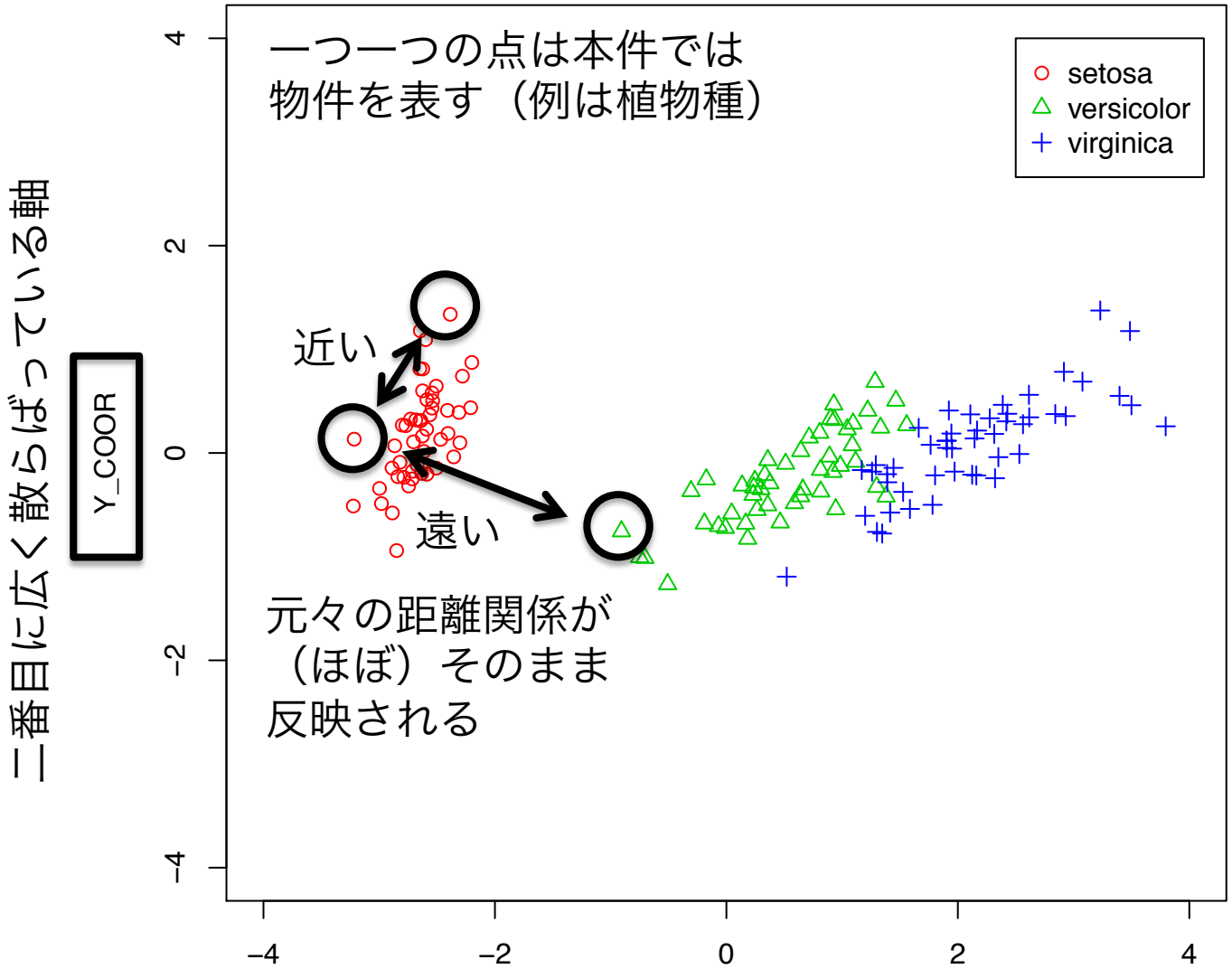
- 分類対象物の関係を低次元空間における点の布置で表現する手法
  - 対象物間の類似性データを元に、似たものは近くに、異なったものは遠くに配置する軸を探す
  - 実用としては二か三次元で表現する
- 例：近畿地方の都市
  - 都市間距離から地図を構成できる

	兵庫	和歌山	大阪	奈良	滋賀	京都
兵庫	0	134	85	116	118	60
和歌山	134	0	68	66	145	141
大阪	85	68	0	32	83	75
奈良	116	66	32	0	79	95
滋賀	118	145	83	79	0	63
京都	60	141	75	95	63	0



http://mjin.doshisha.ac.jp/R/27/27.htmlより転載

MDSの出力結果の見方



軸はデータ点が最も広く
散らばるように選ばれる

最も広く散らばっている軸

MDSの計算手続き

- 距離（非類似度）を算出する
  - まずデータ点同士の距離（の測り方）を定義する
    - 本件ではユークリッド距離を用いている
      - 特徴の値の差の自乗を全て足し合わせたものの平方根
  - 全てのデータ点の組み合わせにつき、距離を求める
    - 距離行列を作る
- 距離の違いを最も良く表現する低次元表現を算出する
  - 距離行列を固有値分解する
  - 大きな固有値に対応する固有ベクトルから順に採用する
    - 二次元の場合、大きい順に二つ
  - データ点と固有ベクトルの内積をとる

MDSによる探索空間の形成

- 利用する特徴の選定
  - ある程度の幅をもって分布する特徴が望ましい
    - 後で回帰分析を行う為
  - 以下の4特徴を利用
    - 賃料
    - 駅からの徒歩分数
    - 床面積
    - 築年数
- 計算手続き
  - 距離行列の生成
    - 各特徴につき正規化
    - ユークリッド距離により距離を算出
  - MDSにより二次元空間に変換

推薦の基本アルゴリズム

- アルゴリズムの特徴
  - 探索空間を切り離して絞り込むことで利用者の好みを推定する
    - 図形的な操作による探索空間の切り離し
      - 「線を引き」「色を塗る」ことで切り離す箇所を決定する
    - 分類器を利用した探索空間の切り離し
      - Jubatusの分類器を利用し、関心のなさそうな部分を切り離す
  - 残された空間から検索条件を導出する
    - 回帰分析によって空間に対応する特徴の値を求める
- 基本的な流れ
  - 利用者に二物件提示する
    - 提示する物件は探索空間中で適度に離れた地点から選ぶ
  - 選ばれなかった物件側を探索空間から切り離す
    - ただし二組四物件を組にして判定する
  - 一定回数選択したら検索条件化する

二回一組での探索空間の切り離し

- なぜ二回一組なのか
  - 二回の選択（二組四物件から）で場合を尽くせる
    - ある軸から二物件選び、それに直交する軸からもう二物件選ぶ
  - 選ばれなかった二物件の側を切り離す
- 効率的に探索を行う為には
  - 分布を代表するような物件を選ばなければならない
    - 極端な物件の間で選択した場合、切り離す部分が多すぎたり少なすぎたりするかもしれない
    - 物件分布の重心を通る、直交する二軸上から選ぶ
  - 物件同士はある程度離れていなければならない
    - 近いと特徴の値も近くなる為、選びにくい
    - 軸方向の物件分布についての、20%点と80%点を基準とする（基準点）
      - 基準点に近い物件から選ぶ

主成分分析(PCA)

- 主成分分析(Principal Component Analysis, PCA)とは
  - データの分散をもっともよく説明するいくつかの軸（潜在変数）によってデータを表現する手法
    - いうなれば、データ集合をどの方向から見たらそれぞれの点をよく区別できるのか、その方向を知る為の方法
    - 「分散をもっともよく説明する軸」というのはつまり、その軸方向でデータのばらつきがもっとも大きくなる軸のこと
    - ばらつきが大きい軸というのは、「要因」である
      - 多変量データの「要因」を探る用途に利用される
- 計算方法
  - 固有値分解による方法
    - 分散共分散行列を固有値分解する
    - 大きな順にいくつかの固有値に対応する固有ベクトルを得る
  - 特異値分解による方法（実装ではこちらを利用）
    - データ行列をそのまま特異値分解する
    - 大きな順にいくつかの特異値と、対応する左特異ベクトルを得る

PCAによる軸の設定

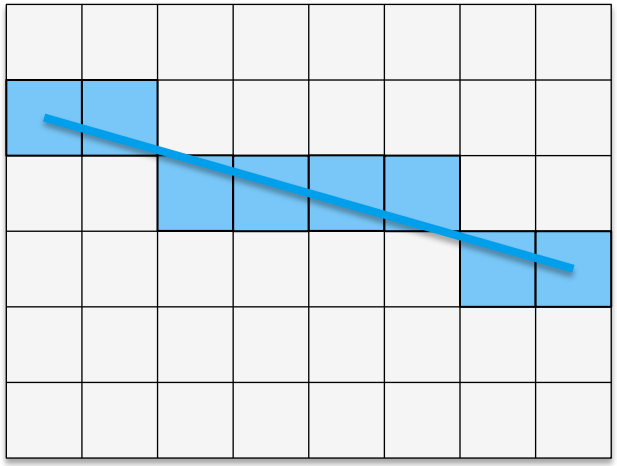
- 目的
  - 二次元空間上で二分探索を行いたい
- そこで、
  - 二分探索を効率的に行う為には、どの方向で見たらデータのばらつきが最も大きくなるのかを知る必要がある
    - ばらつきが大きい方向の二つを提示すれば、より選び易いはず
  - PCAによってばらつきの大きな軸を選べばよい

探索空間のタイル化

- 「探索空間の切り離し」によって実現したいこと
  - 探索空間中の物件に対して、「切り離されていない」「切り離されている」のいずれかを判定する
  - 物件全体から「切り離されていない」物件の集合を抽出する
- 実現方法は様々
  - 切り離し領域を方程式として管理
    - 精緻な判定が可能
  - 物件ごとにフラグを付与
    - 汎用性が高い
  - 探索空間をタイル化し、タイルごとにフラグを付与
    - 計算量が物件数に依存しない
    - 可視化が容易
- タイル化による「切り離し」状態の管理を採用した

Bresenhamの線描画アルゴリズム

- タイル化された空間における「切り離し」
  - 切り離す領域（非凸多角形）を方程式として表現するのは困難
    - 物件が軸上にない場合は近くから選ぶが、最も近い物件でもかなり遠くにある場合がある為、非凸多角形になることがある
  - 「画像の塗りつぶし」が適用可能
- 塗りつぶしによる「切り離し」の2ステップ
  - 境界を引く
    - 複数の線分
  - 外側を塗りつぶす
- 線分の描画
  - Bresenhamアルゴリズム
    - CG分野の古典アルゴリズム（50年もの）
    - 線分の傾きを蓄積してプロットする点を決定
    - 最適化により整数演算のみで実現可能

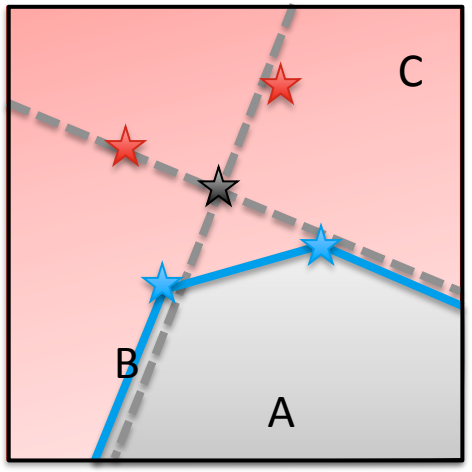


どのようにして「外側」を決定するのか

- 領域の定め方
  - （閉じた）境界により囲む
  - 領域に属する点を見つける
- 外側を直接定義するのは困難
  - 外側に属する点を見つけるのが困難なため
  - 内側が定義できれば、外側は「内側の補集合」として定義できる
- 「内側に属する点」候補
  - 物件分布の重心
  - 利用者が選択した物件の座標（2点）
- 多数決による内側の決定
  - これらの点が常に内側に属するとは限らない
  - 3点の多数決をとることにより安定して内側を決定可能

Flood fillによる塗りつぶし

- Flood fill
  - 与えられた点を起点として、閉領域を「洪水」のように塗りつぶしていくアルゴリズム
  - バケツツールとしておなじみ
- Flood fillによるタイル化された切り離し領域の決定手順
  - 空間全体を色Aで初期化する
  - Bresenhamのアルゴリズムにより境界線を色Bで描画する
  - 内側候補の3点のうち境界線に乗っていない（＝色がAである）1点を取り、Flood fillアルゴリズムにより色Cで塗りつぶす
  - 内側候補の3点の色を調べる
    - Cが多数であれば、AとBが切り離すべき領域
    - Aが多数であれば、BとCが切り離すべき領域



物件分布の重心
選択物件
非選択物件

分類器の重みベクトルの利用

- 重みベクトルには利用者の選択結果が直接反映される
  - 重みベクトルによる評価値が低い物件は切り離しが可能
  - ただし切り離しはタイル単位で行う
  - タイルに属する物件の評価値の平均をタイルの評価値とする
- タイルの切り離し基準
  - 以下の2条件を同時に満たすとき切り離す
    - 評価値が一定の基準値以下
    - 評価値を物件全体の評価値の順位に当てはめたときに下位N%に相当する
  - 採用したパラメータは「基準値＝0」 「N＝20」

検索条件の生成

- 目的
  - 利用者の好みを反映した検索条件を生成
  - それによる物件の一覧表示によって推薦する
- 前提
  - 切り離されずに残った物件は利用者の好みを反映している
- どうやって生成するのか
  - 残った物件の特徴の値を集計
    - 個々の物件の特徴の値はばらつきが大きいので不安定
  - 残った領域から検索条件を生成すれば安定するのでは

回帰分析による特徴の値の推定

- 回帰分析(Regression Analysis)が適用可能
  - 重回帰分析を行う
    - 独立変数：二次元空間の座標
    - 従属変数：特徴の値
  - 特徴の値の「傾き」と「切片」が得られる
    - 座標を特徴の値に、領域を特徴の値の範囲に変換可能
- 特徴の値の推定に用いる領域を定める
  - 以下の条件を満たすタイルの集合
    - 探索により切り離されていない
    - タイル上に物件が存在する
    - 切り離されていない物件の重心から一定の範囲内にある
      - 重心が利用者の好みの「ど真ん中」であると仮定
- 回帰分析の結果に基づき、領域に対応する特徴の値の範囲を検索条件とする

事例2: 動画配信サービス

- 動画視聴後に表示されるお薦め動画をユーザ毎に最適化
  - オンラインでの最適化を行いたい
  - 多腕バンディットで動画のジャンルを決定
- 猫レコメンダ
  - そのままでは似たような動画ばかりが推薦される
  - 推薦結果に多様性を導入する
- 論文
  - 宇田川拓郎, 渡邉卓也, 山中章裕, 室井浩明, 東山昌彦, 小田哲, 小田桐優理, 宮井康宏, 志村誠, 本庄利守, "推薦の粒度を学習の進展に応じて切り替えるmulti-armed banditアルゴリズム", 研究報告数理モデル化と問題解決（MPS）, vol. 2015-MPS-103, no. 30, pp. 1-6, 情報処理学会, 2015.

多腕バンディット問題

- 問題
  - あなたはカジノの客
  - 目の前に複数の（多腕）スロットマシーン（バンディット）
    - それぞれのスロットマシーンの報酬の期待値は異なる
  - どのようにレバー（腕）を引けば効率よく儲けることができるか
- 二律背反
  - 早期に出の良さそうなスロットマシーンに集中してしまうと、報酬の期待値の見積もりを誤る可能性が高くなる
  - いつまでも出の悪いスロットマシーンを選び続けると、儲けが少なくなる
- バランスをとることが大事
  - 探索: 報酬の期待値を見積もる
    - 出の悪そうなスロットマシーンも引いてみる
  - 活用: 儲けを最大化する
    - 出の良さそうなスロットマシーンをたくさん引く

多腕バンディットの特性

- 各種のアルゴリズム（戦略）が存在
  - ε-greedy
  - softmax
  - UCB1
  - Exp3
- 実装は比較的容易
- コールドスタート問題
  - 学習初期は報酬の期待値の推定が不正確

ε-greedyアルゴリズム

- 最も単純な多腕バンディットアルゴリズム
- 経験上、それなりの性能
- 戦略
  - 探索: 確率εでランダムに（一様分布に基づき）各腕を選択
  - 活用: 確率1 – εで最も報酬の期待値の高い腕を選択
  - εを学習の進展に応じて減少させる戦略も

多腕バンディットによる動画の推薦

- 動画視聴後にお薦めする動画を選ぶ
  - 推薦するジャンルを多腕バンディットにより決定する
    - ユーザ毎にジャンルの数だけのスロットマシーンを用意
  - そのジャンルから動画を選び、ユーザに推薦する
- ユーザの行動から報酬を登録する
  - もしその動画がクリックされたら報酬1、されなかったら0
  - ただし「クリックされなかった」ことは直接検出できない場合がある
    - 推薦した時点でいったん報酬0として登録し、クリックされたら1を登録するよう実装
- 次第にそのユーザの好むジャンルの動画がよく推薦されるようになる
  - 報酬の期待値: クリック率
- 実装
  - Jubatusをフレームワークとして利用

推薦の粒度切り替え

- コールドスタート問題に対処する必要がある
  - 数個の動画しか見ていないユーザが多数派となる傾向にある（ロングテール）
  - 多くのユーザで学習が進まない
- 学習の進度に応じて推薦の粒度を切り替える
  - 最初はユーザ群全体について学習した結果から推薦を行う
  - 十分に学習の進んだユーザについてはユーザ毎の推薦を行う
- アルゴリズム
  - 報酬を登録する際に、以下の両方に登録する
    - ユーザ群全体向けスロットマシーン群（ジャンル数）
    - 個々のユーザ向けスロットマシーン群（ユーザ数 × ジャンル数）
  - 推薦回数を記録しておき、
    - 一定回数以上のユーザには個々のユーザ向けの推薦を採用
    - そうでないユーザにはユーザ群全体向けの推薦を採用

シミュレーションによる実験

- ユーザの行動についての仮定
  - アクセス頻度: ガンマ分布
    - 各ユーザのアクセス頻度をガンマ分布からサンプリング
    - ロングテールな分布としている
  - ジャンルの嗜好: ベータ分布
    - 各ジャンルについてのクリック率をベータ分布からサンプリング
    - どのジャンルが好きかについてばらつきが出るようパラメータ設定
- 一定間隔でユーザの嗜好を変動させる
  - 人気のある動画が移り変わる現象を模擬
  - 正規分布に従うノイズを付加
- 推薦の粒度切り替え方式のクリック率は良好

猫レコメンダ

- よく言われるのが...
  - 犬猫を出しておけばクリック率が上がる
  - 本当か？
- 猫動画を推薦したい
  - 猫に関するタグを利用すれば猫動画は検出できる
  - しかし猫動画には似た動画が多い
  - そのままでは似たような動画ばかりが推薦される

推薦に多様性を与える

- Maximal Marginal Relevance (MMR)
  - Jaime Carbonell and Jade Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries", SIGIR, pp. 335–336, 1998.
  - 通常、商品の特徴の類似度で推薦する場合は、似ている商品から順に推薦する
    - ほぼ同一の商品ばかりになる可能性がある
  - 多様性を導入する為の並び替えを行う
    - 上位から順に推薦する商品を決定する
    - 基準となる商品との類似度と、それより上位の推薦が決定した商品群との非類似度の重み付き平均を得点とする
      - 上位の商品群との非類似度は、(-1 x 商品群に含まれる商品との類似度の最大値)を採用する
    - 重みで非類似度をどの程度重視するか調整する
  - 計算量: $O(n^2)$

多様な猫を推薦するレコメンダ

- 方式
  - 動画に付されたタグのビットベクトルを特徴ベクトルとする
  - 動画視聴開始から終了までの間にMMRの計算を行う
    - 視聴中の動画を基準とする
    - 特徴ベクトル間のハミング距離を距離尺度とする
  - 動画視聴終了時、得点上位の猫動画を推薦する
- 計算コストについて
  - $O(n^2)$ではあるが、比較的時間の余裕がある
  - ビットベクトル間のハミング距離は高速に計算できる
    - xor + popcount

事例3: ショッピングモール

- 協調フィルタリングによる推薦
  - 特異値分解により行動履歴を次元圧縮
  - 圧縮した履歴で近傍探索
- 商品の類似度に基づく推薦
  - ユーザの閲覧した商品を基準とする
  - MMRにより多様性を導入
  - ユーザの滞在時間情報も加味
- 論文
  - 執筆予定

協調フィルタリング

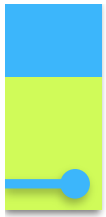
- ユーザの行動履歴によって商品やユーザの類似性を推定
  - 対象とする行動: 商品の閲覧や購買
- いくつかのバリエーション
  - 「この商品を買った人はこのような商品も買っています」
    - 商品ベース協調フィルタリング
    - ユーザ x 商品行列を商品方向に近傍探索して類似商品を推薦
    - Amazon方式
  - 「あなたと似たような商品を買った人はこのような商品も買っています」
    - ユーザベース協調フィルタリング
    - ユーザ x 商品行列をユーザ方向に近傍探索して類似ユーザを抽出した後、それらのユーザによる購入数の多い商品を推薦
    - Amazon方式より歴史は古い
- 現在では商品ベースが主流か？
  - 一般的にはユーザベースの方が省資源

協調フィルタリングによる商品の推薦

- ユーザの行動履歴を蓄積
  - 閲覧や購買によるユーザ x 商品行列を作成する
- 次元圧縮
  - 特異値分解による
  - ノイズが除去されることを期待
- 商品方向に近傍探索を行い推薦する
  - しかし商品数が1000万のオーダなので、リアルタイムな計算による推薦は難しい
- 我々の解: GPGPUによる力押し事前計算
  - 予め全商品について全商品との距離計算による近傍探索を済ませておき、近傍の商品情報を保存しておく
  - 日次バッチとしたいので、一日以内に終わることは必須条件

GPGPUとは

- Graphics Processing Unit (GPU)
  - 主にゲームの映像を描画する為に発展してきたプロセッサ
  - 中身は高並列度のベクトル計算機
    - 最近はCPUにもベクトル演算器が入っているが、それらよりもずっと並列度が高い
- General Purpose GPU (GPGPU)
  - GPUを汎用のベクトル計算機として利用
    - かつてのベクトル型スーパーコンピュータは科学技術計算に利用
    - GPUは安価なので手軽にビジネス用途にも活用できる
  - nVidiaのCUDA（C言語ベース）を利用するのが一般的
    - 実行方式やメモリアクセスパターンについて慎重に検討する必要がある
  - ディープラーニングにより利用場面が拡大
    - 今や機械学習には欠かせない要素となった



knn_finder

- 密ベクトルの近傍探索プログラム
  - https://github.com/sodium-g/knn_finder
  - GPGPUにより高速処理
    - ベクトル間の距離を全ての組み合わせについて計算し、近傍を出力
  - 距離尺度: コサイン距離
    - 近々ユークリッド距離に対応予定
  - 現状説明書きがありません
- kNN CUDAに基本アイデアを依拠
  - http://vincentfpgarcia.github.io/kNN-CUDA/
  - メモリレイアウトとインサーションソートの利用は共通
- knn_finderでの拡張
  - GPUのメモリの許す限りのデータに対して近傍探索可能
    - kNN CUDAにはサイズ制限あり
  - インサーションソートとバイトニックソートを組み合わせて利用

knn_finderの性能

- Amazon Web Services (AWS)で性能計測
  - インスタンス: g2.2xlarge
  - GPU: nVidia GRID K520 (GeForce GTX 680に近い)
  - GPUのメモリ: 4 GB
  - 料金: 時間あたり$0.898
- 計測結果
  - 800万件, 100次元: 約400分
  - 300万件, 250次元: 100分弱
  - バッチ処理が一日以内に終わる！

インサーションソート

- アルゴリズム
  - 列の先頭の要素から順に、自分より前が小さくなるまで前に移動
  - 例: 1 5 3 2 4
    - 1 3 5 2 4
    - 1 3 2 5 4
    - 1 2 3 5 4
    - 1 2 3 4 5
- 単純なソート法
  - 計算量: $O(n^2)$
  - CPUで実行すると遅い
  - GPUの場合にはメモリアクセスパターンとの親和性が高い
    - 複数の列を同時にソートする場合に、各列でほぼ同じインデックスのデータにアクセスすることになる
    - GPUではそうしたメモリアクセスパターンのアルゴリズムを採用しないと高速に実行できない

商品同士の類似度による推薦

- リアルタイムに類似商品を推薦
  - ユーザの閲覧した商品を基準として類似商品を出す
  - 商品の説明文を特徴として利用
  - Latent Dirichlet Allocation (LDA)により説明文のトピック分布を推定
  - トピックベクトルによりJubatusで近傍探索
  - MMRにより多様化を実施
  - ユーザの各商品ページの滞在時間も加味して好みを推定
- クリックからすぐに応答を返す必要がある
  - LDAによる次元圧縮には高速化の意味もある
  - LSHによりさらにベクトルの大きさを圧縮
  - それでも間に合わなければ、推薦対象となる（される）商品を一部に限る
    - 推薦の基準となる商品は全商品

トピック推定

- Latent Dirichlet Allocation (LDA)
  - 文書（商品）のトピック（複数）を推定する
    - ノイズの除去が期待できる
  - 次元圧縮の手段でもある
    - 単語数次元からトピック数次元に圧縮できる
  - 特異値分解に似ているが、確率的モデルに基づく推定である点が異なる
  - 以下が得られる
    - 文書のトピック分布
    - 単語のトピック分布

LDAの実装

- pldaを改修して利用
  - https://code.google.com/archive/p/plda/
  - collapsed Gibbs samplingにより高速にトピック分布を推定
    - 各文書の各単語のトピックをサンプリングしていく
  - MPIにより複数プロセス（あるいはノード）での実行が可能
    - 文書を各プロセスに分散配置
    - 各プロセスのサンプリング結果をイテレーション単位で合併
      - 従ってシングルプロセス時より精度は落ちる
- pldaは単語のトピック分布しか出力しない
  - 文書（商品）のトピック分布を出力するように改修

トピック分布の補完

- トピック分布の計算出来ていない商品が存在する
  - LDAはバッチ処理だが、商品は随時追加される為
- 全ての商品について説明文は取得できる
  - 説明文から単語を抽出する
- 商品説明文のトピック分布を推定する
  - pldaの出力した単語のトピック分布を元に説明文中の単語についてサンプリングを行い、トピック分布を推定する
- 推定したトピック分布を基準として推薦を行う

滞在時間の加味

- 滞在時間に基づいてユーザの好みを推定したい
  - 基本的には興味のあるページはより長く見るはず
  - ウェブページの滞在時間は非常にばらつきが大きい
    - ページを数秒単位で移動することもあれば、閲覧したまま一日以上放置することもある
  - 外れ値を除外する必要がある
- ユーザ単位で対数正規分布を仮定して分布を推定
  - 95%点より外側を除外
  - 分布のパラメータとして中央値とMean Absolute Deviation (MAD)を採用
  - MADとは
    - 中央値を元にした標準偏差のようなもの
    - （平均を元にした）標準偏差よりも頑健

複数要素の組み合わせによるスコアリング

- 元々のMMRは...
  - 類似度と非類似度の重み付き平均
- MMRを拡張
  - 以下の重み付き平均
    - 類似度
    - 非類似度
    - 滞在時間
    - 何個前の商品を基準に推薦された商品か
  - 各要素は事前に正規化しておく
  - 重みを変化させることで最適化できる

多様化についての検討

- 性能的に推薦対象にできる商品が限定される
  - $O(n^2)$の壁
  - 事前に推薦対象とする商品の絞り込みを行う必要
- 絞り込みの過程で類似商品が削られ推薦結果の多様性が一定程度確保できる
  - 結局オンラインでの多様化は必要ない？

推薦のシーケンス

- ユーザが商品を閲覧
- 閲覧された商品の近傍の商品リストを取得
  - Jubatusにより近傍探索
  - 距離尺度: Hellinger距離
  - 平方根をとればコサイン距離と一致するのでLSHを利用
- ユーザの滞在時間をKey Value Store (KVS)より取得
  - 前回閲覧した商品の滞在時間がここで判明する
- 前回閲覧時のスコア上位の商品リストと合併した上でスコアリングを行う
  - 拡張MMRによりスコアを算出
    - 類似度、非類似度、滞在時間、何個前の商品を基準に推薦された商品か
  - 何個前の商品を基準に推薦された商品かによる得点が次第に減っていくので、より以前に推薦された商品は順位が下がっていく
- スコア上位の商品を推薦

スケーラビリティの確保

- AWSのオートスケーリングを利用
  - ノードが状態を持たなければオートスケーリング可能
  - ノード起動時にはデータファイルを転送する
  - ノード停止時キャッシュ類は破棄
- 状態はKVSに保存
  - ユーザの閲覧した商品
  - 滞在時間
- 推薦プログラムの動作
  - KVSからの滞在時間の取得時、自分の持っていない閲覧商品と滞在時間の履歴を補完する

おわりに

- 事例を基にレコメンデーションの実際と関連技術を紹介
  - 住宅情報サービス
  - 動画配信サービス
  - ショッピングモール
- 他にも弊社は様々なシステムに関わっています
  - Gbps級ストリーム処理基盤
  - 特定用途向けルールエンジン（DSL処理系）
- ここまでお付き合いいただきありがとうございました